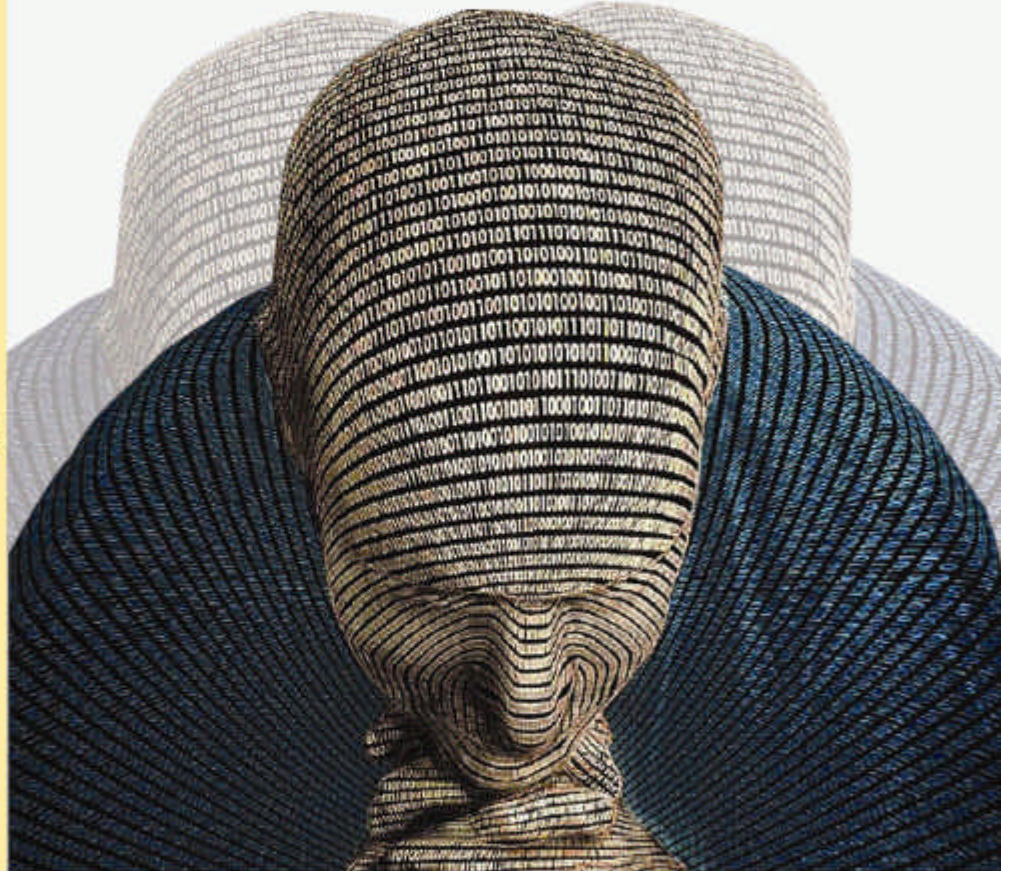


BAŞTAN SONA

VISUAL BASIC 6

KONULAR

- Kurulum
- Çalışma ortamı
- Temeller
- Kontroller
- Formlar
- Fonksiyonlar
- ActiveX bileşenleri
- Dosyalar
- Veritabanı yaratmak
- ADO ve erişim yöntemleri
- Raporlama
- DLL ve API
- Ve daha birçok konu...



```
Bicimle()  
With Text1  
.Text = 30  
.Font.Bold = True  
.Interior.Color = RGB(255, 255, 0)
```



İÇİNDEKİLER

Sayfa

VISUAL BASIC	1
1-VISUAL BASIC'IN ÖZELLİKLERİ	3
NESNEYE DAYALI PROGRAMLAMA	3
OLAY -TEMELLI PROGRAMLAMA	3
TÜMLESİK PROGRAM GELİSTİRME ORTAMI	4
BİR VISUAL BASIC PROGRAMININ KISIMLARI	4
BİLGİSAYAR VE PROGRAMLAMA NEDİR?	4
PROGRAM NEDİR?	5
PROGRAMLAMA DİLİ NEDİR?	5
PROGRAMLAMANIN TARİHİ	6
PROGRAM GELİSRİRME SÜRECİ	6
İYİ BİR PROGRAMIN NİTELİKLERİ	7
PROGRAM GELİSTİRME ADIMLARI	7
2-VB 60 KURULUMU	9
Konfigürasyonları	15
SÜRÜMLER	16
NET nedir?	16
Visual Basic NET	16
Common Language Runtime Destegi	16
Nesne Yönelimli Tasarım	16
Bos Düğüm	17
Diğer Yeni Özellikler	17
Microsoft® NET'in Avantajları	17
Gelecek Kusak İs Entegrasyonu	17
İşletmelere Yönelik Kazançlar	18
Microsoft Net ile İlgili Standartlar	18
Microsoft Net Framework	18
XML Web Servisleri	18
Hailstorm	18
Akıllı Cihazlar	19
İstemci Yazılımları	19
Visual Basic 60 Uygulamalarını Visual Basic NET'e Güncelleme	19
Visual Basic NET Upgrade Tool	20
Günümüzün Uygulamalarını Güncellemek için Mimarî Rehber	21
Visual Basic 60'ın Sistemden Kaldırılması	22
3-VISUAL BASIC'TE ÇALIŞMA	25
BASLATMA	25
NEW SEKMESİ	25
EXISTING SEKMESİ	26
RECENT SEKMESİ	27
ÇALIŞMA ORTAMI	27
FORMLAR	27

KOD EDITÖRÜ	28
ARAÇ KUTUSU	29
BİR ARAÇ KUTUSU DENETİMİNİN ÇIKARTILMASI	30
PROJECT PENCERESİ	30
ÖZELLİKLER (PROPERTIES) PENCERESİ	30
FORM LAYOUT PENCERESİ	32
OBJECT BROWSER	33
ORTAMIN DÜZENLENMESİ	34
FILE MENÜSÜ	34
EDIT MENÜSÜ	35
VIEW MENÜSÜ	36
PROJECT MENÜSÜ	37
FORMAT MENÜSÜ	39
DEBUG MENÜSÜ	40
TOOLS MENÜSÜ	41
ADD-INS MENÜSÜ	43
WINDOW MENÜSÜ	44
HELP MENÜSÜ	44
4-VISUAL BASIC TEMELLERİ	45
Bir Visual Basic Programının Yapısı	45
Proje Kavramı	45
Projeye Başlama	45
Project Explorer	46
Formlar ve Modüller	46
Form Modülleri	47
Standart Modüller	48
Class Modüller	49
Bir Procedure' un Yazılması	49
Bir Procedure' un Yapısı	51
Genel Yordamlar	52
Olay Yordamları	53
Fonksiyonlar	54
Bir fonksiyon yordamının yapısı	54
Yeni Bir Yordam Yaratma	55
Yordamların Çağırılması	56
Parametre Kullanımı	57
Diğer Formlardaki Yordamları Çağırma	58
Class Modüllerindeki Yordamları Çağırma	58
Standart Modüllerdeki Yordamları Çağırma	58
Standart Modüllerdeki Yordamları Çağırma	58
Kod Yazma Kuralları	59
Otomatik Kod Tamamlama	59
KOD EDITÖRÜ DÜZENLEME SEÇENEKLERİ	60
Pencere Düzenlemeleri	61
Kod Penceresi Kisayolları	61

Menü Kisayollari	62
Koda Açıklamalar Eklemek	64
Uzun Bir Satiri Birçok Satira Bölmek	65
Birçok Deyimin Bir Satir Olarak Bileştirilmesi	65
Visual Basic'te Adlandırma	65
Deyimler	66
Aritmetik İşlemler	67
İşleçlerin (Operatörlerin) Öncelikleri	69
Metotlar	69
Özellikler	70
With Deyimi	71
Olaylar	72
Veri Elemanlarının Tanımlamaları	73
Degiskenler	73
Degiskenlerin Tanımlanması	73
Degisken Tanımlamada Hatalar	75
Açık Tanımlama	75
Açık Tanımlamaya Zorlama	76
Kapalı Tanımlamalar	76
Degiskenler Nerede Tanımlanacak?	77
Kapsama Alanı	78
Yerel Degiskenler	78
Static Degisken Tanımlama	83
Sabitler	84
Option Deyimleri	85
Veri Tipleri	87
Veri Tiplerinin Dönüştürülmesi	91
Veri Tiplerinin Degisken Adlarıyla Kullanımı	97
Diziler (Arrays)	97
Dinamik bir dizi tanımlamak	100
Dizilerin Alt ve Üst Sınırlarıyla Tanımlanması	101
Diğer Dizileri İçeren Diziler	102
Dizilerde Option Base Deyimi Kullanımı	103
Kullanıcı Tanımlı Veriler	104
İç İçer Olan Veri Yapılar	105
Program Denetimi	107
Döngüler	108
DoLoop Deyimi	109
While ve Until Sistemi	114
ForNext Deyimi	115
Dizileri İşlemek İçin Döngülerin Kullanılması	118
Karar Yapıları	119
IfThenElse Deyimi	120
Select Case Deyimi	124
GoTo Deyimi	129

Exit Deyimi	130
End Deyimi	131
Operatörler	132
5-FONKSIYONLAR	135
A STRING FONKSIYONLAR	135
Asc() Fonksiyonu	135
Chr() Fonksiyonu	136
IsArray() Fonksiyonu	136
Instr() Fonksiyonu	136
Len () Fonksiyonu	137
Left(), Left\$() Fonksiyonu	137
Lcase(), Lcase\$() Fonksiyonu	138
Ltrim, Trim, Rtrim Fonksiyonu	138
Mid, Mid\$ () Fonksiyonu	139
Right(), Right\$() Fonksiyonu	139
Space, Space\$ () Fonksiyonu	140
String, String\$ () Fonksiyonu	140
StrComp() Fonksiyonu	141
Tab, Spc () Fonksiyonu	141
Ucase(), Ucase\$() Fonksiyonu	142
B MATEMATİKSEL FONKSIYONLAR	142
Abs() Fonksiyonu	142
Atn() Fonksiyonu	143
Cos() Fonksiyonu	143
Exp() Fonksiyonu	144
Fix() Fonksiyonu	144
Hex() Fonksiyonu	144
Int() Fonksiyonu	145
Log() Fonksiyonu	145
Oct() Fonksiyonu	146
Rnd() Fonksiyonu	146
Sgn() Fonksiyonu	147
Sin () Fonksiyonu	147
Sqr() Fonksiyonu	148
Tan() Fonksiyonu	148
Val() Fonksiyonu	149
CDEĞİSKEN KONTROL FONKSIYONLARI	149
IsDate() Fonksiyonu	149
IsNull() Fonksiyonu	150
Lbound() Fonksiyonu	150
TypeName() Fonksiyonu	151
VarType() Fonksiyonu	151
D DISK İSLEM FONKSIYONLARI	152
ChDir() Fonksiyonu	152
ChDrive Fonksiyonu	153

CurDir Fonksiyonu	153
Dir Fonksiyonu	153
Environ Fonksiyonu	154
FileDateTime Fonksiyonu	154
FileLen Fonksiyonu	155
FileCopy Fonksiyonu	155
Get Attr Fonksiyonu	155
Kill Fonksiyonu	156
MkDir() Fonksiyonu	157
Name Fonksiyonu	157
Rmdir Fonksiyonu	157
EDOSYALAMA FONKSIYONLARI	158
Open Fonksiyonu	158
Write ve Print Fonksiyonu	158
Input Fonksiyonu	159
Close Fonksiyonu	159
FileAttr Fonksiyonu	159
FreeFile Fonksiyonu	160
Loc Fonksiyonu	161
Lof Fonksiyonu	161
Seek Fonksiyonu	162
EOF Fonksiyonu	162
F Format Dönüşüm Fonksiyonlari	163
Array () Fonksiyonu	163
Ccur () Fonksiyonu	163
Cdbl () Fonksiyonu	164
CInt () Fonksiyonu	164
CLng () Fonksiyonu	164
CSng () Fonksiyonu	165
CStr () Fonksiyonu	165
Cvar () Fonksiyonu	166
DataValue () Fonksiyonu	166
Format,Format\$ () Fonksiyonu	167
Str,Str\$ () Fonksiyonu	167
TimeSerial () Fonksiyonu	167
G Yükleme Fonksiyonlari	168
End Fonksiyonu	168
Exit Sub Fonksiyonu	169
Inputbox Fonksiyonu	169
Load Fonksiyonu	169
LoadPicture Fonksiyonu	169
Msgbox Fonksiyonu	170
Show Fonksiyonu	171
Unload Fonksiyonu	171
6-FORM KULLANIMI	173

SDI FORMLAR	175
FORMA AIT BAZI ÖNEMLİ OLAYLAR	177
FORMUN METHODLARI	179
MDI FORMLAR	184
Bir MDI Uygulama Yaratmak	184
Çalışma Zamanında MDI Formun Özellikleri	184
Alt Formların Kullanımı	187
MDI FORMDA MENÜLER	189
WINDOW(PENCERE) MENÜSÜ YARATMAK	190
MDIFORMUN BAZI OLAYLARI	191
MODULLER	193
7-VISUAL BASIC KONTROLLERİ	195
A-DEFAULT COMPENENT LIST	195
1-Label(Etiket)	195
Label Nesnesinin Özellikleri	195
Label Nesnesinin Bazı Önemli Olayları	196
Label Nesnesinin Methodları	197
2-Textbox(Metin Kutusu)	202
Textbox Nesnesinin Özellikleri	203
Textbox Nesnesinin Bazı Önemli Olayları	204
Textbox Nesnesinin Methodları	205
3-Command Button(Komut Düğmesi)	208
Command Button Nesnesinin Bazı Önemli Olayları	209
Command Button Nesnesinin Methodları	210
4-Check Box(Isaret Kutusu)	211
Check Box Nesnesinin Özellikleri	211
Checkbox Nesnesinin Bazı Önemli Olayları	212
Checkbox Nesnesinin Methodları	213
5-Option Button(Seçenek Düğmesi)	214
Optionbutton Nesnesinin Özellikleri	215
Option Button Nesnesinin Bazı Önemli Olaylar	216
Option Button Nesnesinin Methodları	217
6-Frame Control(Çerçeve)	218
Frame Nesnesini Özellikleri	219
Frame Nesnesinin Önemli Olayları	219
Frame Nesnesinin Methodları	220
7-Picturebox(Resim Kutusu):	221
Picturebox Nesnesinin Özellikleri	222
Picturebox Nesnesini Bazı Önemli Olayları	223
Picturebox Nesnesinin Methodları	225
8-Combobox	226
Combobox Nesnesini Özellikleri	227
Combobox'ın Methodları	228
Combobox'ın Olayları	232
9-Listbox	236

Listbox Nesnesini Özellikleri	237
Listbox'ın Metotlari	238
Listbox'ın Olaylari	238
10-HScrollbar	243
HScrollbar Nesnesini Özellikleri	244
HScroll'ın Metotlari	244
HScroll'ın Olaylari	245
11-VScrollbar	246
VScrollbar Nesnesini Özellikleri	246
VScroll'ın Metotlari	247
VScroll'ın Olaylari	247
12-Timer	248
Timer Nesnesini Özellikleri	248
Timer'ın Metotlari	249
Timer'ın Olaylari	249
13-Drivelistbox	249
Drivelistbox Nesnesini Özellikleri	250
Drivelistbox'ın Metotlari	251
Drivelistbox'ın Olaylari	251
14-Filelistbox	251
Filelistbox Nesnesini Özellikleri	252
Filelistbox'ın Metotlari	252
Filelistbox'ın Olaylari	253
15-Dirlistbox	253
Dirlistbiox Nesnesini Özellikleri	254
Dirlistbox'ın Metotlari	254
Dirlistbox'ın Olaylari	254
16-Shape	255
Shape Nesnesini Özellikleri	255
Shape'ın Metotlari	255
17-Line	257
Line Nesnesini Özellikleri	257
Line'ın Metotlari	258
18-Image Kontrolü	258
Image Properties (Özellikler)	259
Image Methods (Metodlar)	260
Image Events (Olaylar)	263
19-Ole Kontrolü	264
Ole Properties (Özellikler)	266
Ole Methodlari	267
Ole Events (Olaylar)	274
20-Data Kontrolü	275
Data Properties	275
Data Methods	276
Data Events (Olaylar)	280

B-Microsoft Common Dialog Kontrolü	281
CommonDialog Properties (Özellikleri)	282
C-Microsoft MSFlexGrid (Izgara Kontrolü)	296
MSFlexGrid Properties (MSFlexGrid Özellikleri)	297
MSFlexGrid Metodlari (Methods)	300
MSFlexGrid Events (Olaylar)	304
D-Microsoft Forms 20 object Library	306
MultiPage	306
Multipage Fonksiyonlari	307
ToogleButton	308
ToggleButton Fonksiyonlari	310
Tabstrip	311
TabStrip Fonksiyonlari	312
SpinButton	313
SpinButton Fonksiyonlari	315
E-Microsoft Hrerarchikal FilexGrid Control	316
MshflexGrid	316
MSHFlexGrid Fonksiyonlari	319
F-Microsoft Internet Transfer Control	321
Inet	321
Inet Fonksiyonlari	322
G-Microsoft MAPI Controls	323
Mapi Session	323
Mapi Fonksiyonlari	323
Mapi Messages	324
Mapimessages Fonksiyonlari	325
H-Microsoft Masked Edit Control	326
MasketBox	326
MaskedTextBox Fonksiyonlari	328
I-Microsoft Windows Common Controls 60	329
TOOLBAR	329
Toolbar Fonksiyonlari	331
STATUS BAR	332
StatusBar Fonksiyonlari	333
PROGRESS BAR	334
ProgressBar Fonksiyonlari	335
TREEVIEW	336
TreeView Fonksiyonlari	338
LISTVIEW	339
ListView Fonksiyonlari	341
IMAGELIST	342
ImageList Fonksiyonlari	343
SLIDER	344
Slider Fonksiyonlari	345
IMAGECOMBO	346

ImageCombo Fonksiyonlari	348
I-MICROSOFT MULTIMEDIA DTCs	349
Page Transitions	349
Page Transitions Fonksiyonlari	350
Time Lines	351
Time Lines Fonksiyonlari	352
J-MICROSOFT REMOTEDATA CONTROL	353
MSRDC	353
MSRDC Fonksiyonlari	355
K-MICROSOFT RICH TEXTBOX CONTROLS	357
RichTextBox	357
RichTextBox Fonksiyonlari	359
L-MICROSOFT SYSINFO CONTROLS	362
SysInfo	362
SysInfo Fonksiyonlari	362
M-MICROSOFT TABBED DIALOG CONTROL	364
SSTab	364
SSTab Fonksiyonlari	366
N-MICROSOFT WINSOCK CONTROLS	367
Winsock	367
Winsock Fonksiyonlari	367
8-DOSYA ISLEMLERİ	369
Rasgele Erisimli Dosyalarda Okuma ve Yazma Islemi	370
Sirali Erisimli Dosyalarda Okuma ve Yazma Islemi	371
Arama Modülü	377
Düzeltilme Modülü	379
Silme Modülü	382
Listeleme Modülü	384
9-PROGRAM KODU YAZARAK VERİ TABANI DOSYASI HAZIRLAMAK	389
MDB Dosyasina Yeni Kayit Yazmak	404
Veri Tabani Dosyasindaki Kayitlarin Arasinda Dolasmak	407
Veri Tabani Dosyasinda Kayit Aramak	410
Kayitlarda Degisiklik Yapmak	415
Veri Tabani Dosyasindan Kayit Silmek	418
10-VERİ ERİSİM YÖNTEMLERİ	432
Vbasic'de veritabanlari seçenekleri	437
DAO	437
DAO ve Jet	438
Dao nesnelerinin sematik gösterimi	439
DBEngine	440
Properties	440
Metodlari	441
Workspace	442
Metodlari	442

Database	443
Metodlari	443
TableDef	444
Properties	444
Metodlari	445
Recordset Degiskeni Olusturmak	446
Properties	447
Methods	450
Field	453
Properties	453
DAO'nun Forma Eklenmesi	455
Dao ayarlari	456
Text'lerin özellikleri	458
Tablodaki kayitlar arasinda dolasma k	463
Tabloya Yeni Kayit Girmek ve Kayit Silmek	466
Kayit arama silme düzeltme içeren bir prog	472
Stok yöneticisi programi	479
RDO	508
RDO 20'in Yenilikleri	508
Bir Baglantiyi Açmak	509
Veritabani Degistirme	509
Remote Data Control'un Kullanimi	510
RDO DAO karsilastirilmesi	513
RDO'nun zayıf Yönleri	514
ODBC	514
ODBC Mimarisi	515
ODBC Verisine Erisim	517
Adim Adim Windows 9x te DSN Tanimi	519
ADO Veri Erisim Teknigi	521
ADO Data Control'un Proje Çalışırken Düzenlenmesi	533
DSN kullanilmadan Baglantinin Yapilmasi	534
DataGrid ,DataList,DataCombo Bilesenleri	537
ADO (ActiveX Data Objects) Temelleri	542
OLEDB Hakkinda	543
Connection Nesnesi	545
Cursor Location Özelligi	546
DATA Environment	560
Örnek Programlar Ve Açiklamalari	576
11-CYRISTAL REPORTS KULLANIMI	598
Crystal Reports Menüleri	598
FILE	598
EDIT	599
INSERT	599
FORMAT	600
DATABASE	601

REPORT	603
WINDOW	604
HELP	604
CRYSTAL REPORTS İLE ÖRNEK RAPOR HAZIRLAMA	605
DOSYA İSLEMLERİ	606
Dosyaya kayıt yazma	606
Dosyadan kayıt okuma	611
Kayıt düzeltme	612
Dosyadan kayıt silme	612
Menü hazırlama	613
12-VB'de GRAFİK KULLANIMI	620
ChartType	622
ColumnCount	627
RowCount	627
Row,Column	627
Data	627
ChartData	627
ColumnLabel, RowLabel	628
ShowLegend	628
AllowDynamicRotation	628
AutoIncrement	629
TitleText,FootnoteText	630
Title,Footnote	630
VtFont	630
Location	630
LocationVisible	630
LocationLocationType	630
TextLayout	630
TextLayoutOrientation	630
TextLayoutHorzAligment	630
TextLayoutVtVerticalAligment	631
Chart Metodlari (Methods)	631
EditCopy	631
EditPaste	631
13-MICROSOFT OFFICE'E BAĞLANMAK	632
Bir İşletme Bilgi Sistemi Olusturmak	632
İşletme Bilgi Sistemlerinin Kullanım Alanları	632
Siparis girme sistemi	632
İnsan kaynakları yönetimi	632
Mali analiz aracı	632
Envanter yönetim bilgi sistemi	632
Yönetim bilgi sistemi	632
Proje yönetim sistemi	633
OLE (Nesne Bağlama ve Gömme Elemanı)	633
OLE Teknolojisinin Gelisimi	634

DDE Teknolojisi (Dynamic Data Exchange)	634
OLE 10	634
Bir Word dokümanı içine gömülü Excel dokümanı	635
OLE 20	635
COM Teknolojisi (component object model)	635
UUID Yapısı (universally unique identifier)	636
Yapılandırılmış Doküman Saklama Yöntemi (OLE Structured Storage)	636
OLE Özdevinim Yapısı (OLE Automation)	636
OLE Denetimini Kullanmak	637
Automation'ı Kullanarak Uygulama Nesnelerini Programlamak	643
Visual Basic'te Automation'ı Kullanmak	644
COM Teknolojisi	644
Nesne Teknolojisi	644
Visual Basic ile Bir Client Yaratmak	645
REFERANSLARINDÜZENLENMESİ	645
NESNELERİN TANIMLANMASI	646
NESNELERİN YARATILMASI	646
Application Nesnesi ile Çalışmak	648
Microsoft Word Nesne Modeli	648
Document Nesnesi ile Çalışmak	649
Visual Basic Object Browser	651
Word Nesnelerini Görmek için Object Browser Kullanımı	651
Visual Basic'ten Word'ü Kullanmak	655
Yazım Denetim programını çalıştırın	659
Visual Basic'ten Excel'i Kullanmak	660
İpotek programını çalıştırın	662
Visual Basic'ten Outlook'u Otomatikleştirmek	663
Visual Basic'ten PowerPoint'i Kullanmak	666
14-WEB İÇİN DİNAMİK HTML SAYFALARI TASARLAMAK	670
Dinamik HTML	670
Yeni Bir Programlama Teknolojisi	670
Adım Adım DHTML Program Gelistirmek	671
DHTML Page Designer ile Çalışmaya Başlamak	672
Yeni Bir DHTML Uygulaması Açmak	672
DHTML Page Özellikleri	674
Bir HTML Sayfasına Metin Ekleme	675
Page Designer'daki Metni Biçimlendirme	676
Karakterleri Ayırmak için SPAN İmlerini Kullanmak	677
Properties Penceresiyle ID Öznitelikleri Atama	678
Baska bir HTML sayfasına köprü yaratmak	679
WebSansim Projesini Kaydetme	680
DHTML Uygulamasını Çalıştırma	680
Microsoft Word'de HTML Belgeleri Yaratmak	681
Sanslihtm Yardım Dosyasını Yaratmak İçin Word'ü Kullanma	682

DHTML Sayfalarına Öğeler ve ActiveX Denetimleri Ekleme	683
Araç kutusu Öğeleriyle Çalışmaya Başlamak	683
DHTML Araç Kutusunu Belgelendirmek	684
Button	684
SubmitButton	684
ResetButton	685
TextField	685
TextArea	685
PasswordField	686
Option	686
Checkbox	686
Select	686
Image	686
Hyperlink	687
HorizontalRule	687
FileUpload	687
HiddenFile	687
InputImage	687
List	688
Öğeleri Yaratmak ve Özelleştirmek	688
Websansim Uygulamasına Öğeler Ekleme	688
DHTML Uygulamanızdaki Dosyaları Yeniden Adlandırma	689
Sayfadaki Bir Metin Öğesini Silme	689
Sayfaya Bir Image Öğesi Ekleme	690
Sayfaya Bir Button Öğesi Ekleme	690
DHTML Sayfasına ActiveX Denetimleri Ekleme	691
Araç Kutusuna Bir ActiveX Denetimi Ekleme	691
Sayfada Bir Multimedia MCI Denetimi Yaratma	691
DHTML Öğeleri İçin Olay Yordamları Yaratmak	692
DHTMLPage_Load Olay Yordamını Yaratma	692
DHTMLPage_Load Kodunu İnceleme	693
Button1_OnClick Olay Yordamını Yaratma	694
Button1_OnClick Kodunu İncelemek	694
BenimDHTML7 Uygulamasını Çalıştırma	695
15-VISUAL BASIC ERRORLER	699
Hata Denetimi	699
Daha Az Hatayla Karşılaşmak	701
Otomatik Tamamlamalar	702
Editör Sekmesi seçenekleri	702
Kod düzenlemeleri	702
Pencere Düzenlemeleri	703
Kod düzenlemeleri	703
Çalışma Zamanı Hataları	706
Tuzaklanabilir Hatalar	706
Hata Düzeltme Araçları	706

Locals Window, Immediate Window ve Watch Window	709
Locals Window	709
Immediate Window	710
Watch Window	710
Programı Kaldığı Yerden İşletme	711
Kesme Noktası (Toggle Breakpoint)	712
Kodun Yeniden Çalıştırılması	712
Çağrılar Diyalog Kutusu	712
On Error Deyimi	712
Hata Denetimi İçin Yöntemler	714
Resume Deyimi	715
Err Nesnesi	716
End ve Stop Deyimi	717
Ek Hata Mesajları, Sebepleri ve Çözümleri	717
Çalışma Zamanı Hata Mesajları	717
Program kodu yazarken karşılaşılan hata mesajları	718
Locals Window	731
Immediate Window	731
Watch Window	731
16-API Nedir ?	732
VB'de API Tanımı	733
API Tanım Dosyası	735
Windows ve Sistem hakkında Bilgi Veren API 'ler	737
Bos bellek miktarını öğrenmek	738
Mikroişlemci tipini ve sayısını öğrenmek	739
dwProcessorType	739
dwNumberOfProcessors	740
Windows versiyonunu ve ortamını öğrenmek	740
Klavye kod sayfasını öğrenmek	741
Klavye tipini öğrenmek	742
Modern Inkey\$	742
Windows dizinini öğrenmek	743
LpBuffer	743
nSize	743
System dizinini öğrenmek	744
Windows Ayarları Hakkında Bilgi	744
GetSystemMetrics	745
Disk Sürücüler Hakkında Bilgi Veren Apiler	749
Sistemdeki disk sürücülerini öğrenmek	749
Volume Bilgisini Öğrenme	751
GetVolumeInformation	751
nVolumeNameSize	751
lpVolumeSerialNumber	751
lpMaximumComponentLength	751
lpFileSystemFlags	751

IpFile System Name Buffer	752
Sürücü Boyutu	753
IpTotalNumberOfBytes	754
Windows Altında Çalışan Bütün Formlara Hükmetmek	756
Çalışan formların handle numaralarını bulmak	756
Çalışan formların başlıklarını değiştirmek	757
Çalışan formların durumunu değiştirmek	760
Form simge durumunda mı?	760
Form ekranı kaplamış mı?	760
Form gizli mi?	761
Masa Üstünün handle numarası	762
Formu en üstte tutmak ve gizlemek	762
Form Başlığının rengini değiştirmek	763
Windowsu Kapatmak	764
Formlara Sekil Vermek	765
CreatePolygonRgn	766
Formu Taskbarda Gizleme	769
Menülerle ilgili API'ler	769
Kontrol-System Menüsü	770
System Menusunun Handle Numarasını Öğrenmek	770
System Menüüne Yeni Seçenekler Ekleme	771
Mesaj Kuyrugundaki Mesajı Almak	772
Menu çubuğunun handle numarasını öğrenmek	777
AltMenülerin handle numarasını öğrenmek	777
Menülerin alt menü sayısını öğrenmek	779
Menülere resim eklemek	780
Kapat Düğmesini Pasif Yapma	783
Sikistirilmiş Dosyalarla İlgili API'ler	784
Sikistirilmiş Dosyayı Açmak ve Hedef Dosyayı Olusturmak	784
Sikistirilmiş Dosyayı Hedef Dosyaya Açmak	786
Dosyaları Kapatmak	787
Ses kartının API'lerle kullanılması	788
Wav dosyalarını çalmak	789
Wav dosyalarını çalacak donanım var mı?	789
Wav dosyalarını çalacak donanımın kapasitesi nedir?	790
Ses ayarı yapmak	792
Ses Girişi İle İlgili API'ler	795
Ses kaydı yapacak donanım var mı?	796
Ses kaydı yapacak donanımın kapasitesi nedir?	796
MIDI Çıkışı İle İlgili API'ler	797
MIDI formatındaki dosyaları çalacak donanım var mı?	797
MIDI çıkış kapasitesi nedir?	797
MIDI çıkış ses seviyesini öğrenmek ve ayarlamak	798
MIDI giriş donanımı var mı?	799
Ses kartının sunduğu diğer sunduğu fonksiyonlarla ilgili API'ler	799

Yardimci Fonksiyonlarin Sayisi?	799
Desteklenen Yardimci Fonksiyonlarin sitesi nedir?	799
Yardimci birimlerin ses seviyesini öğrenmek ve ayarlamak	800
Resim Isleme	801
StretchBlit	805
Ekran Yakalama	808
Nesnelerin hDC numarasini öğrenmek	808
Mouse ile seçmek	810
Program Dosyalarindan ICON alma	812
DIGER API'LER	814
Listelere Yatay Kaydırma Çubuğu	814
Listede Arama	815
Kendiliginden Açılan ComboBox	816
Fare Kapani	816
Fareyi Oynatma	817
Yazici Hakkinda Daha Çok Bilgi	817
INI Dosyasindan okuma	820
Registry işlemleri	820
Programinizi Task Listesinden Gizleme	824
Egik Yazma	824
Bekletme	826
Duvar Kagidini Degistirme	827
Windowsun çalışma süresi	827
Belgeler Menüsüne Doküman Ekleme	828
Dizin Seçme Penceresi	828
Bilgisayar Adini Öğrenme	830
System Tray Uygulamalari	831
17-NEDEN IIS?	840
KURULUM	841
Common Files	842
Documentation	842
File Transfer Protocol (FTP) Server	842
Frontpage 2000 Server Extensions	842
Internet Information Services Snap-In	842
Internet Services Manager (HTML)	842
NNTP Service	842
SMTP Service	842
Visual InterDev RAD Remote Deployment Support	843
World Wide Web Server	843
IIS'E ERISIM	843
IIS'DE GENEL AYARLAR	846
Master Properties	847
Enable Banwidth Throttling	847
Computer MIME Map	847
Server Extensions	848

General	848
Options	849
Permissions	849
IIS'TE YERALAN SERVİSLER	849
WEB SİTESİ OLUSTURMA	852
WEB SİTESİNİN AYARLARI	855
WEB SİTESİNİN AYARLARI –II	869
KAPSAMLI ÖRNEK VISUAL BASIC PROGRAMLARI	879

VISUAL BASIC

Visual Basic 10 yıllık bir geçmişi olan görsel bir dildir. Fakat görsel dil olmadan önce sadece BASIC'in olduğu dönemler 1950'li yıllara kadar dayanır. BASIC kelimesinin açılımı "Beginners All-purpose Symbolic Instruction Code" (Yeni başlayanlar için çok amaçlı sembolik talimat kodu) kelimeleridir. Basic kodu yazmak için GWBASIC ve QBASIC editörleri kullanılarak yazılırdı ve Dos tabanlı idi ve görsel dillerdeki kadar program yazmak kolay değildi. Günümüzde Basic ile program yazmak artık rafa kaldırıldı diyebiliriz. Yıl 1991'de Basic dili güzel bir hamle yaparak Visual BASIC'in 1.0 sürümü çıkarıldı. Basic dili diğer programlama dillerine oranla pek önemsenmediği için VB 1.0 sürümünde pek önemsenmedi. Bundan iki yıl sonra 1993 yılında Visual Basic 3.0 sürümü bir çok gelişme göstererek piyasaya çıktı. Tekrar bundan iki yıl sonra gelişimini sürdürerek 4.0 sürümü 32bit uygulama desteği ile piyasaya sunuldu.

1996 yılında Visual Basic 5.0 sürümü ile karsımıza çıktı. Ve en son olarak Visual BASIC'in en güncel sürümü bir çok sürümü ile karsımızda. (Visual BASIC'e kısaca VB denmektedir.) VB artık Windows tabanlı uygulamalar için önde gelen geliştirme araçlarından biri olarak piyasadadır. Visual'in kelime manası görsel anlamındadır. VB Görsel olusu, programcının bu dili öğrenme sürecini kısaltmakta ve dilin kullanımı kolaylaştırmaktadır. Aynı zamanda Windows tabanlı olması kullanıcıda bir göz asinalığını sağlar. VB'nin görsel olması kadar önemli olan Olay güdümlü olmasıdır. Örneğin; kullanıcı penceredeki bir komut düğmesi üzerinde farenin düğmesine tıklandığında bir olay meydana gelir. Bu olay VB de Click() olay yordamıyla islenir. Visual Basic ortamının önemli bileşenleri şunlardır;

1-VISUAL BASIC'IN ÖZELLİKLERİ

Özellikle Windows ortamında programlama geliştirme aracı olarak kullanılan Visual Basic diğer klasik programlama dillerine göre farklı özelliklere sahiptir. Klasik programlama dilleri ise genellikle karakter tabanlı işletim sistemlerinde kullanılan özellikler klavye ağırlıklı kullanıcı sistemine sahiptir.

Text-mod ortamda

Kullanıcılar seçim için klavye kullanırlar.
Kullanıcıların programın kontrolünde fazla bir etkisi olmaz.
Her programın kullanıcı arabirimi birbirinden farklıdır.
Çoklu programlama genellikle olmaz.
Ekrandan program çıktısı almak yapılamaz.

Windows ortamında

Kullanıcılar seçim işlemlerinde genellikle fare kullanırlar.
Menüler, düğmeler, iletişim kutuları ile programlarda ortak arabirimlerin yapılmasını sağlar.
Kullanıcılar programın kontrolünde oldukça etkindirler.
Aynı anda birçok programın çalıştırılması sağlanır.
Ekrandan çıktılar alınır.

NESNEYE DAYALI PROGRAMLAMA

VB nesneye dayalı (Object-Oriented Programming, OOP) bir programlama dilidir. Çok sayıda nesne hazır olarak VB içinde bulunur programcılar bu nesneleri istedikleri gibi kullanabilirler. OOP üç prensibe sahiptir:

- Encapsulation:** Nesne hakkındaki bilgiler ve işlemler anlamına gelir.
- Inheritance:** Bir nesnenin başka bir nesne üzerine kurulmasıdır.
- Polymorphism:** Belli bir işlemin birçok nesne tarafından kullanılmasıdır.

OLAY -TEMELİ PROGRAMLAMA

Olay-temelli programlama ,klasik programlamanın aksine kullanıcıların işlemlerine göre programın yanıt vermesi temeline kurulu bir programlama sistemidir. Klasik programlama dillerindeki deyimlerin yanı sıra Visual Basic'te nesneleri işlemek için metodlar ve özellikler kullanılır. VB de her nesnenin kendi özellikleri, metodlar ve olayları vardır. Programlama hemen hemen bunların kullanılmasıyla oluşur.

Özellikler (properties) bir nesnenin renk,biçim vb. niteliklerin temsil eder.Metodlar (methods) ise nesnenin işlemlerini, olaylar (events) ise nesnenin sahip olduğu temel tepkimeleri gösterir.

TÜMLESİK PROGRAM GELİSTİRME ORTAMI

Visual Basic bir programlama dilinden öte birşey olduğu için “tümlesik uygulama geliştirme ortamı” olarak anılır.Tümlesik ortam sayesinde kapsanan işlemler:

- Ekran tasarımı.
- Kod yazma.
- Hata giderme.
- Program paketleme.
- Veritabanı erişimini sağlama
- Ve diğer olanaklar.

BİR VISUAL BASIC PROGRAMININ KISIMLARI

Bir program belli kısımlardan oluşur:

- Kullanıcı arabirimi.
- Bilgi işleme.
- Bilgi saklama.
- Raporlama işlemleri.

BİLGİSAYAR VE PROGRAMLAMA NEDİR?

Bilgisayar çok basit düşündüğümüzde üç ana görevi yerine getiren bir makinedir. Girilen bilgiyi alır (INPUT), işler (PROCESSING) ve bu işlenmiş veriden bir sonuç (OUTPUT) çıkarır. Bilgisayar, sadece donanım olarak çalışmaz. Çünkü yazılım olmadan, donanım ne yapacağını bilemez. Bilgisayar donanımına ne yapacağını söyleyecek bir komutlar dizisi gerekir. Yapacağı görevleri, ona anlatan komutlara program diyebiliriz. Yani donanım “sen bunu yap, sonra bulduğun sonucu söyle araya ekle” gibisinden işler yaptırmak programın veya bir başka deyişle yazılımın işidir. Bir programcı olarak bundan fazlasını bilmek elbette ki avantajdır. Ama bilgisayarın bütün özelliklerini bilmeniz gerekmez. Yani yazacağınız bir program için o bilgisayarın özelliklerini bilmeseniz de olur.

Bilgisayarın anladığı tek dil, Makine Dilidir. Bu 16’lık (Hexadecimal) sistemden oluşan bir programlama tipidir. Makine dilini anlamak çok zordur ve bu dili kullanmak için o bilgisayarın donanım özelliklerini mutlaka bilmeniz gerekir. C de ekrana yazı yazmanızı sağlayan “printf();” gibi çok basit bir fonksiyon,

makine dilinde 1A BB OD BC D5 FF C2 F7... gibi çok daha karmaşık ve hiçbir anlam ifade etmeyen bir hâle dönüşür. Makine dili programlama dilleri arasında en alt seviyedir.

PROGRAM NEDİR?

Program, günlük hayatta bir sorunu bilgisayar ile çözmek, rutin işlemleri kolaylaştırmak için yazılan yazılımlardır. Bir program bilgisayar üzerinde çalışır ve insanların günlük hayatlarını kolaylaştırır.

Kişinin program yazması için öncelikle Genel Programlama Bilgisine sahip olması gerekir. Pesinden bir Programlama Dili bilmek gereklidir. Burada önemli olan programlama bilgisidir. Bu konuda kendinizi iyi hissedebiliyorsanız herhangi bir programlama dili ile programlarınızı yazabilirsiniz. Dil tercihi yazılacak programa, soruna ve platforma uygun olarak yapılabilir.

PROGRAMLAMA DILI NEDİR?

Programlama Dili bilgisayarda çözülecek bir sorun için çözümün bilgisayara adım adım yazılmasını sağlayan formal kuralları olan ve bu kurallara siki sikiya bağımlılığı gerektiren bir tanımlar kümesidir.

Belki daha kısa bir tanımi ile sizinle bilgisayar arasında bir tercümandır demek doğru olur.

Bir sorun çözüleceği zaman öncelikle iyice anlaşılmış olmalıdır. Sonra bu sorunu çözebilecek bir çözüm zihinsel olarak hazırlanır. Bu çözüm bilgisayara uygun bir çözüm olmalıdır. Söyleki her çözüm bilgisayarda uygulanamaz. Çünkü her çözümün takip ettiği yol yeteri kadar basit olmayabilir. Üretilen çözüm son derece basit adımlarla anlatılabilir.

Bu adımlarla çözümün anlatılmasına Algoritma denir Bu adımlar alt alta yazılmak suretiyle oluşturulan çözüm bilgisayar için uygundur. Ancak ihtiyaç var ise bu adımlar Akis Çizgesine çevrilebilir. Algoritma doğal bir dille yazılır ve siki sikiya kuralları bulunmaz. Anlaşılmasının kolay olması yeterlidir. Akis Çizgesinde belirlenmiş semboller yer alır ve bu semboller tüm dünyada standart'tir. Kismen formal olan bu çizge sorunun çözümünü daha evrensel bir dille ifade eder.

Son adım olarak sıra Akis Çizgesi veya Algoritma ile elde edilen çözümün bir Programlama dili ile Bilgisayar ortamına aktarılmasına gelmiştir. Programlama dili son derece standart tanımlar içerir ve bir programı yazarken bu tanımlardan bir an için bile uzaklaşamaz. O nedenle bir program parçasından baskalarının baska şeyler anlaması mümkün değildir. Yazılan bu programlar bir derleyici vasıtası ile Makine diline çevrilir varsa hataların bulunmasını sağlar ve kullanıcı bu hataları düzeltir.

PROGRAMLAMANNIN TARİHİ

Programlamanın tarihi oldukça eskidir.1940-1950 yılları arasında programcılar oldukça fazla kodlanmış makine komutlarıyla programlarını yazıyorlardı, o günlerde programcılık oldukça zordu.

İlk programlar makine dili ile hazırlanıyorlardı.Makine dili belli bir bilgisayar sisteminde kullanılan bir programlama dilidir.Makine dili programları özellikle anlaşılması zor olan ve tamamıyla donanımdayalı olarak geliştirilen programlardır.

Makine dillerine yakın ancak yine belli bir mikroislemci için geliştirilmiş dillerden birisi de ASSEMBLY programlama dilidir.Assembly programlama dilinde makine komutları yerine mikroislemcinin anlayacağı belli bir işlemi ifade eden assembly kodları kullanılır.

Bu süreç içinde belli amaçlar ve alanlar için birçok üst düzey programlama dili geliştirildi: ADA,BASIC,FORTRAN,COBOL,PASCAL,PL/I,C,C++... Üst düzey programlama dilleriyle program yazmak makine diliyle yada Assembly dsiyle program yazmaktan daha kolaydır.Çünkü bu dillerde işlemler belli bir prosedürler ve mantıklar şeklinde ortaya konur ve çözülür.

1980'li yıllar üst düzey programlama dillerinin yaygın olarak kullanıldığı yıllar olmuştur.

1990'li yıllar windowslu yıllar oldu.Windows işletim sistemleri kullanıcı etkileşimi ile çalışır.Windows üzerinde çalışan bir program özellikle kullanıcının kolay ve etkin bir biçimde kullanılabilmesi için geliştirilmiştir.

Windows işletim sistemlerinin yaşamımıza girmesiyle programlama görsel ve olay- temelli olarak gelişmiştir.

PROGRAM GELİŞTİRME SÜRECİ

Küçük yada büyük nasıl bir program geliştirirseniz geliştirin, belli bir program-geliştirme adimini takip etmeniz gerekir.Bu adımla şunlardır:

1. Gereksinimlerinizi tanımlayın.
2. Bu gereksinimleri karşılamak için program gereksinimlerini planlayın.
3. Bilgi akışını, hesaplamaları, karar yapılarını ve bilgi yapılarını semalastirir.
4. Programın kullanıcı arabirimini tasarlayın.
5. Programın çıktılarını tasarlayın.
6. Program kodunu yazın.
7. Programı test edin.
8. Programın bakım işlemlerini yapın.

IYI BIR PROGRAMIN NITELIKLERI

Programlar belli bir isin bilgisayar tarafından yerine getirilmesini saglarken; programların belli niteliklere sahip olması gerekir. İyi program niteliklerine sahip olmak, aynı zamanda program geliştirme sürecinin de amaçlarındandır. İyi bir programın nitelikleri:

1. Görsellik
2. Kolaylık
3. Doğruluk
4. Hızlı
5. İyi bir belgeleme
6. Kolayca değiştirilebilme, güncellenebilme
7. Etkin bir kodlama
8. Etkin bir işletim

PROGRAM GELISTIRME ADIMLARI

Programlama işleminin amacı iyi bir program geliştirilmesidir. Bunun yani sıra programlama işlemi yani bir program yazmak için belli takip edilmelidir. Değişik sayıda açıklanabilen bu adımların en yaygın kullanılanları şunlardır:

1.Sistem Analizi: Bir karmaşıklığın bileşenlerini , amaçlarını, önceliklerini tanımlamak amacıyla yapılan çalışmalardır.

2.Sistem Tasarımı: Asamasında sistemin bilgi akışı, girdiler, çıktılar sematik olarak ortaya koyar.

3.Algoritmalar: Bu işi yaparken atacağımız adımları açıklamaktır. Program tasarımı; yapılacak işlemleri açıklamak ve programcıya yol göstermek için kullanılan bir diğer yöntem de akış semaları hazırlamaktır.

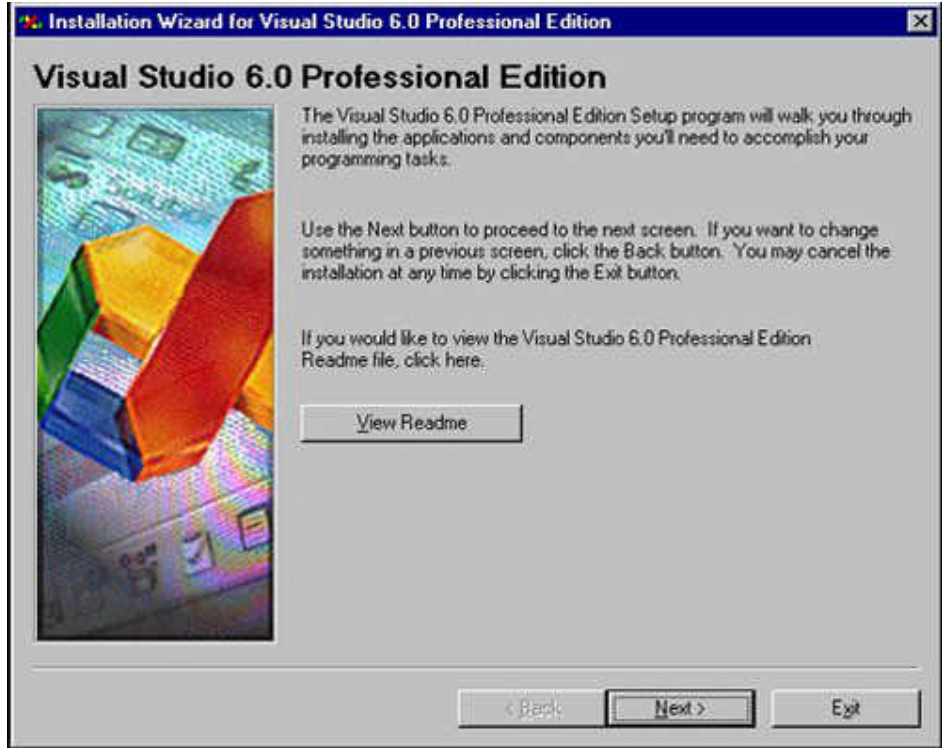
4.Karar Tabloları: Karmaşık işlemlerin açıklanması için bir kılavuz olacak biçimde hazırlanırlar. Karar tablolarında durumlar ve işlemler yer alır.

5.Verit Akış Semaları: Verit işlemlerinde yaygın olarak kullanılır. Verit akış semaları verinin akışını ve dönüşümünü girdiden-çıkıya doğru gösterir. Diğer bir deyişle bilgi hareketini gösteren bir çizelgedir.

6:Kodlama: Program; programlama dili komutlarıyla yazılır. Kodlama belli kurallar çerçevesinde yapılan bir işlemdir.

2-VB 6.0 KURULUMU

Visual Basic Microsoft'un Visual Studio 6.0 program paketini içinde gelir ve buradan kurulur. İlk önce Visual Studio 6.0 Cd 'sini CD-Rom 'a takalım. CD'yi taktikten sonra kurulum programı otomatik olarak başlayacaktır. Karsımıza ilk önce bir hoş geldin penceresi çıkacaktır.

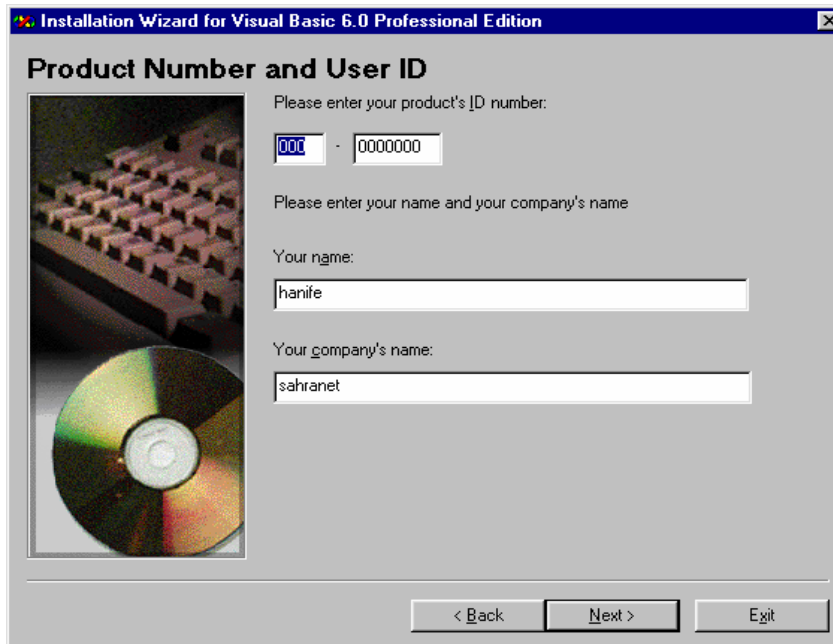


Next'i tıklayarak bir sonraki aşamaya geçelim. Burada bize her Microsoft ürününde olduğu gibi bir anlaşma metni sunulacak bunu 'I accept the agreement' seçeneğini işaretleyerek Next düğmesine basarak ilerliyoruz.

Microsoft Visual Basic 6.0



Daha sonra bizden kurulumun devam etmesi için gerekli olan kurulum sifresini istiyor.

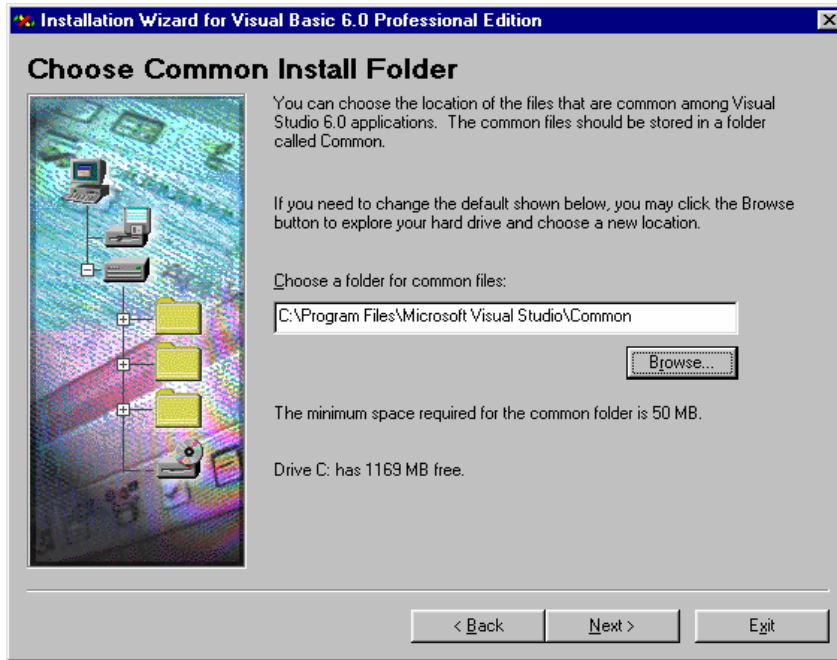


Gerekli kisimlari doldurup Next ' e tiklayalim. Karsimiza kurulum tipini seçmemizi isteyen bir ekran çıkar .Burada bize uygun olan kurulum tipini seçtikten sonra Next 'e tiklayalim.

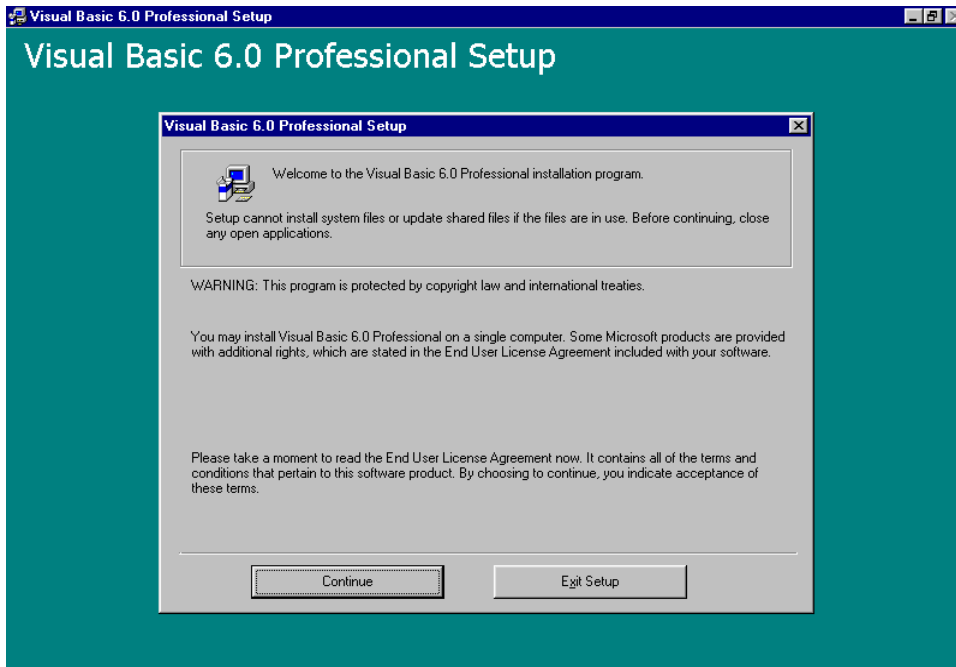


Sonraki ekranda bize programın yüklenecegi yolu belirtmemizi istiyor. Uygun olan bir yol belirtip Next'i tiklayalim.

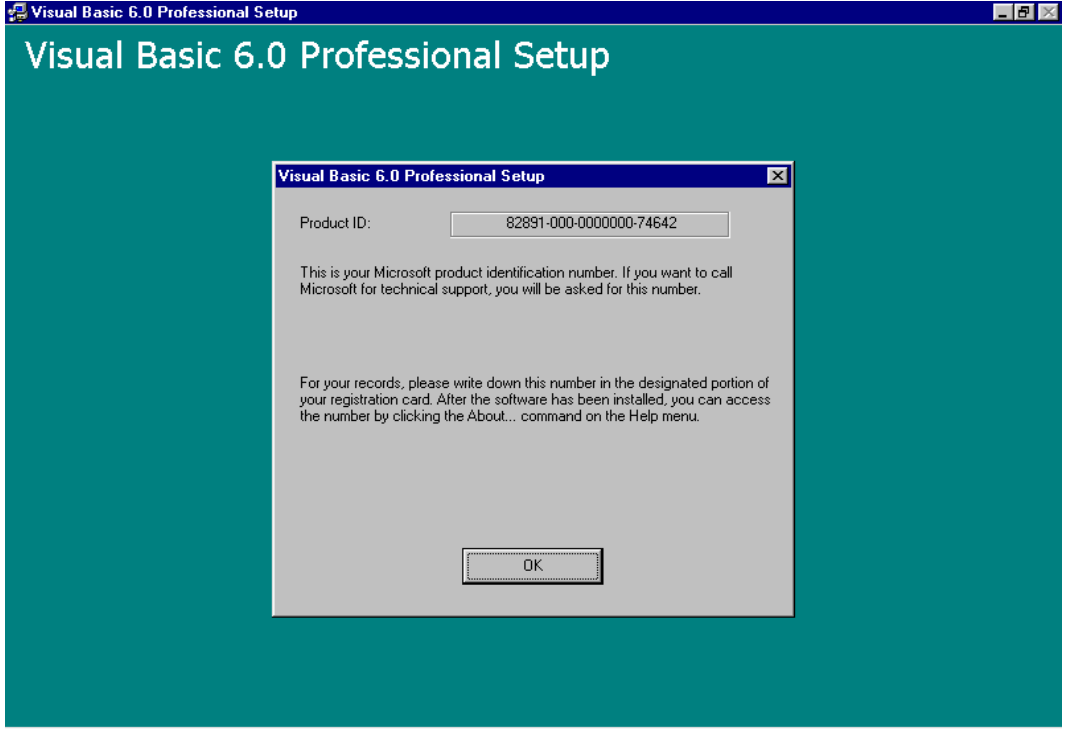
Microsoft Visual Basic 6.0



Nihayet kurulumu başlayabildik. Karsımıza çıkan ekrandan Continue düğmesini tıklayarak ilerliyoruz.



Ardından bize Microdofttan destek alabilmemiz için gerekli olan bir numara (Product ID) verecek bunu bir tarafa kaydedebilirsiniz. Tamam düğmesine basıp ilerleyelim. Daha sonra gelecek ekranda ise bize yüklenecek bileşenleri seçmemizi sağlayacak bir ekran gelecek .Custom düğmesini tıklayarak seçimimizi yapalım.

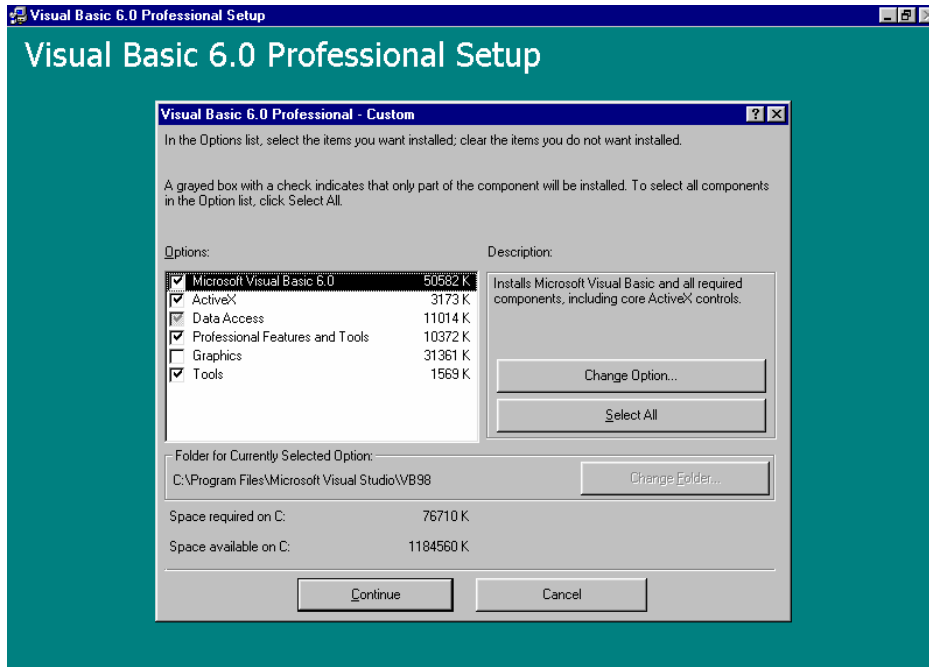


Karsimiza çıkacak ekrandan gerekli bileşenleri seçelim.Bu ekranda Microsft 'un diger yazilim gelistirme araçlarını da görebilirsiniz..

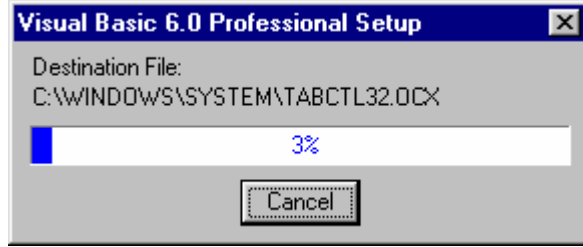
Microsoft Visual Basic 6.0



Tabii ki biz su anda Visual Basic ile ilgilendigimiz için sadece Visual Basic ile alakali olan seyleri seçecegiz.



Continue düğmesini tıklayarak kurulumu devam edelim. Kurulum programı gerekli alan için sabit diskimizi tarayacak ve gerekli alan varsa kurulum başlayacaktır.



Bu asama da bittikten sonra program bizden Windows'u restart etmemizi isteyecek. Programın bir sonraki açılışında düzgün çalışması için Windows'u yeniden başlatalım.

Konfigürasyonlari

Visual Basic 6.0 'in kurulacağı bir makinede olması gereken minimum konfigürasyona gelince;

İşletim sistemi: Win 95, Win 98, Win NT 4.0, Win 2000

İşlemci: 80486 veya daha üstü bir microişlemci

Harddisk Alanı: Minimum 50 MB

Ekran kartı: VGA yada daha yüksek çözünürlüğü destekleyen bir ekran kartı

RAM: 16 MB RAM

SÜRÜMLER

Visual Basic 1.0 Runtime Files Sürüm:

Windows 95/98/NT/2000; 156 KB; Freeware

Visual Basic 2.0 Runtime Files Sürüm:

Windows 95/98/NT/2000; 199 KB; Freeware

Visual Basic 3.0 Runtime Files Sürüm:

Windows 95/98/NT/2000; 285 KB; Freeware

Visual Basic 4 (16 bit) Runtime Module Sürüm:

Windows 95/98/NT/2000; 1.41 MB; Freeware

Visual Basic 4 (32 bit) Runtime Files Sürüm:

Windows 95/98/NT; 1.3 MB; Freeware

Visual Basic 5.0 Runtime Files Sürüm:

Microsoft Visual Basic 6.0

Windows 95/98/NT/2000; 1.24 MB; Freeware

Visual Basic 6.0 Runtime Files Sürüm:

Windows 95/98/NT; 1 MB; Freeware

.NET nedir?

.NET, Microsoft'un pazarladığı gelecek kusak uygulama geliştirme araçları sürümünün ötesinde bir anlama sahiptir. İnternet'i işletim sistemi haline getiren .NET, hem işletim sistemi, hem de İnternet düşüncesinin kapsamını genişletmektedir. Uygulamalarınızı güncelleyerek ve .NET üzerinde yazılım geliştirme çalışması yaparak, yepyeni performans olanaklarından ve gelişmiş özelliklerden yararlanabilecek, bu sayede uygulama geliştirme süresinden tasarrufu sağlayabilecek, XML Web Servisleri gibi yeni uygulama sınıfları oluşturabilecek, Web üzerinde dosya saklama ve kullanıcı tercihi yönetimi gibi yeni yapıtasi servislerinden istifade edebileceksiniz. Visual Studio .NET, Microsoft'un ve dünyanın her yerindeki uygulama geliştiricilerin bu yeni hizmet sınıflarını oluşturabilmesini sağlayan yeni kusak araçlardan oluşur.

Visual Basic .NET

Visual Basic .NET, Visual Basic'in kapsamlı yeniden tasarım işleminden geçirilmiş halidir. Visual Basic .NET'i daha sağlam bir yapılandırılmış programlama dili haline getiren pek çok yeni özellik eklenmiştir. En büyük değişiklik, Visual Basic .NET'in artık yönetilen bir dil olmasıdır. Visual Basic .NET artık kendi yerel derleyicisine sahip değildir, ama önceki bölümde açıklanan common language runtime ögesine derlenmektedir. Visual Basic .NET'in, runtime ile çalışabilmesi ve yeni programlama modelini kullanabilmesi için kapsamlı değişiklikler yapılmıştır. Visual Basic .NET, daha yüksek nesne yönelimi düzeyine sahiptir ve önceki sürümlere oranla daha yüksek type safety düzeyi sunmaktadır. Bu değişiklikler sayesinde, Visual Basic 6.0 projelerini .NET Framework sistemine tasirken kodları yeniden yazmak yerine, basit bir bağlantı noktası kullanmak yeterli olacaktır. Yeni özelliklerden bazıları aşağıda sunulmaktadır.

Common Language Runtime Destegi

Visual Basic .NET common language runtime yaklaşımını temel almaktadır; COM-tabanlı bir Visual Basic .NET yoktur. Visual Basic 6.0 uygulamalarınızı Visual Basic.NET ortamına tasımak için küçük bir güncelleme çalışması yapabilir ve runtime kazançlarından yararlanabilirsiniz.

Nesne Yönelimli Tasarım

- Dil açısından en çok istenen özellik eski bileşenlerin kullanılabilmesidir. .NET sayesinde, artık Visual Basic programcileri eski uygulamaları

kullanabilecek ve yeni Inherits anahtar sözcüğü ile mevcut sınıfları alabilecek ve Overrides ile taban sınıfı işlevini silecektir. Eski öğeleri kullanma özelliği, yönetilen bir dille oluşturulan tüm sınıflarda çalışacaktır.

- Visual Basic .NET'e eklenen bir başka özellik de işlev asiri yüklemesidir. Uygulama geliştiriciler artık aynı ada sahip olan, ama farklı argüman türleri ve dönüş türleri içeren işlevleri oluşturabileceklerdir.
- Visual Basic .NET içindeki oluşturucuları kullanan uygulama geliştiriciler, bir sınıfın yeni örneklerini oluştururken, argümanları sınıfa eşzamanlı olarak iletebilecektir.

Bos Düğüm

Serbest düğümlendirme sayesinde, uygulama geliştiriciler zaman uyumsuz yürütme özelliğini kullanarak, daha ölçeklenebilir ve daha güvenli uygulamalar geliştirecek. Yürütülmesi uzun süren veya harici kaynaklara gerek duyan işlev çağrıları, artık işlem için ikincil düğüm oluşturularak, uygulamanın geri kalan kısmının zaman uyumsuz olarak çalışmaya devam etmesini sağlayacaktır.

Diğer Yeni Özellikler

- Visual Basic .NET içindeki kesin tür denetimi, Visual Basic 6.0 içindeki gizli tür zorlamayı sıkılaştırmaktadır. Visual Basic 6.0'da, hemen hemen her tür başka bir türe gizli olarak çevrilebilmekte, ama tür sınırları aşıldığında çalışma zamanı hatası oluşturmaktadır. Visual Basic .NET, artık çalışma zamanında hataya neden olabilen tüm çevrimler için derleme zamanı hataları oluşturabilmektedir.
- Yapılandırılmış istisna işleme özelliği, Visual Basic 6.0'daki On Error GoTo veya Resume Next hata işleme özelliğinin yerini almıştır. Yapılandırılmış istisna işleme sayesinde uygulama geliştiriciler, Try...Catch deyimlerini kullanarak normal koşullarda ve istisna koşullarında çalışan kodları yazabilmektedir.
- Windows Forms, Visual Basic .NET'te uygulama geliştirme için yeni teknolojidir ve diller arasında uyumludur.

Microsoft® .NET'in Avantajları

Gelecek Kusak İş Entegrasyonu

Önemli bilgileri ancak çeşitli sistemlerde izole bir şekilde saklı tutabildiğiniz günler sona erdi. Bir şirketle veya iş ortakları ile kurulan iş entegrasyonu, değerli olan zamanınızdan ve kaynaklarınızdan tasarruf etmenizi sağlar. Bu durum masraflarını dış kaynaklardan sağlayan bir

bisikletçi dükkani kadar bir tedarik zincirinde yüzlerce iş ortagini entegre eden büyük bir araba üreticisi firma için de geçerlidir.

İşletmelere Yönelik Kazançlar

Microsoft .NET, yazılım uygulamalarının bir arada daha kolay bir şekilde çalışmasını sağlamak amacıyla Internet'i kullanarak bir yandan şirket içinde ve şirketler arasında daha kolay bir entegrasyon sağlarken diğer yandan tüketicilerle daha iyi bağlantılar oluşturmak yönünde fırsatlar yaratıyor. .NET platformunun araçları ile şirketler, iş uygulamalarını geliştirerek ve koruyarak aynı zamanda da çalışanlarının her yerden ve her türlü akıllı cihazdan en önemli bilgilerle iç içe olmalarını sağlayarak zaman ve maliyet açısından gelişmeler kaydedebiliyorlar.

Microsoft .Net ile İlgili Standartlar

Web sayesinde öğrenilen en önemli derslerden biri, veri ve etkileşimleri tanımlamaya yönelik oluşturulan standartların (HTML gibi) çözümleri özel teknolojilere oranla çok daha geniş kitlelere ulaştırabildiğidir. .NET platformu, endüstri standartlarının teknoloji temellerinde oluşturulmuştur.

.NET ile ilgili en önemli standartlar XML, SOAP, UDDI, WSDL, C# ve CLI olarak sıralanabilir; her biri burada kısaca açıklanacaktır.

Microsoft .Net Framework

.NET Framework, Microsoft .NET platformunun geliştirilmesine yönelik temeli oluşturan bir üründür. .NET Framework ve cihaz odaklı .NET Compact Framework, XML'e yönelik kapsamlı bir destek sağlayarak XML Web Servisleri ile ilgili yönetilebilir, güvenli bir çalışma ortamı sunar. .NET Framework'deki en önemli teknolojiler Common Language Runtime, sınıf kitaplıkları ve ASP .NET'dir.

XML Web Servisleri

XML Her ne kadar kolay olsa da, yazılımı oluşturma ve kullanma şeklimizi tam anlamıyla değiştiriyor. Web, kullanıcıların uygulamalarla iletişim kurma şekillerinde bir devrim yaratırken XML de uygulamaların diğer uygulamalarla nasıl iletişim kurduğu yönünde, daha da geniş bir bakışla açıklanacak olursa verinin kolaylıkla uyarlanabildiği veya dönüştürüldüğü evrensel bir veri biçimi sağlanarak bilgisayarların diğer bilgisayarlarla nasıl iletişim kurduğu yönünde bir devrim yaratıyor. SOAP ve UDDI'yi kapsayan XML tabanlı standartlar, XML Web servisleri olarak bilinen uygulamalar arası iletişime yönelik açık bir metodolojiyi içermektedir.

Hailstorm

Microsoft, kod adi "HailStorm" olan ve bir çok uygulamanin ihtiyaç duyacağı kullanıcı odaklı bir XML Web servisleri seti geliştiriyor. "HailStorm" hizmetleri belirli bir aygıt, uygulama, hizmet veya agdan ziyade insanlara uygun olarak geliştirilmektedir. Kullanıcıların verileri ve bilgileri üzerinde denetim kurmalarını sağlar ve size herhangi bir cihazdan istediğiniz zaman bu verilere erişebilme imkanını tanır. Kullanıcının kendi bilgilerine kimlerin erişebildiğini denetlemesini sağlayarak kişisel bilgileri korur ve böylece kullanım kolaylığına ve kişiselleştirme özelliğine yeni bir boyut kazandırır. "HailStorm" hizmetleri, Microsoft'un oluşturduğu ilk XML Web servsidir.

Akıllı Cihazlar

Akıllı cihazlar, PC'leri, dizüstü bilgisayarlarını, iş istasyonlarını, telefonları, el bilgisayarlarını, Tablet PC'leri, oyun konsollarını ve .NET evreninde çalışabilmeleri için oluşturulan yazılımı kullanan diğer cihazları kapsamaktadır.

Bu yazılımın kullanımıyla akıllı cihazlar, kullanıcılara bir yandan güçlü ve yoğun bir kullanım deneyimi sunarken diğer yandan bilgileri üzerinde daha fazla denetim kurmalarını sağlayarak Internet ve yerel bilgi işlem gücünün üstün özelliklerini etkin hale getirirler.

İstemci Yazılımları

Son kullanıcılar bilgisayarları istemci yazılımları aracılığı ile kullanmaktadırlar. Microsoft, son kullanıcılara zengin içerikli bir kullanım sağlayan istemci yazılımlarını oluşturmak konusunda engin deneyimlere sahip. İstemci yazılımı Microsoft .NET'in başarısında oldukça yönlendirici bir rol oynuyor; geniş yelpazedeki istemcileri desteklemek üzere altyapı oluşturmalarının yanı sıra Microsoft aynı zamanda da PC'leri ve akıllı cihazları güçlendirmek için yeni nesil yazılım istemcileri geliştiriyor.

Visual Basic 6.0 Uygulamalarını Visual Basic .NET'e Güncelleme

Microsoft Visual Basic uygulama geliştirme sisteminin sonraki sürümü olan Microsoft Visual Basic® .NET, güçlü Microsoft Windows® tabanlı uygulamaları, ölçeklenebilir veri erişimi bileşenleri, düşük kapasiteli Web-tabanlı uygulamaları, yüksek korunma düzeyine sahip uygulama çözümlerini ve XML Web Servislerini geliştirme işlemlerini hızlı bir şekilde yapabilmenizi sağlayacak birincil araç olarak yeniden tasarlanmıştır. Microsoft, Visual Basic 6.0'a bazı yeni özellikler eklemekle yetinmek yerine, Visual Basic .NET'i

Microsoft Visual Basic 6.0

Microsoft .NET Framework üzerinde yeniden olusturdu ve yeni platformdan birinci sinif dil olarak tasarladı. Bu sayede Visual Basic ile uygulama gelistirenler, Visual Basic .NET içindeki gelismis özelliklerden yararlanarak farkli ihtiyaçlara uygun uygulamalari hemen olusturabilecek.

Visual Basic. NET'i kullanan uygulama gelistirciler önceki Visual Basic sürümlerine oranla, çok daha güçlü ve zengin araç kümesine erişim sağlayabiliyor. Güçlü müşteri talebi doğrultusunda geliştirilen Visual Basic .NET'in sundugu kapsamlı yeni özellikler arasında, siki denetim, yapılandırılmış istisna işleme ve dolaysız .NET Framework erişimi bulunuyor. Buna ek olarak, Visual Basic dili kusursuzlaştırılarak eski anahtar sözcükler kaldırıldı, gereksiz fazlalıklar yok edildi ve dil verimliliği artırıldı. Bu özellikleri gerçekleştirmek için dilde bazı söz dizimi değişiklikleri yapıldı.

Visual Basic .NET Upgrade Tool

Dille ilgili bu gelişmiş özelliklerin en verimli şekilde kullanılabilmesini sağlamak için, uygulama gelistircilere Visual Basic 6.0 projelerini Visual Basic .NET'e güncelleyebilecek bir araç da sunuluyor. Visual Studio .NET Professional, Enterprise Developer ve Enterprise Architect sürümlerinde bulunan bu araç, uygulama gelistircileri sürüm yükseltme sürecinin tüm aşamalarında yönlendiriyor ve yeni bir Visual Basic .NET projesi olusturuyor (mevcut Visual Basic 6.0 projeleri değiştirilmiyor).

Visual Basic 6.0 projeleri, Visual Basic .NET içinde açıldığında sürüm yükseltme aracı otomatik olarak çalışıyor. Bu araç dilde aşağıdakileri içeren bir dizi söz dizimi güncellemesi yapıyor:

- **Degiskenleri Nesneye Çevirme.** Önceki Visual Basic sürümleri artık Visual Basic .NET'te desteklenmeyen Variant (Degisken) veri türünü destekliyordu. Variant olarak tanımlanan tüm degiskenler Object (Nesne) türü degiskenlere çevriliyor.
- **Tamsayıları Kisaya, Uzunları Tam Sayıya Çevirme.** Visual Basic 6.0'da, 32-bit tam sayılar için Long (Uzun) veri türü; 16-bit tam sayılar için de Integer (Tam Sayı) veri türü kullanılıyordu. Visual Basic .NET'te 64-bit tam sayılar için Long (Uzun) veri türü; 32-bit tam sayılar için Integer (Tam Sayı) veri türü; ve 16-bit tam sayılar için de Short (Kısa) veri türü kullanılıyor. Upgrade Tool 32-bit ve 16-bit veri türlerini otomatik olarak Integer ve Short değerlerine degistiriyor.
- **Parametresiz Varsayılan Özellikler.** Visual Basic 6.0'da çoğu nesnenin varsayılan özelliği bulunuyordu ve bunlar programlama kısayolu olarak atlanabiliyordu. Upgrade Tool bu parametresiz varsayılan özellikleri Visual Basic .NET'te kullanım için hazırlıyor.

- **Sifirsiz Sinir Dizileri.** Visual Basic 6.0 herhangi bir tam sayinin alt ve üst sinirlarinda dizi kullanimina izin veriyordu. Visual Basic .NET Upgrade Tool dizi sarma sinifini kullanarak, sifirsiz sinir dizilerini otomatik olarak güncellestiriyor.

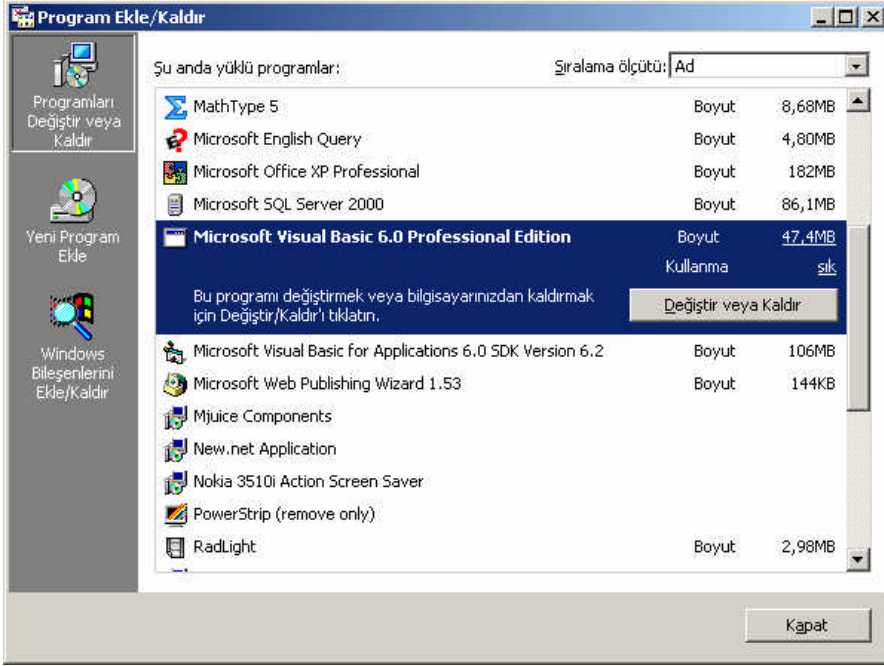
Visual Basic .NET Upgrade Tool, Visual Basic 6.0 formlarini Windows Forms olarak güncellestiriyor. Windows Forms, Visual Studio .NET içinde paylasilan yeni bir form paketidir. Yerel düzeyde erisilebilirlik desteginin yani sıra, yerinde menü düzenleyicisi içermektedir.

Günümüzün Uygulamalarini Güncellemek için Mimarî Rehber

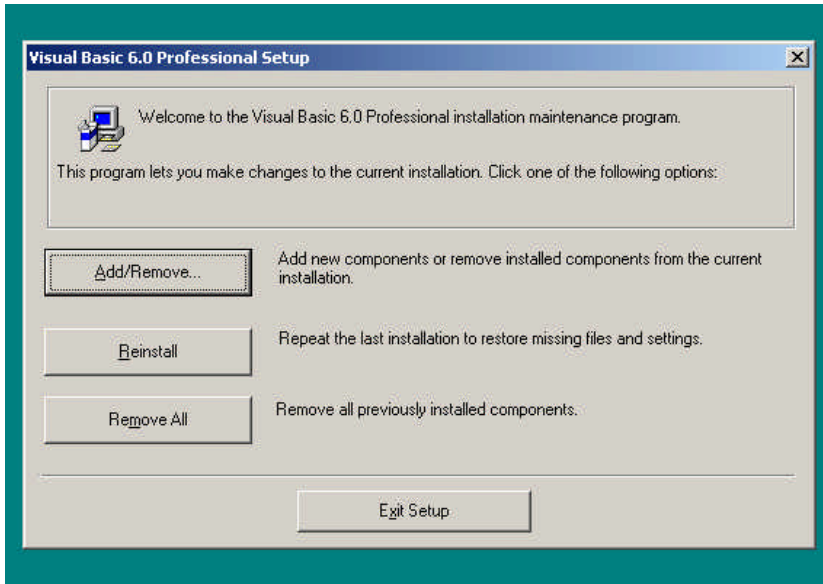
Uygulama gelistiriciler Upgrade Tool kullanimina ek olarak, Visual Basic 6.0 kodlarinin Visual Basic .NET'e mümkün olduğu ölçüde kusursuz olarak güncellemek için gerekli adimlari atabilir. Microsoft Visual Basic 6.0 projelerini güncellerken, gerekli olabilecek elle degisiklik yapma gereksinimini en aza indirmek için, bir dizi mimarî tavsiye sundu. Bu tavsiyeler asagidakileri içermektedir:

- Erken degisken baglama kullanimi.
- Null (Bos) doldurma kullanilmamasi.
- Veri erisimi için Microsoft ActiveX® Data Objects (ADO) kullanimi.
- Tarih saklamak için Double (Çift) veri türünün kullanilmamasi.
- Kullanici tanimli türlerde sabit uzunlukta dizi kullanilmamasi.
- Alt deger yerine sabit kullanilmasi.

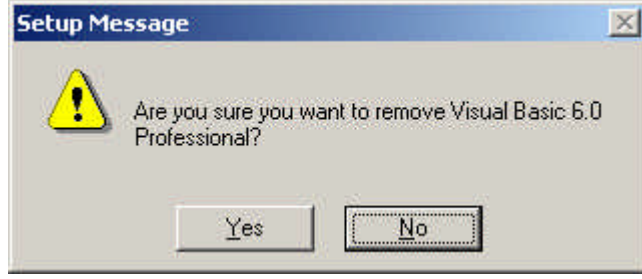
Visual Basic 6.0'in Sistemden Kaldirilmesi



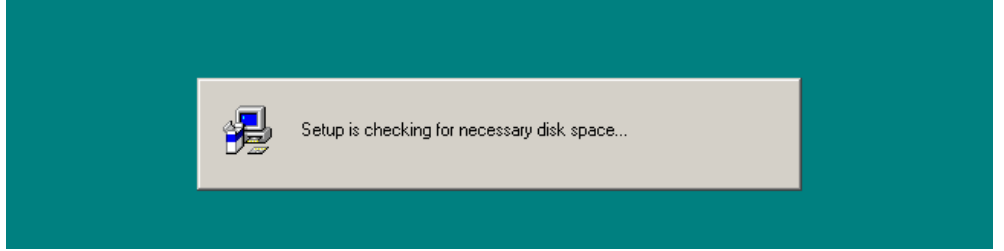
Windows ta control panelden program ekle/kaldır seçeneğini tıklayarak yüklü olan Microsoft Visual Basic 6.0 seçeneğini seçin ve kaldır seçeneğini tıklayın.



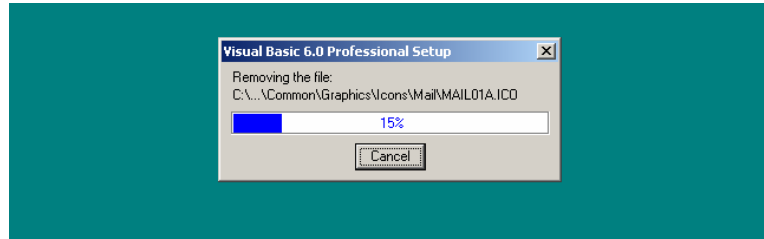
Gelen menüden component eklemek veya kaldırmak için Add/Remove seçenegini, Tekrar yükleme yapmak için Reinstall seçenegini, V Basic i tamamen kaldırmak için ise Remove All seçenegini seçiniz. Biz tamamen kaldırmak için Remove All seçenegini seçiyoruz.



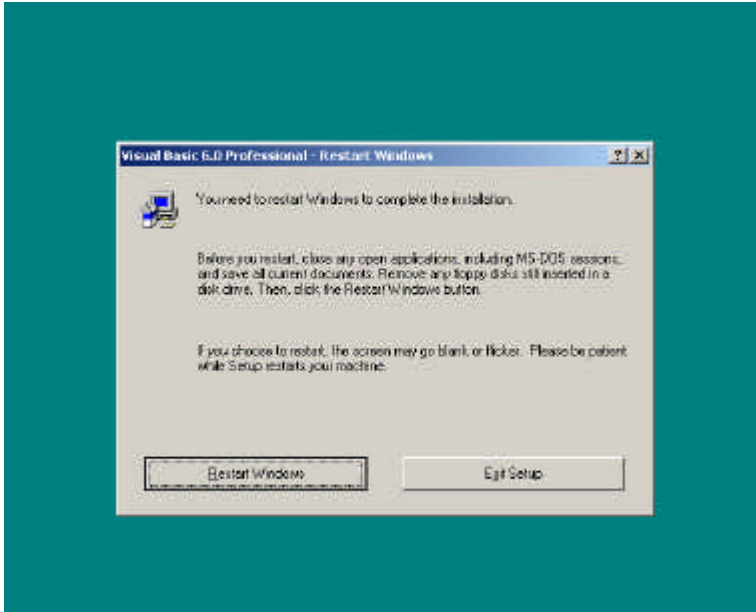
Bize bu işlem için emin olup olmadığımızı soruyor.Yes diyerek geçiyoruz.



Bilgisayarda gerekli bos alan varmi diye kontrol ediyor.



Kaldırma işlemine baslıyor.



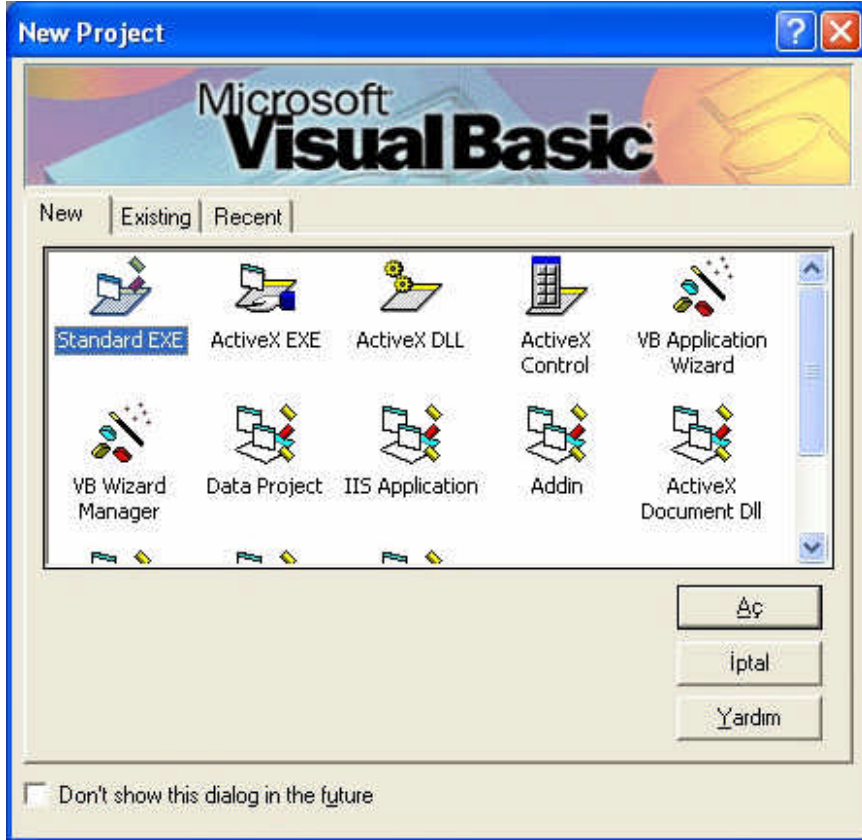
Kaldırma işlemi bittikten sonra bilgisayarınızı yeniden baslatmanız gerekiyor. Hemen yeniden baslatmak için Restart Windows seçeneğini, daha sonra baslatmak için Exit Setup seçeneğini seçiyoruz.

3-VISUAL BASIC'TE ÇALIŞMA

BASLATMA

Visual Basic'i ilk kez başlattığınızda Project Wizard açılır ve **New Project** iletişim kutusunu görürsünüz. Bu pencereden size uygulamanızı geliştirme konusunda bir başlangıç noktası teskil edecek olan çeşitli proje türlerinden birini seçebilirsiniz. Bu pencerenin 3 sekmesi vardır; **New**, **Existing** ve **Recent**.

NEW SEKMESİ :



New sekmesinden bir proje sablonu seçerek Visual Basic'in uygulamanızın temellerini oluşturmasını sağlayabilirsiniz. Bu işlem özellikle Visual Basic'te yeniyseniz bir uygulamayı tasarlamaya ayıracağınız süreden önemli bir tasarruf sağlayabilir. **New** sekmesi size çeşitli proje sablonları sunar;

Microsoft Visual Basic 6.0

- * Standart EXE
- * ActiveX DLL
- * VB Application Wizard
- * Data Project
- * Add-In
- * ActiveX Document EXE
- * VB Enterprise Edition Controls
- * ActiveX EXE
- * ActiveX Control
- * VB Wizard Manager
- * IIS Application
- * ActiveX Document DLL
- * DHTML Application

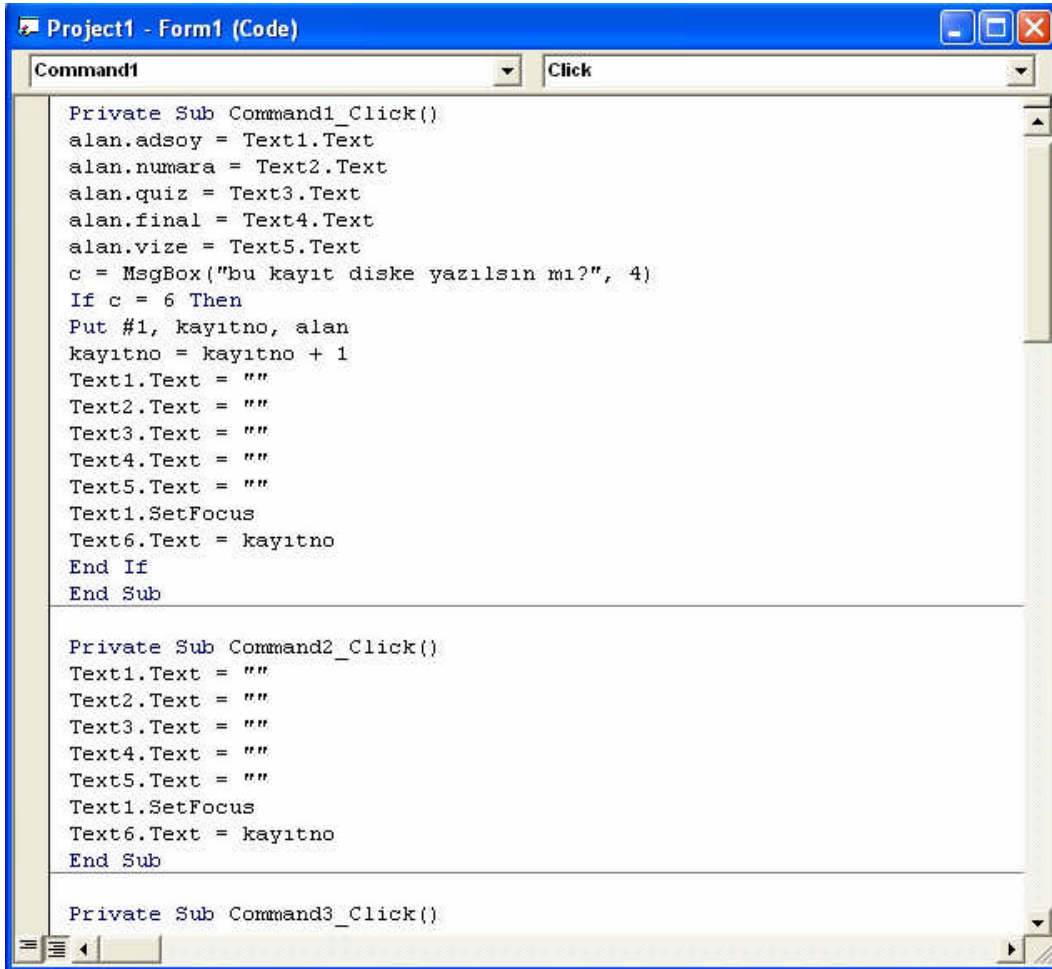
EXISTING SEKMESİ:



Existing sekmesi mevcut bir projeyi seçmemizi sağlar. Bu, Visual Basic ile gelen örnek bir proje yada geçmiste üzerinde çalışmış olduğunuz bir proje olabilir. Visual basic ile daha çok çalıştıkça bu sekmeyi de daha sık isaretleyeceksiniz.

bir form veya **Code** penceresini görürsünüz. Yukarıdaki şekilde formun her iki yanının ortasında siyah renkli küçük noktalar olmasına dikkat edin. Bu kutulara *çapa* (anchor) adı verilir. Bir çapayı fareyle sürükleyerek formun boyutunu değiştirebilirsiniz.

KOD EDITÖRÜ



Bu aslında içinde çok sayıda üretkenlik aracı bulunan, turbo sarjlı bir metin editörüdür. Ona ne isim verirsiniz verin, işinizin büyük kısmını yapacağınız pencere budur. **Kod Penceresi**'ni **form Layout Penceresi**'nde bir form veya denetimi çift tıklayarak açabilirsiniz. Bir formu çift tıklarsanız bu formun bir yordamına götürülürsünüz. Formu **Project Explorer Penceresi**'nde çift tıklayarak açın yada **Project Explorer** içinde **View Code** düğmesini tıklayın. Bir denetimi tıklarsanız bu denetimin bir yordamına

götürülürsünüz. **Kod Penceresi** açıldıktan sonra seçili form üzerindeki bütün nesnelerin bütün yordamlarına gidebilirsiniz.

ARAÇ KUTUSU



Araç Kutusu adından da anlasılacağı gibi uygulama arabirimini oluşturmak için gereksiniminiz olan parçaları içerir. Yandaki şekilde gösterilen araçların hepsi, sol üstteki imleç hariç uygulamanızda ki bir forma yerleştirmek isteyebileceğiniz nesne veya öğelere karşılık gelmektedir. Bu araç veya nesnelere *denetim* adı verilir. Bunların çoğu Visual Basic'in birer parçasıdır ve bunlara *yerlesik* yada *standart* denetimler adı verilir. Bunlara örnek olarak komut düğmesi ve metin kutusu denetimleri verilebilir. Bölüm 3 bu denetimleri daha ayrıntılı olarak ele alacak. Visual Basic kur ayarlarınıza bağlı olarak araç kutusunda daha az veya çok sayıda denetim olabilir.

ARAÇ KUTUSUNUN DÜZENLENMESİ :

Denetimlerinizi düzenlemek için kullanabileceğiniz sekmeler tanımlamanıza izin verir. Özgün denetimlerinizi kategorilere göre düzenlemek isteyebilirsiniz. Örneğin bütün Internet özgün denetimlerini ayrı bir sekmede tutalım. Araç kutunuza yeni bir Internet sekmesi eklemek için aşağıdaki adımları izleyin:

1. Araç kutusunun boş bir yerini sağ tıklayın.
2. Açılan menüden **Add Tab** komutunu verin.
3. Visual Basic sizden yeni bir sekme girmenizi istediğinde Internet yazın.
4. OK düğmesini tıklayın.

5. Artık yeni bir araç kutusu sekmesi oluşturabileceğinize göre istediğiniz bütün denetimleri de bu sekmeye sürükleyebilirsiniz; Örneğin görüntü denetimini az önce

olusturdugunuz Internet sekmesine sürükleyiniz. Bu anda araç çubuğunuzda Internet ile ilgili herhangi bir denetiminiz olmayacak.

6. Microsoft ve diğer üçüncü parti şirketler tarafından oluşturulan özgün denetimleri eklemek için araç kutusunu sağ tıklayarak açılan menüden **Components** yada ana menüden **Project => Components** komutunu verin.
7. Kullanılabilir denetimler listesindeki bu örnekte Microsoft Internet Controls olan eklemek istediğiniz denetimin yanındaki kutuyu işaretleyin.
8. Bu denetimler araç kutusuna eklemek için OK düğmesini tıklayın.

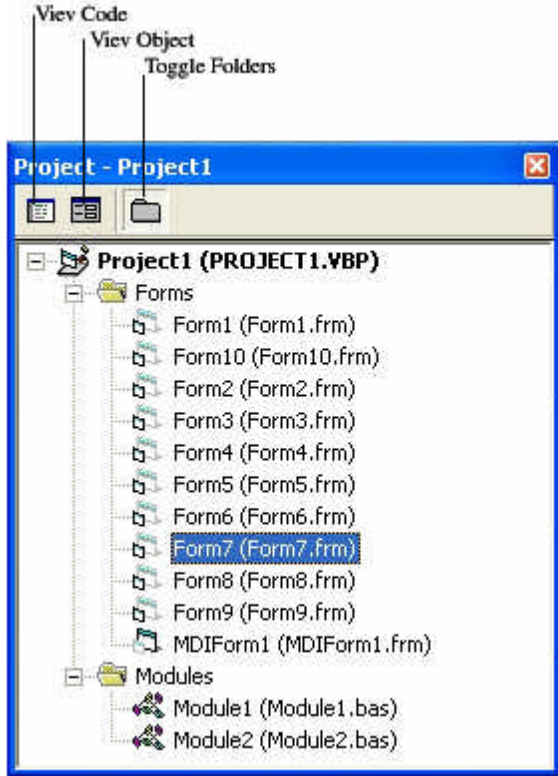
Tanımlayacağınız sekme ve kategorilerin isimleri tamamen kişisel tercihlerinize bağlıdır. Dilediğiniz gibi sekmeler oluşturun ve denetimlerinizi istediğiniz gibi düzenleyin.

BİR ARAÇ KUTUSU DENETİMİNİN ÇIKARTILMASI

Bir denetimi çıkartmak için **Custom Controls** iletişim kutusunda ilgili kutunun isaretini kaldırmanız yeterlidir. Yerlesik denetimleri araç kutusundan çıkartamayacağınızı bilmelisiniz, bu yüzden komut düğmesi gibi denetimler daima mevcut olacaktır. Internet denetimlerini çıkartmak için;

1. Araç kutusunu sağ tıklayın.
2. Açılan menüden **Components** komutunu verin.
3. Geçtiğimiz alıstirmada deentimleri eklediginiz gibi, onlari ilgili denetimin yanındaki onay kutusunun isaretini kaldırarak çıkartirsiniz. Microsoft Internet Controls başlığının yanındaki onay isaretini kaldırın.
4. OK düğmesini tıklayın.

PROJECT PENCERESİ



Visual Basic'de ekranın sağ tarafında, araç çubuğunun hemen altında Project Explorer penceresi yer alır. Yandaki Project Explorer, projenizdeki çeşitli elementlere (formlar, sınıflar ve modüller) hızla erişmenizi sağlar. Project Explorer penceresi, alt klasörleri açıp kapatmanıza olanak vermesi açısından büyük oranda Windows Explorer'a benzer.

Uygulamanızı oluşturan nesnelerin hepsi bir projeye paketlenir. Bunu daha sonra kullanmak, test etmek, hata gidermek veya geliştirmek için kaydederseniz Visual Basic bu projeye varsayılan dosya uzantısı olan .VBP'yi (Visual Basic Project) verir.

Basit bir proje genellikle uygulamanız tarafından kullanılan pencere olan bir form içerir. Project Explorer penceresi formlara ek olarak ayrıca sınıf modüllerini ve sınıfları da listeler.

NOT: Büyük uygulamaların genellikle birkaç form, modül, ve sınıf olur. Bunlarda Project Explorer penceresinde listelenir.

Bir formu görmek için onu Project Explorer içinde seçin ve **View Object** düğmesini tıklayın. Bu formla ilişkili bir kod varsa **View Code** düğmesi tiklanarak kendine ait bir pencerede görülebilir. Project Explorer'ı sağ tıklarsanız size bu pencereye özgü çok sayıda seçenek içeren bir menü

sunulur. Örneğin, bu açılan menüden form ve kod modülleri ekleyebilir, çıkartabilir ve basabilirsiniz. Projenizden bir nesneyi çıkartmak isterseniz bu nesnenin ismini Project Explorer penceresinde sağ tıklayın ve **Remove** komutunu verin. **Remove** komutunun yanında bu nesnenin adı görülür.

ÖZELLİKLER (PROPERTIES) PENCERESİ

(Name)	Form7
Appearance	1 - 3D
AutoRedraw	False
BackColor	&H8000000F&
BorderStyle	2 - Sizable
Caption	TÜM HESAP KAYIT
ClipControls	True
ControlBox	True
DrawMode	13 - Copy Pen
DrawStyle	0 - Solid
DrawWidth	1
Enabled	True
FillColor	&H00000000&
FillStyle	1 - Transparent
Font	MS Sans Serif
FontTransparent	True
ForeColor	&H80000012&
HasDC	True
Height	7350
HelpContextID	0
Icon	(Icon)
KeyPreview	False

(Name)
Returns the name used in code to identify an object.

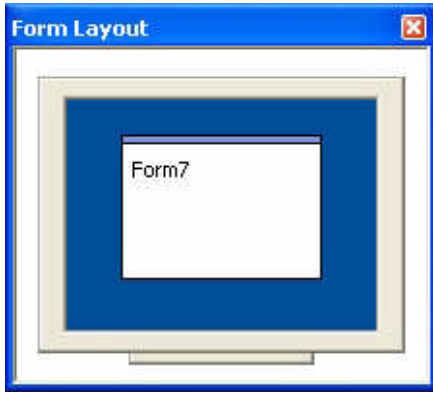
Project penceresinin hemen altına **Properties** penceresi bulunmaktadır. **Properties** penceresi seçili nesnelerin çeşitli karakteristiklerini (veya özelliklerini) sunar. Bu kavramı netleştirmek için bir uygulamadaki bütün formların birer nesne olduklarını düşünün. Bir formun üzerinde görünen bütün denetimler de (örneğin bir komut düğmesi) birer nesnedir. Visual Basic'te ki her nesnenin renk boy gibi karakteristikleri bulunur. Diğer karakteristikler ise sadece nesnenin görünümünü değil, aynı zamanda tutumunu da etkiler. Bir nesnenin bütün bu karakteristiklerine onun özellikleri adı verilir. Bu yüzden bir formun özellikleri vardır, üstelik bir formun üzerine yerleştirilen bütün denetimlerin de özellikleri vardır. Bu denetimlerin tümü **Properties** penceresi'nde görüntülenir.

Properties penceresi'nde bir nesneye ait olan özelliklerin bir listesini görürsünüz. Bu özelliklerden oldukça çok sayıda vardır ve tümünü görmek için listeyi kaydırmanız gerekebilir. Neyse ki

bu özelliklerin çoğu kendi kendini açıklar. (*Caption, Width, Height gibi [Tabi bunlar İngilizce bildiginiz varsayılırsa doğrudur.]*), fakat bazıları da oldukça seyrek kullanılır. Belirli bir özelliğin ne ise yaradığından emin değilseniz onu işaretleyerek **Properties** penceresi'nin alt tarafında küçük bir açıklamasını görebilirsiniz. Özellikleri görmek için listeyi kaydırabilmenize ek olarak uygun sekmeyi tıklayarak onları alfabetik olarak veya kategorilerine göre de sıralayabilirsiniz. Kullanacağınız yöntem şahsi tercihinize bağlıdır.

Bir denetim, örneğin bir komut düğmesi bir formun üzerine yerleştirildiğinde Properties penceresi bu denetim seçildiği zaman onun özelliklerini gösterir. Aralarında temel formun da yer aldığı farklı nesnelerin özelliklerini bu nesneleri sırayla tıklayarak görebilirsiniz. Alternatif olarak **Properties penceresi**'nin üst tarafındaki açılır listeyi istediğiniz denetimi seçerek bunun özelliklerini seçerek bunun özelliklerini kullanmak için kullanabilirsiniz. Çoğu özellik tasarım kipinde ayarlanır, ancak pek çoğu da çalışma kipinde değiştirilebilir.

FORM LAYOUT PENCERESİ



Form Layout Penceresi basit ama faydalı bir araçtır. Bu pencerenin amacı size etkin formun temsili bir görünümünü sunarak neye benzediğini ve çalışma kipinde ekrandaki konumunu göstermektedir.

Bu pencerede iken fare'nin sağ tuşuna basıldığında karşımıza çıkan popup menüde Startup Position seçeneğinin içerisindeki

MANUAL: fare ile istenilen yere formu taşıyabilmek için seçili olmalıdır.



CENTER OWNER: Programa ait açık pencere varsa onun ortasında açar

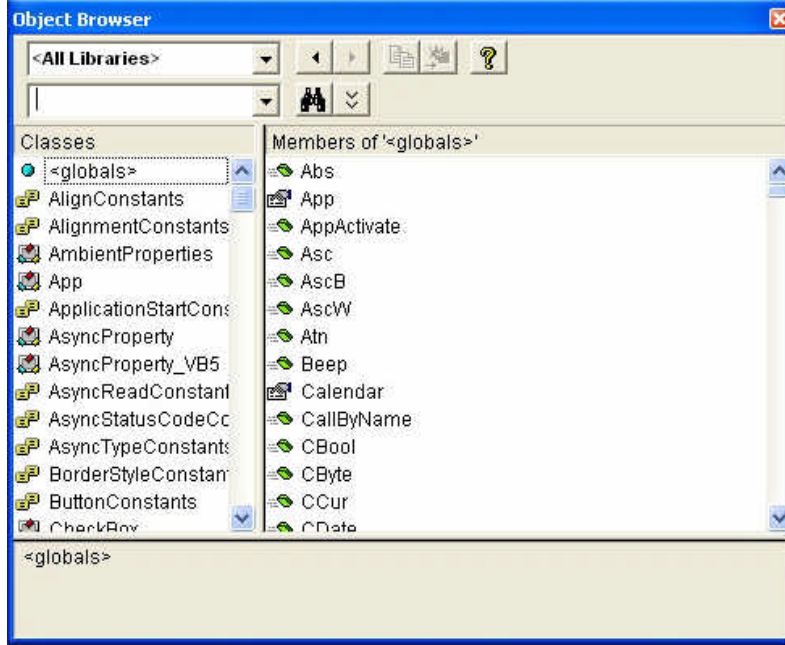
CENTER SCREEN: Ekranın tam ortasında formu odaklar.

WINDOWS DEFAULT: Standart konuma odaklar.

Form Layout Penceresi uygulamanız çalıştığında formun ekranda ne kadar yer kaplayacağını belirleme açısından da faydalıdır. **Form Layout Penceresi**'ni kullanmak için aşağıdakileri yapın.

1. **Form Layout Penceresi**'nde formu tıklayın ve onu bu pencerenin ortasındaki monitör grafiğinin merkezine taşıyın.
2. **Run => Start** komutunu vererek programı çalıştırın.

OBJECT BROWSER



Object Browser sizin kullanimınıza açılan yada gösterilen çeşitli özellik, olay ve yöntemler arasında dolasmanizi saglar. Ona **View** menüsünden **Object Browser** komutunu vererek yada **F2** tusuna basarak erisebilirsiniz.

Bu nesneler nereden gelir ve herhangi bir yerde hangi nesnelerin kullanilabilecegini ve bunlari hangi özellik ve yöntemleri sunduklarini nasil bilebilirsiniz? Bütün nesneler, ister Visual Basic'te yerlesik olsun, ister siz olusturmus olun bir tip kütüphanesine sahiptir. Bir tip kütüphanesi, nesnenin özellik, yöntem ve olaylarinin istediginiz herhangi bir anda basvurabileceginiz bir katalogdur. Aralarinda sizin özgün ActiveX denetim ve (EXE ve DLL formatindaki) ActiveX bileşenlerinizin de yer aldigi bütün nesnelerin **Object Browser** içinde görüntülenebilen bir tip kütüphanesi vardir. Neyse ki bileşenlerinizin üyelerini görüntülemek için özel bir eylem yapmaniz gerekmez. **Object Browser** bunlari tanımlarini çalıştırabilir dosyadan toplar (ancak tanımlarini sunmaniz gerekir).

Kullanilabilir nesnelerin tip kütüphaneleri yukarida görülen **Object Browser** içinde görüntülenir.

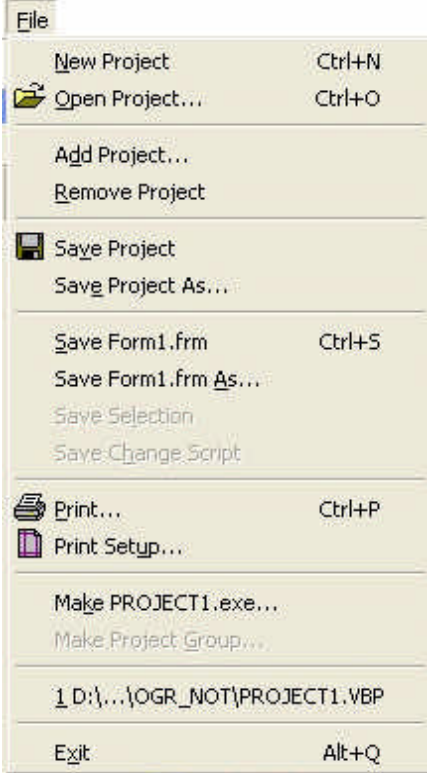
Object Browser yardım dosyalari veya referans malzemesinin bir alternatifi degildir. Ancak deneyimli programcilar bile muhtemelen Database nesnesi gibi karmasik nesnelerin yapılarini hatirlayamaz. **Object Browser**, Database nesnesi ile Veri tabani uygulamalari programlama isini basitleştirir.

Yani kisacasi Visual Basic veya nesneleri kullanima sunan diger uygulamalar içinde bulunan bütün siniflar **Object Browser** içinde açıklanir.

Object Browser bu açıklamaları ilgili EXE veya DLL dosyalarından alan ve bunları görselleştirmesi ve içinde dolması kolay olan bir şekilde sunan bir uygulamadır.

ORTAMIN DÜZENLENMESİ

FILE MENÜSÜ



Yanda açık haldeki resmi bulunan **FILE** menüsünün elemanları ve kullanıldığı yerler şunlardır:

NEW PROJECT: Yeni boş bir proje açar.

ADD PROJECT: Diske kayıtlı olan bir projeyi açmak için kullanılır.

REMOVE PROJECT: Project penceresinde seçili olan form proje, form yada modül siler.

SAVE PROJECT: Üzerinde çalışılan projeyi kaydeder.

SAVE PROJECT AS: Üzerinde çalışılan projeyi farklı bir isimde veya farklı bir yere kaydeder.

SAVE FORM1: Project penceresinde seçili olan nesneyi (form, modül, vb..) kaydeder.

SAVE FORM1 AS: Project penceresinde seçili olan nesneyi (form, modül, vb..) farklı isimde veya farklı yerde kaydeder.

PRINT: Bir proje yada form dizaynını yazdırmak için bu seçeneği kullanırız.

PRINT SETUP: Yazıcı ve yazdırma ayarlarını yapmak için kullanılır.

MAKE PROJECT EXE: Üzerinde çalışılan projeyi EXE uzantılı bir programa çevirir

EXIT: Programdan çıkar

EDIT MENÜSÜ

Altındaki Yandaki açık hali bulunan **EDIT** menüsü üzerinde çalışılan proje form modül vs..'de gerekli düzenlemelerin yapılması için gerekli komutları içermektedir.

UNDO .. : Son yapılan işlemi iptal etmek için kullanılır.



REDO .. : Son olarak yapılan geri alma işlemini iptal eder.

CUT: Seçili olan veri kesilerek panoya atılır.

COPY: Seçili olan verinin bir kopyası panoda oluşturulur.

PASTE: Panoda bulunan veri kursörün bulunduğu mevkiye yapıştırılır.

DELETE: Seçili vaziyette bulunan veri silinir.

SELECT ALL: Sayet form aktif ise tüm elemanlar, Kod penceresi aktif ise tüm yazılan kodlar aktif duruma getirilir.

FIND: Kod penceresi üzerinden istenilen bilgiyi buldurmak için kullanılır.

FIND NEXT: Arama sonrasında istenilen kelimenin sonrasında varsa onu göstermede kullanılır.

REPLACE...: İstenilen bir ifadeyle diğer bir ifadeyi otomatik olarak değiştirmek için kullanılır.

INDENT: Kod penceresindeki ifadeleri 8 karakter (1 tablik) içerden baslatır.

OUTDENT: Kod penceresindeki ifadeleri 8 karakter (1 tablik) dışardan baslatır.

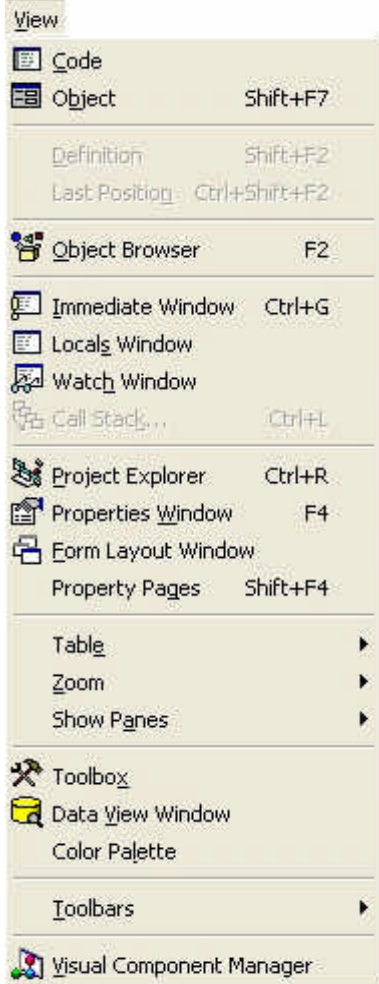
INSERT FILE: Kursörün bulunduğu noktadan itibaren uzantisi TXT, BAS, veya CLS olan bir dosya eklemek için kullanılır.

LIST PROPERTIES/METHODS: Menü seçeneği ile aşağı yukarı açılan özellik listeleme kutusunu görüntüler.

QUICK INFO: Menü seçeneği ile bir ifadenin içerdiği parametreleri görüntüler.

PARAMETER INFO: Menü seçeneği ile yazılan bir fonksiyonun içerdiği parametre tipi öğrenilebilir.

VIEW MENÜSÜ



Yanda açık hali bulunan **VIEW** menüsü görünüm denetimlerini kontrol eden komutları içerir.

CODE: Seçili olan forma ait kod penceresini görüntüler.

OBJECT: Seçili olan formun tasarım şeklini görüntüler.

DEFINITION: Bu komut ile seçili olan değişken, sabit veya tipinin tanımlandığı yere odaklanır.

LAST POSITION: Kursörü entere basılmadan önceki son pozisyona odaklar.

OBJECT BROWSER: Bu seçenek ile üzerinde çalışılan projenin ve VB kütüphanelerinin bulunduğu pencereye gidilir.

IMMEDIATE WINDOW: Immediate penceresine gider. Bu pencere debug modunda değişkenler üzerinde işlem yapmak için kullanılır.

LOCALS WINDOW: Locals penceresi görüntülenir. Bu pencere de debug modunda kullanılır. O anda projede bulunan kontrol elemanlarının özelliklerine ait pencereye local penceresi denilmektedir.

WATCH WINDOW: Watch Penceresi görüntülenir. Bu pencere de debug modunda kullanılır ve DEBUG_ADD Watch menüleri ile eklenen değişkenlerin o anki durumlarını gösterir. F7 ve F8 tuşlarına basılarak değişkenlerin program çalışması esnasındaki değişimi adım adım izlenebilir.

PROJECT EXPLORER: Proje dosyalarının gösterildiği pencereye ulaşılır.

PROPERTIES WINDOW: Properties penceresini görüntülemek için kullanılır.

FORM LAYOUT WINDOW: Bu seçenek ile formun ekrandaki konumunu gösteren bir pencere açılır. Bu pencere aracılığı ile formu istenilen yere tasimak mümkündür. Bu pencerede iken farenin sag tusuna basildiginda karsimiza çıkan popup menüde Startup Position seçeneğinin içerisindeki



MANUAL: fare ile istenilen yere formu taşıyabilmek için seçili olmalıdır.

CENTER OWNER: Programa ait açık pencere varsa onun ortasında açar

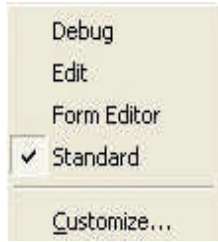
CENTER SCREEN: Ekranın tam ortasında formu odaklar.

WINDOWS DEFAULT: Standart konuma odaklar.

TOOLBOX: Kontrol elemanlarının yer aldığı toolbox menüsüne ulaşmakta kullanılan seçenektir.

COLOR PALETTE: Renk penceresini görüntülemek için kullanılır.

TOOLBAR: Bu seçenek ile VB'de kullanılan araç çubukları gösterilip gizlenebilir. Bu seçenek kullanıldığında karsimiza aşağıdaki menü çıkar.



DEBUG: Bu seçenek debug araç çubuklarını gösterir yada gizler.

EDIT: Bu seçenek te edit araç çubuklarını gösterir yada gizler.

FORM EDITOR: Bu seçenek ise Form editör araç çubuğunu gösterip gizlemede kullanılır.

STANDART: Standart seçeneği standart araç çubuklarını gösterip gizlemede kullanılır.

CUSTOMIZE: Bu seçenek ile bir çok işlem yapmak mümkün olmaktadır. Bunları şöyle sıralaya biliriz:

- 1) Visual Basic menülerinin yerini değiştirmek veya Türkçeleştirmek.
- 2) Yeni araç çubukları oluşturulabilir, silinebilir, ve yeniden adlandırılabilir. Yapılan değişiklikler iptal edilebilir.
- 3) Customize menüsü açıkken menülerin yerini değiştirmek te mümkündür

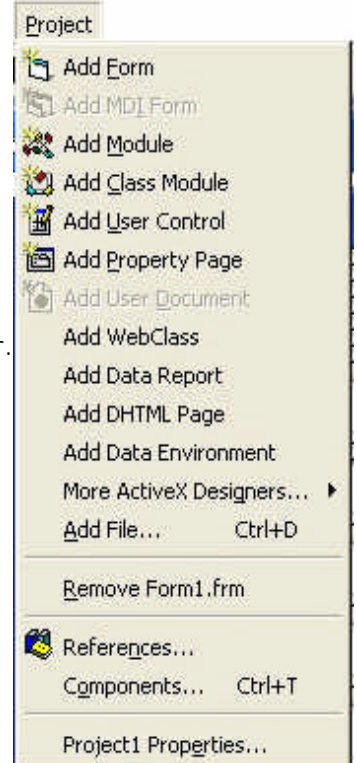
PROJECT MENÜSÜ

ADD FORM: Projeye yeni bir form eklemek için kullanılır.

ADD MDI FORM: Projeye yeni bir MDI form eklemek için kullanılır.

ADD MODULE: Projeye yeni bir module eklemek için kullanılır.

ADD CLASS MODULE: Projeye yeni bir class module eklemek için kullanılır.



Microsoft Visual Basic 6.0

ADD USER CONTROL: Projeye yeni kullanıcı kontrolü eklemek için kullanılır.

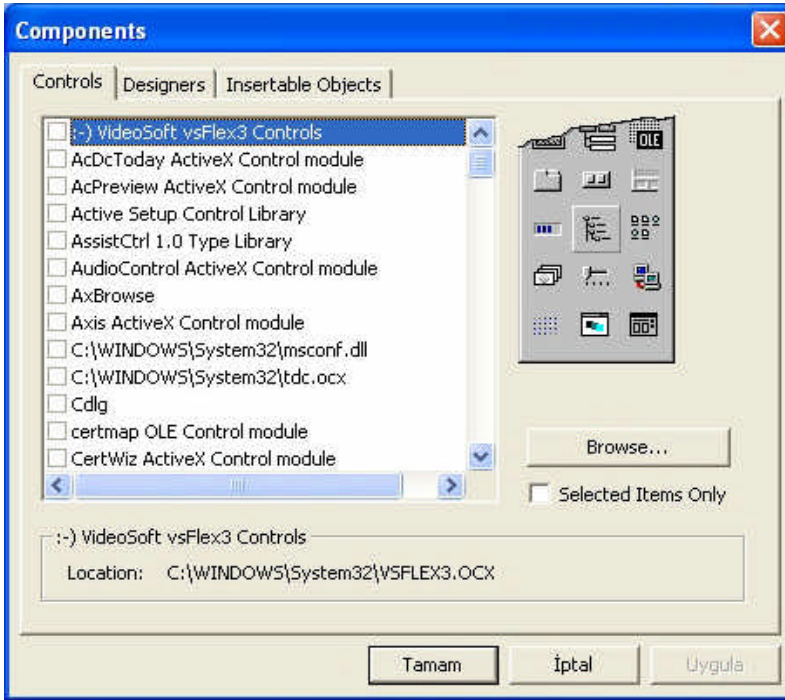
ADD PROPERTY PAGE: Projeye yeni bir property page eklemek için kullanılır.

ADD FILE: Projeye yeni bir EXE, CLS, BAS uzantılı dosya eklemek için kullanılır.

REMOVE FORM1: Projeden seçili olan nesneyi kaldırır.

REFERENCES: Referans penceresi görüntülenerek burada object browser da gözükecek veri kütüphaneleri seçilir yada devreden çıkarılır.

COMPONENTS....: Components penceresi görüntülenerek Toolbox penceresi eklenecek yada kaldırılacak elemanlar seçilir.



PROJECT PROPERTIES: Burada proje ile ilgili özelliklerin ayarları yapılır. Mesela program her derlendiğinde versiyonun bir artırılması gibi...

Projede birden fazla form varsa, çalışmaya hangi formu başlanacağı, programda kullanılacak bazı dosya isimleri de bu pencerede ayarlanır.

FORMAT MENÜSÜ:

ALIGN: VB'de birden fazla kontrol elemanini aynı hizaya getirmek oldukça zahmetli ve sıkıcı bir iştir. İşte bu seçenek seçildiğinde karşımıza çıkan alt



menüsündeki sağdaki durumlarla çeşitli yönlerden çoğul kontrol elemanlarını tek bir seçimle, kolaylıkla yapabilmemizi sağlamaktadır.

LEFTS: Seçili olan elemanların sol taraflarını hizalar.

CENTERS: Bu seçenek seçildiği takdirde elemanların merkezleri hizalanır.

RIGHTS: Seçili elemanların sol tarafları hizalanır.

TOPS: Seçili olan elemanların üst taraflarını hizalar.

MIDDLES: Seçili elemanları üst üste getirerek aralarını hizalar.

BOTTOMS: Seçili olan elemanların alt taraflarını hizalar.

TO GRID: Seçili elemanların form üzerindeki noktalara denk gelmesini sağlar.



MAKE SAME SIZE: Form üzerinde birden fazla kontrol elemanının yükseklik ve genişliklerini aynı seviyeye ayarlamayı kolaylaştıran bu seçenek kendisine ait alt menüde;



WIDTH: Seçili elemanları aynı genişlikte yapmaya yarar.

HEIGHT: Seçili elemanların yüksekliklerini aynı seviyeye ayarlar.

BOTH: Seçili elemanların hem yüksekliğini hem de genişliğini aynı seviyeye ayarlamaya yarar.

SIZE TO GRID: Seçili elemanları form üzerindeki noktalara odaklar.

HORIZONTAL SPACING: Form üzerindeki birden fazla kontrolün arasındaki yatay boşlukları ayarlamak için kullanılır.

İlgili kontrolleri seçtikten sonra bu seçeneğe ait alt menüdeki;

MAKE EQUAL: Elemanların aralarındaki yatay boşlukları eşitler.

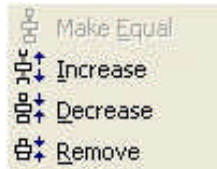
INCREASE: Elemanların aralarındaki yatay boşlukları artırır.

DECREASE: Elemanların aralarındaki yatay boşlukları azaltır.

REMOVE: Elemanların aralarındaki yatay boşlukları tamamen kaldırır.



VERTICAL SPACING: Form üzerindeki birden fazla kontrolün arasındaki dikey boşlukları ayarlamak için kullanılır. İlgili kontrolleri seçtikten sonra bu seçeneğe ait alt menüleri kullanılır.



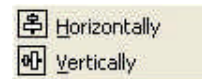
MAKE EQUAL: Elemanların aralarındaki dikey boşlukları eşitler.

INCREASE: Elemanların aralarındaki dikey boşlukları artırır.

DECREASE: Elemanların aralarındaki dikey boşlukları azaltır.

REMOVE: Elemanların aralarındaki dikey boşlukları tamamen kaldırır.

CENTER IN FORM: Bu seçeneğin alt menüsündeki seçenekler ile form üzerindeki seçili olan elemanları formun orta noktasına ayarlar.



HORIZONTALLY: Seçili olan elemanlari formun orta noktasina yatay olarak ayarlar.

VERTICALLY: Seçili olan elemanlari formun orta noktasina dikey olarak ayarlar.

ORDER: Form üzerindeki kontrollerden alt alta olanlar varsa bunlari üste veya alta almak için bu seçeneğin alt menüsündeki seçenekler kullanilir.



BRING TO FRONT: Altteki elani üste alır.

SEND TO BACK: Üstteki elemani alta alır.

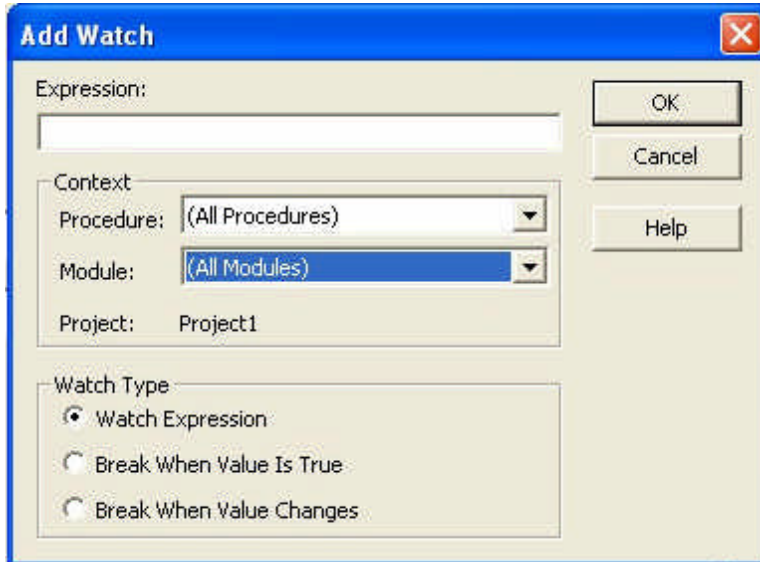
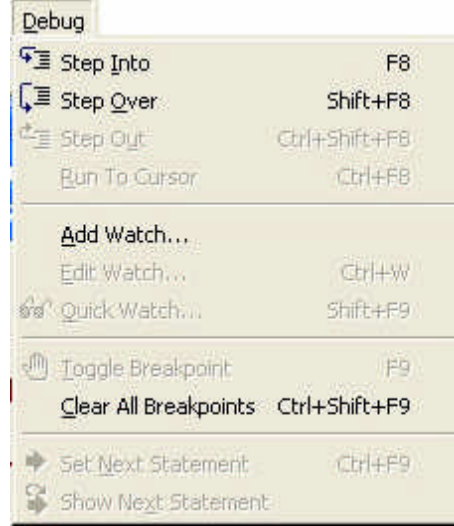
LOCK CONTROLS: Seçili olan elemanlari kilitleyerek bunlari yerlerinin degistirilmesini saglar.

DEBUG MENÜSÜ:

STEP INFO: Programi adim adim çalıştırmaya yarayan seçenektir.

STEP OVER: Bu seçenekte programi adim adim çalıştırırken procedure çağrılari bir satirmis gibi tek satirda islenir, procedurelere girilmez.

ADD WATCH: Asagida sekli görünen ADD WATCH penceresi ile program akisi esnasinda izlenecek degiskenleri seçmeye yarar.



Solda **BREAK WHEN VALUE IS TRUE** seçenegi seçili durumda iken program çalıştiginda alakali degiskene sifirdan farklı bir deger yüklenirse program kirilir.

BREAK WHEN VALUE CHANGES seçenegi seçili durumda iken program çalıştiginda alakali degiskene bir

CLEAR ALL **BREAK POINTS:** Tüm kesisme noktaları silinir. Değer yüklenirse program kilitlenir.

RUN MENÜSÜ:



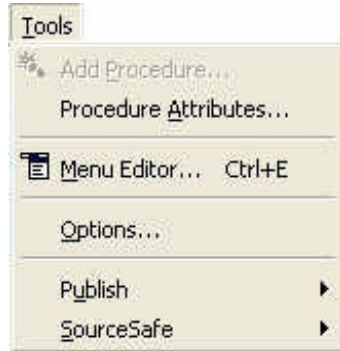
START: Tasarımı tamamlanmış ya da çalışması durdurulmuş bir programı çalıştırır.

BREAK Çalışmakta olan bir programı geçici olarak durdurur.

END: Çalışan bir programın çalışması sona erdirilir.

RESTART: Geçici olarak durdurulmuş bir programı kaldığı yerden çalışmasına devam ettirir.

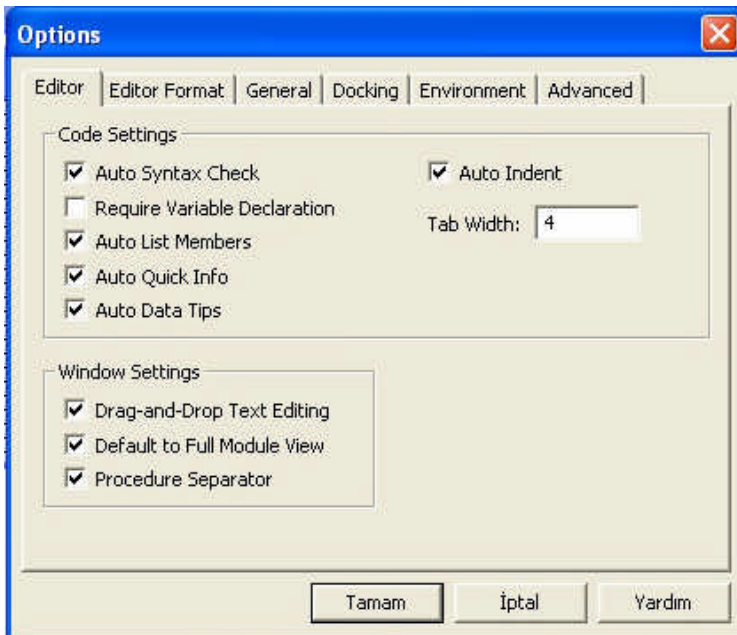
TOOLS MENÜSÜ:



ADD PROCEDURE: Kod penceresine (Function), alt program (Sub), Olay (Event) ve Özellik (Property) eklemek için kullanılır.

MENU EDITOR: Bu seçenek ile menü tasarım penceresine ulaşılır. Bu pencere aracılığı ile istenilen menü tasarımı yapılabilir.

OPTIONS: Bu seçenek seçildiğinde karşımıza gelecek pencere vasıtası ile programın kontrolünü etkileyecek bir çok değişiklik yapılabilmektedir.



Bu pencerenin Editör kısmıyla aşağıdaki işlemler yapılabilir.

AUTO SYNTAX CHECK: Bu seçenek seçili duruma getirilirse kod penceresinde yazılan kodlar otomatik olarak yazım kontrolüne alınır.

REQUIRE VARIABLE DECLARATION: Bu seçenek seçili duruma getirildiği takdirde kullanılan her değişkene tanımlanma mecburiyeti getirilir.

Microsoft Visual Basic 6.0

AUTO LIST MEMBERS: Bu seçenek seçili duruma getirildiğinde kontrol elemanların adı yazıldığı takdirde ilgili olayları liste halinde ekrana getirilecek ve programcıya kolaylık sağlayacaktır.

AUTO QUICK INFO: Bu seçenek seçili duruma getirildiğinde kod penceresinde herhangi bir fonksiyon yazılmaya başlandığında o fonksiyonun parametre kalibi ekranda belirir ve programcının hata yapması engellenir.

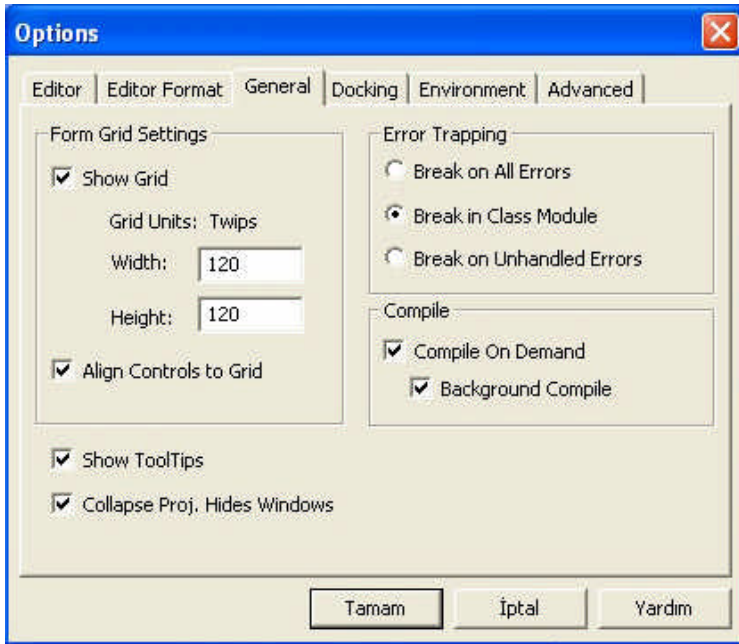
AUTO DATA TIPS: Bu seçenek seçili olduğunda debug modunda fare işaretçisi bir değişkenin üzerine getirildiği takdirde ekranda o değişkenin o anki değeri gösterilir.

AUTO INDENT: Bu seçenek seçili olduğunda kod penceresinde bir satıra tab uygulanırsa o satırlardan sonraki satırlarda da otomatik olarak aynı sütun seviyesinden başlatılır.

DEFAULT TO FULL MODULE VIEW: Bu seçenek seçili olduğunda kod penceresinde bütün kodlar aynı anda görüntülenir.

PROCEDURE SEPARATOR: Bu seçenek seçili ise kod penceresinde prosedürler arasında çizgi işareti gösterilir.

Editör kısmında kod penceresinde yazılan kodların renk, font, büyüklük vb.. ayarlanabilir.



Yandaki **GENERAL** durumu seçili durumdaki haliyle aşağıdaki durumlar ayarlanabilir.

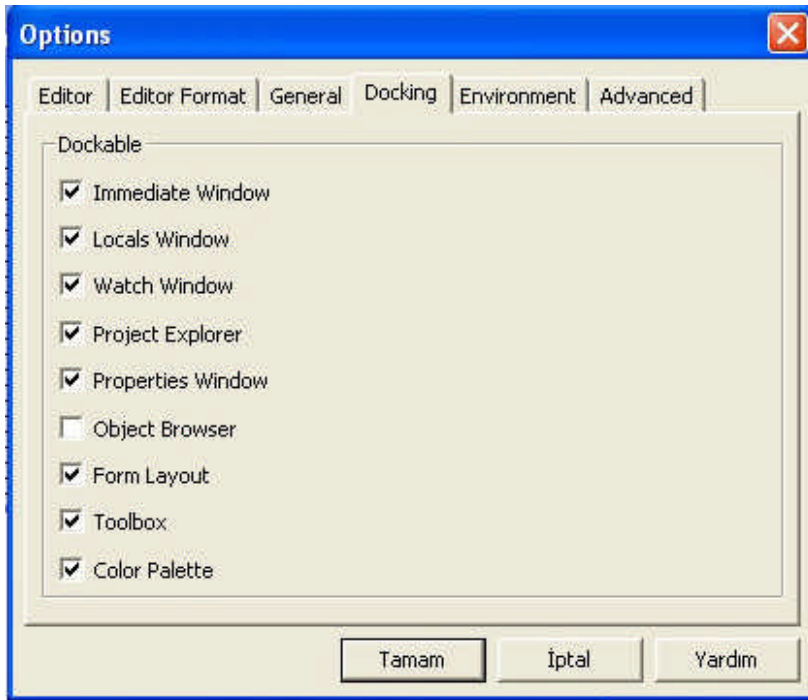
SHOW GRID: Bu seçenek seçili ise form üzerinde bulunan noktalar gösterilir.

WIDTH: Bu seçenek yanındaki kutuya verilen değer, form üzerindeki noktalar arası yatay mesafenin değeridir.

HEIGHT: Bu seçenek yanındaki kutuya verilen deger, form üzerindeki noktalar arasi dikey mesafenin degeridir.

ALIGN CONTROLS TO GRID: Bu seçenek seçili ise form üzerindeki kontroller noktalara isabet ettirilir.

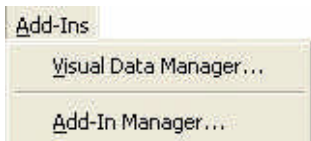
BREAK ON ALL ERRORS: Bu seçenek seçili iken derleyici herhangi bir hatayla karsilasirsa derleme islemi kirlir.



OPTION

penceresindeki bir diger iç pencere de sol tarafta görüldüğü gibi **DOCKING** tir. VB'de kullanılan pencerelerin gurup olarak gösterilip gösterilemeyecegi belirlenir. Sol tarafındaki checkbox kutucugu isaretli olan seçenekler programda gösterilir.

ADD-INS MENÜSÜ:



ADD IN MANAGER: Bu kontrol ile user kontrolde kullanılacak sihirbazlar aktif yada pasif hale getirilir.

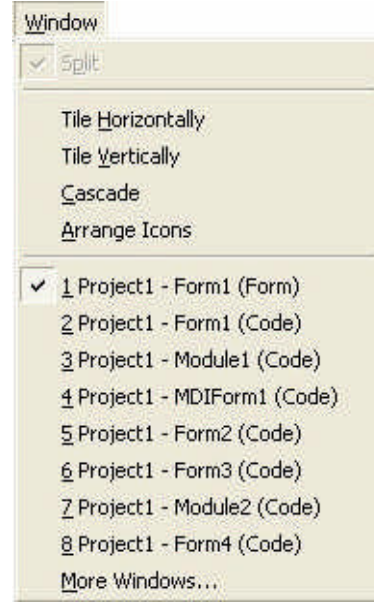
WINDOW MENÜSÜ:

TILE HORIZANTALLY: Projede kullanılan ve o anda görünür olan formlari, kod pencereleri vb.. yatay olarak alt alta siralar.

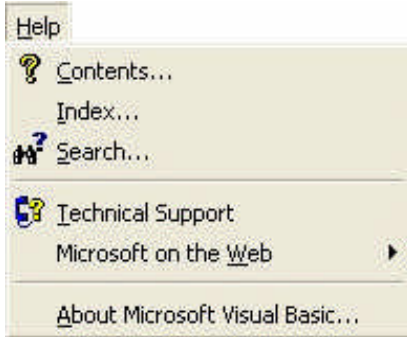
TILE VERTICALLY: Projede kullanılan ve o anda görünür olan formlari, kod pencereleri vb.. dikey olarak yan yana siralar.

CASCADE: Projede kullanılan ve onda görünür olan formlari, kod pencereleri vb.. yatay olarak aktif olan üstte kalacak sekilde siralar.

ARRANGE ICONS: Projede kullanılan ve onda görünür olan formlari, kod pencereleri vb.. minimize konumuna indirger.



HELP MENÜSÜ:



Bu menüde konularina ulasmak ve istenilen bilgiyi elde etmek mümkün. Ayrica VB'nin özellikleri ve etiketi de burada bulunmaktadır. Web sayfası aracılığı ile MICROSOFT'a ulasmak için kısa yollar da burada mevcuttur.

4-VISUAL BASIC TEMELLERİ

Bir Visual Basic Programının Yapısı

Bir Visual Basic programı bir proje olarak geliştirilir. Proje vbp uzantılı bir dosyadır. Proje içinde form ve modül gibi diğer bileşenler yer alır.

Proje Kavramı

Bir uygulama geliştirme sürecinde gerekli birimleri oluşturmak için bütünsel bir ortamda çalışılır. İşte programın bu birimlerinin toplandığı bu çalışmaya proje (project) denir. Yeni bir Visual Basic programına başladığınızda yeni bir proje yaratırsınız. Bunun geçici adı Project1 olur. Ardından proje içinde formlar, modüller ve diğer bileşenleri yaratırsınız.

Bir proje şu birimlerden oluşur:

- Bir proje dosyası. Bütün elemanları takip etmeyi sağlayan bu dosyanın uzantısı vbp'dir.
- Her form için bir frm dosyası.
- Her formun özellikleri için binary (ikili) bir dosya.
- Her class modülü için bir cls dosyası (seçenek).
- Her standart modül için bir bur. dosyası (seçenek).
- ActiveX kontrollerini içeren bir veya daha fazla dosya. Bu dosyaların uzantısı ocx'tir. (seçenek)
- Bir kaynak dosyası res uzantılı (seçenek).

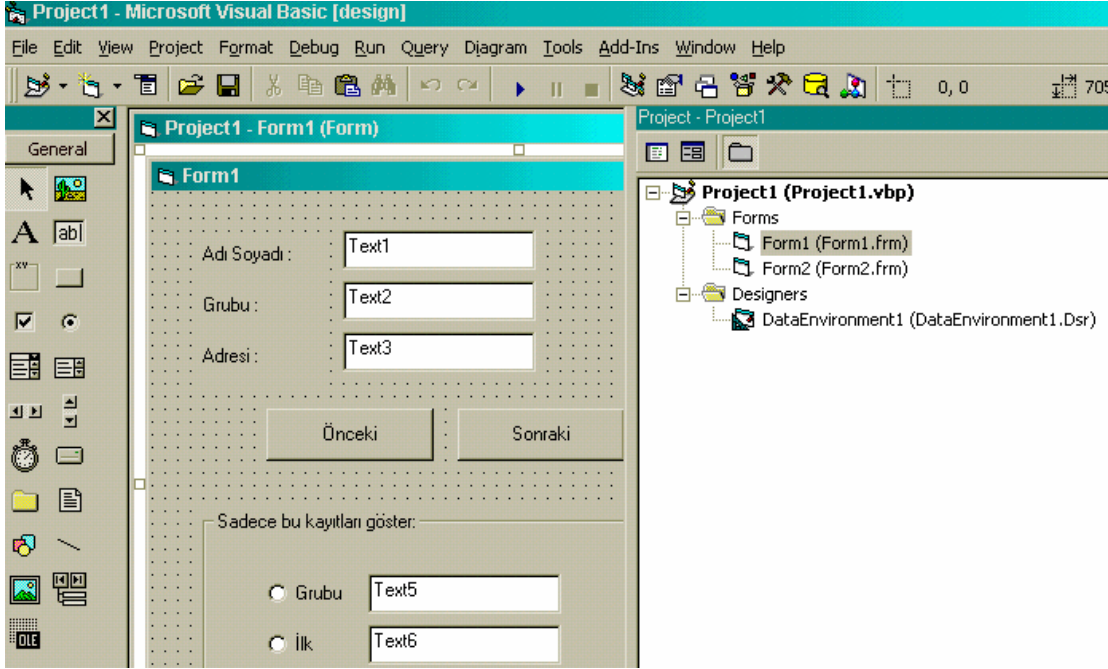
Bir proje dosyası proje ile ilgili nesneleri içeren bir dosyadır. Proje dosyası aynı zamanda projeye özel ortamın da kayıt edilmesini sağlar. Proje dosyası istenirse exe dosya haline çevrilerek doğrudan çalışması sağlanır.

Projeye Başlama

Visual Basic'in başlatılmasıyla beraber ya da bir proje içinde yeni bir uygulamaya başlamak için File menüsünden New Project komutu kullanılır. Ardından geliştirilecek programın Standard EXE ya da diğer işletilebilir birim şekli seçilir. Ardından boş bir form ile Visual Basic tümleşik program geliştirme ortamı programcının karşısına çıkar. Uygulamanın arabirimi ve kodları yazıldıktan sonra bütün kodlar bir proje olarak kayıt edilir. Projenin kayıt edilmesinden önce formların da kayıtları yapılır. Formlar frm olarak, projeler de vbp dosyası olarak kayıt edilir.

Project Explorer

Projeye formlar ve modüller eklendikçe proje hiyerarsik bir sekle girer. Bu durumda proje birimlerinin görülmesi ve islern yapılması için Project Explorer kullanilir. Projeye bir birimin eklenmesi ya da çıkarılması gerektiğinde Project Explorer görünümü sayesinde projenin bütün elemanlari görülür ve kolayca seçilebilir.



Sekil 4.1 Project Explorer

Formlar ve Modüller

Diger programlama dillerinde olduğu gibi Visual Basic'in de belli bir kod kurallari ve yöntemleri vardır. Visual Basic kodu modül içinde saklanır. Üç tür modül ya da diger bir deyişle kodun saklandığı yer vardır:

- Form modülü
- Standart modülü
- Class modülü

Basit uygulamalarda genellikle bir form bulunur. Bütün kodlar bu formun modülünde (kod alanı) yer alır. Uygulama büyüdükçe ek formlar projeye

eklenerek modül sayısı artırılabilir. Projeye modül eklemek için Project menüsünden Add Module komutu kullanılır.

Her modülde ayrı ayrı kodlar yazmanın yanı sıra bir de bütün formlarda ortak olarak kullanılacak kodlara gereksinim duyulduğunda o zaman standart modüller kullanılır. Class modülleri ise özel metod ve özellikleriyle yeni bir nesne yaratmak için kullanılır. Bütün modüller şu bileşenleri içerir:

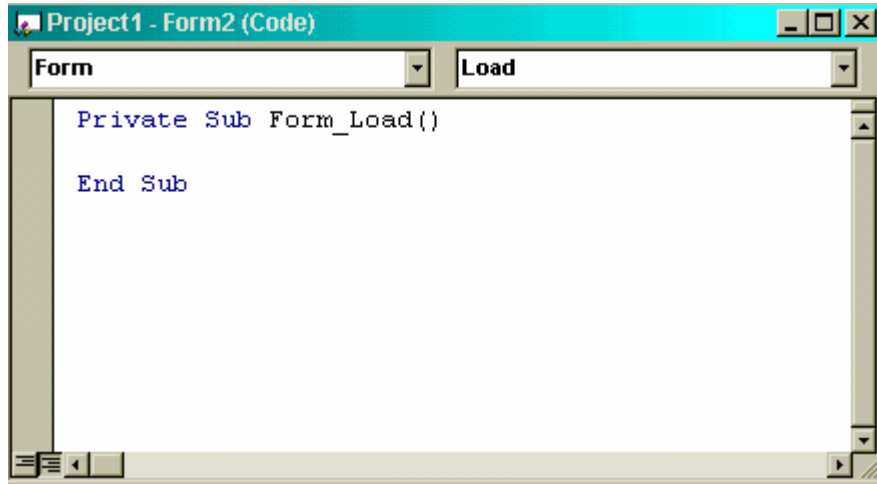
- Declarations (tanımlamalar)
- Procedures (yordamlar)

Tanımlamalar; verilerin tanımlandıkları alanlardır. Modülün en başında olan bu kısımda sabitler, değişkenler, dinamik bağlantı kütüphaneleri tanımlanır.

Procedure'ler ise Sub, Function ya da Property olmak üzere bir blok kodu ifade ederler. Procedure'ların bir diğer özelliği de bir seferde işletilmeleridir. Örneğin bir düğmeye tıklayınca bir procedure'ın çalışması gibi.

Form Modülleri

Form modülleri (FRM dosya uzantılı) bir Visual Basic uygulamasının temelini oluşturur. Form modülleri olay yordamlarını, genel yordamları, ayrıca form düzeyinde veri tanımlamalarını içerir.



Sekil 4.2 Form Modülü

Form modülleri içinde yapılan tanımlamalar ve yordamlar formun ait olduğu uygulamaya aittir.

Bir Modülün Kesimleri

Tanımlama Kesimi:

Procedurler:

```
Sub Hesapla_Click
```

```
...
```

```
End Sub
```

```
Sub İlkIslemler
```

```
...
```

```
End Sub
```

```
Sub Rapor
```

```
...
```

```
End Sub
```

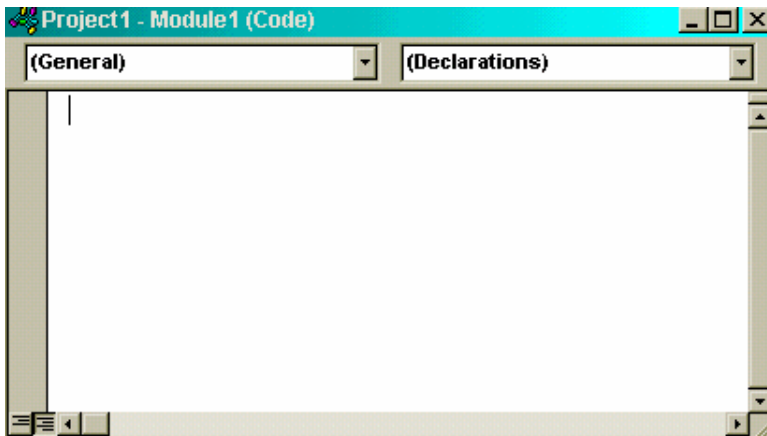
```
Function Komisyon ( x, y,z)
```

```
...
```

```
End Function
```

Standart Modüller

Standart modüller (KAS dosya uzantili) uygulama içerisindeki diğer modüllerden de ulaşılabilen yordam ve tanımlamaları içerirler. Bu modüller uygulamanın her yerinden erişilebilen (global) ya da modül düzeyinde tanımlamaları içerirler. Burada yazılan kodlar belli bir uygulamaya ait olmak zorunda değildir. Standart modüller birçok uygulama tarafından kullanılabilir.



Sekil 4.3 Standart Modül

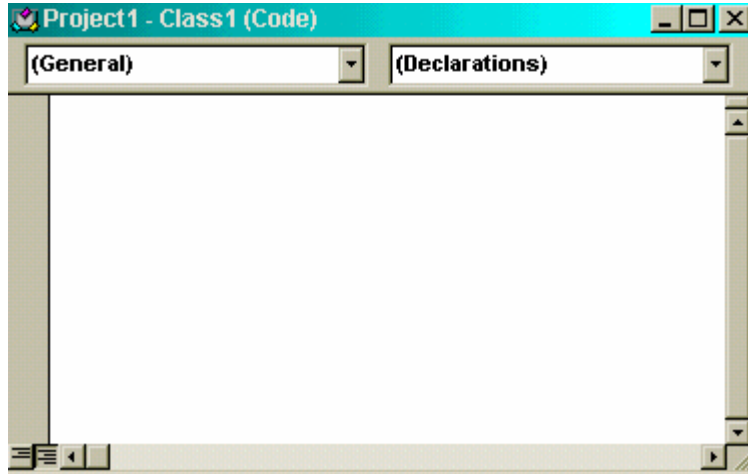
Class Modüller

Class modüller (CLS dosya uzantili) nesneye dayali uygulama gelistirmenin temelidir.Class modüllerinde yazilan kodlarla yeni nesneler yaratilir.Bu nesneler kendi özel metot ve özelliklerine sahiptirler.

Class'lar program içinde kullanılan birimlerdir. Class'lar yeniden kullanılabilir nesneler yaratirlar. Bir class, veri islemek için procedure'lari içerir.Visual Basic kontrolleri de birer class'tirlar. Bütün programlarda yer alirlar.

Class kullaniminin iki yolu vardir. Birincisi program içinde yeniden kullanılabilir nesneler yaratmak. Ikincisi ActiveX DLL ve ActiveX EXE programlar yaratmaktir.

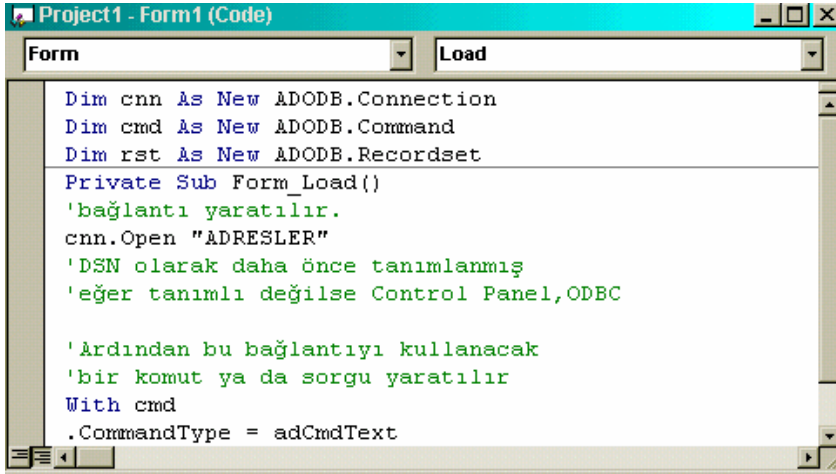
Bir Class modül yaratildikten sonra kendi "özelliklerinizi" yaratabilirsiniz. Projeye yeni bir Class modül eklemek için: Proiect menüsünden Add Class Module komutu seçilir.Bir proje içinde yaratilan Class modülüne sadece o proje içindeki modüllerden veformlardan erisilir. Ancak ActiveK olarak düzenlenen class'lara diger projelerden de erisilebilir.



Sekil 4.4 Class Modül

Bir Procedure' un Yazilmasi

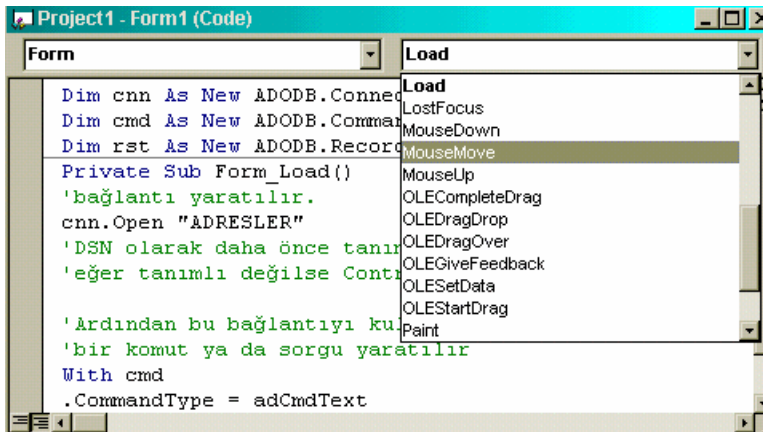
Bir proje içinde bir form üzerinde bir kontrolün yerlestirilmesinin ardindan penceresine geçilir ve form modülü içinde procedure' ler yazilir.Her modül penceresi bir kod editörü olarak kullanilir.



Sekil 4.5 Kod Editörü

Kod editörü içinde nesnelere göre ayrı kod kesimleri vardır. Nesne listesinden bu bölümlere geçmek mümkündür. Bir form modülünde nesne listesi form içinde yer alan kontrolleri ve onların çeşitli olaylara göre sahip oldukları olaylara göre kodları içerir.

Kod editörü olarak kullanılan modül penceresi içinde çok sayıda yordam yer alır. Bu yordamlara Procedure listesinden ya da Object listesinden erişilebilir. Örneğin Command 1 kontrolü Click olayı ile ilgili olarak bir kod kesimine sahiptir.



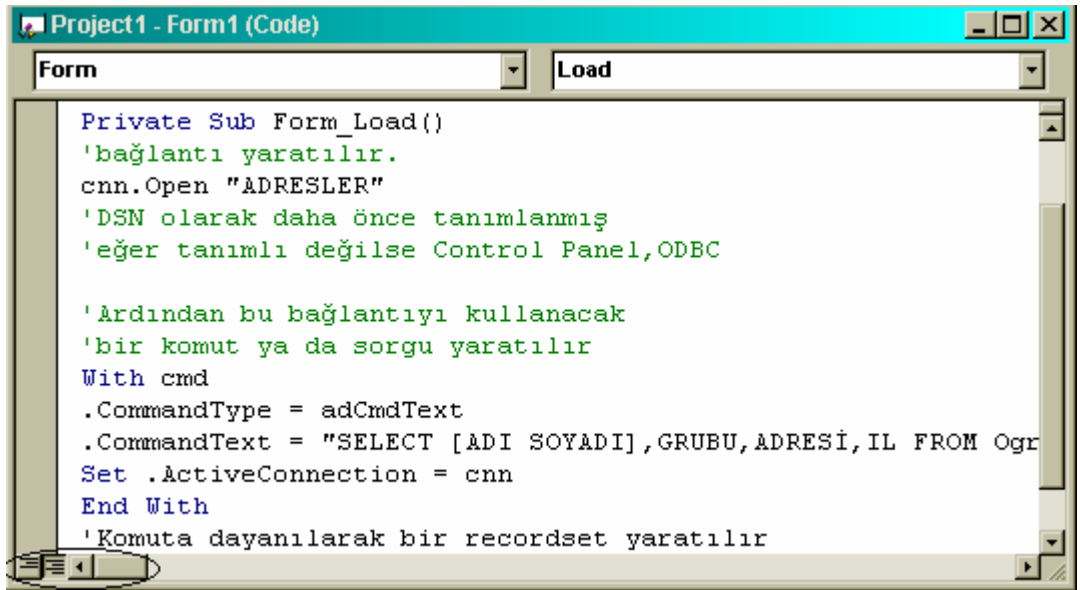
Sekil 4.6 Olaylar

Class modülünde ise genel kesim ve class kesimi bulunur. Standart modüllerde ise sadece genel kesim yer alır. Class modülleri sadece kendisinin baslatıp sonlandırabileceği olay yordamlarını içerirler. Standart modüller ise olay yordamları içermezler bu nedenle sadece General bölümünde tanımlamalara ve genel amaçlı yordamlara sahiptirler.

Tanımlamalar (declarations) kısmında yapılır. Bu bölümde değişkenler, sabitler ve DLL tanımlamaları yapılır. Sub ve Function'lar eklendikçe Procedure listesine eklenirler.

İki Ayri Görünüm

Kod editörü penceresinde bir procedure ya da bütün procedure'lar olmak üzere iki görünüm şekli vardır. Procedure View görünümünde; kod penceresinde sadece bir procedure (yordam) görünür. Full Module View görünümünde bütün procedure'lar görülür. Görünümler arası geçişte modül penceresinin sol alt köşesinde yer alan düğmeler kullanılır.



Sekil 4.7 Görünümler

Bir Procedure' un Yapısı

Bir Visual Basic programı çok sayıda yordamdan (procedure) oluşur. Yapılacak işlemler birer yordam olarak tasarlanırlar. Yordamlar bir program blogu ya da bir program parçası olarak düşünülebilir. Program tasarımı yapılarını saymakla bitiremeyeceğimiz yordamların yararları:

- Yordamlar program işlevlerinin; birbirinden bağımsız küçük birimlere bölünmesini sağlar.

- Yordamlar program hatalarının daha kolay bulunmasını ve giderilmesini sağlar.
- Yordamlar olaylara bağlanarak olay temelli programlamayı oluşturur.
- Yordamlar program bloklarının diğer programlarda da kullanılmasını sağlar.
- Yordamlar birbirlerini kolayca çağırabilir.
- Yordamlar istenildiği kadar yinelenabilir.

Visual Basic'de değişik türde yordamlar (procedure) kullanılır: Sub, Function, Property. Bir Sub yordamı herhangi bir değer döndürmez. Çağrılarak ya da bir olaya tepki olarak çalışır. Function yordamları ise bir değer döndürürler. Örneğin bir faiz hesabının sonucunu döndürürler. Property yordamları ise bir değer döndürürler, atama yaparlar ve nesnelerin referanslarını düzenlerler.

Bir Sub yordamının yapısı:

[Private / Public] [Static] Sub yordam adı (argümanlar)

Deyimler

End Sub

Yordam çalıştırıldığında Sub ve End deyimleri arasında kalan satırlar işletilir. Sub yordamları standart modüller, class modüller ve form modülleri içinde yer alırlar. Sub yordamları varsayım olarak bütün modüllerde Public tanımlanır. Bunun anlamı bu yordamların uygulamanın her yerinden (diğer modüllerden) çağırılabilmesidir.

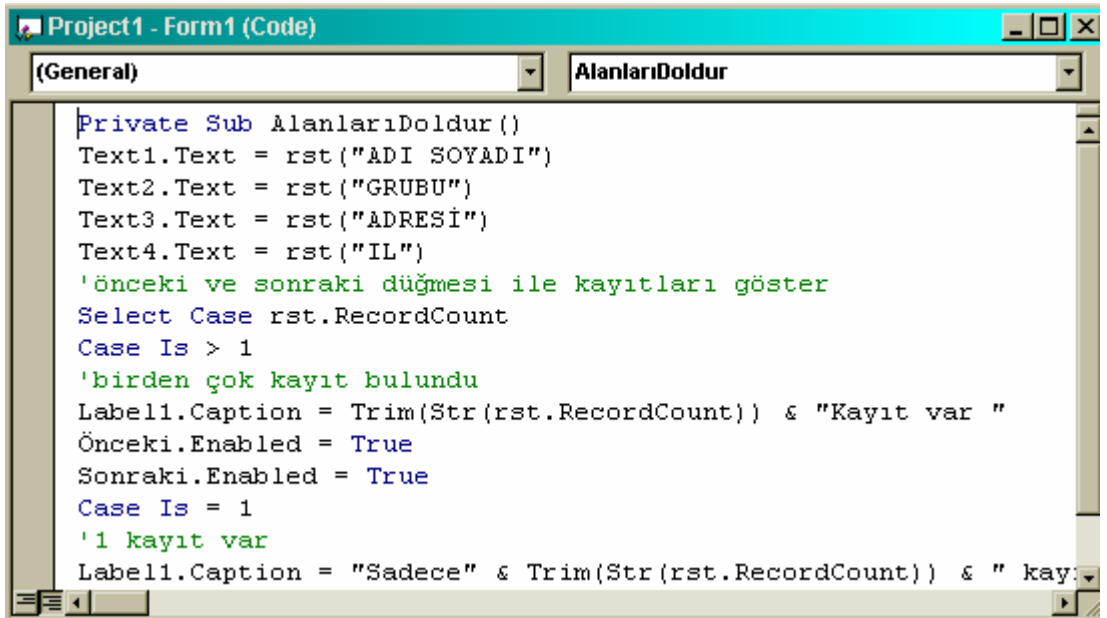
Argümanlar ise tanımlanan bir değişken gibidir. Çağırılan yordamdan geçen değerler olarak tanımlanır. Visual Basic'te yordamlar (procedure) ikiye ayrılırlar: Genel yordamlar, Olay yordamları.

Genel Yordamlar

Bir genel procedure belli bir işlemi üstlenen program parçasıdır. Örneğin bir dosyanın açılması ya da raporun bastırılması için kullanılabilir. Genel yordamlar diğer yordamlardan çağırılırlar. Böylece belli bir işlevi olan bu birim programın her yerinden istenildiği kadar çağırılarak kullanılır.

Buna karşın olay yordamları ise kullanıcının yarattığı bir olay nedeniyle çalışır. Olay yordamları kullanıcıların hareketlerine tepki olarak çağırılırlar. Örneğin bir düğmeye tıklayarak bir işlemin başlatılması. İyi bir programlama stratejisi içinde bir takım ortak görevler genel yordamlar olarak planlanmalıdır. Böylece kod tekrarına gerek kalmaz.


```
Sub yordam adi (arguments)
    -deyimler-
End Sub
```



Sekil 4.8 Genel yordamlar

Olay Yordamları

Olay yordamları bir olayın (event) oluşmasına tepki olarak çalışırlar. Visual Basic nesneleri (formlar, metin kutuları, düğmeler, vb) belli olayları tanırlar. Böylece olay oluştuğunda; o olay için hazırlanmış olay yordamı çalışır.

Bir komut düğmesinin olayları:

Click	Tıklama
GotFocus	Odaklanma üzerine gelme.
KeyDown	Bir tusa basmak.
MouseDown	Farenin bir tusuna basmak.

Microsoft Visual Basic 6.0

Olay yordamları, bir nesnenin alt tire ile olaylara bağlı olarak geliştirilir. Bu adlar Visual Basic kod editörü tarafından otomatik olarak ya da manuel olarak oluşturulur.

Yapisi: Nesne_Olay
Örnek: Command1_Click

Olay yordamları Visual Basic nesnelerinin olaylarına bağlı olarak geliştirilirler: Örneğin Form1_Load olay yordamı bir formun açılması sırasında çalışacak bir yordamı, Command1_Click olay yordamı da bir komut düğmesine tıklandığında çalışacak bir kod birimini gösterir.

```
Sub KontrolAdi_OlayAdi(argümanlar)
    -deyimler-
End Sub
```

Olay yordamlarının adını kod editörü içinde doğrudan yazabileceğiniz gibi Visual Basic tarafından otomatik olarak koda eklenen yordam ve olay adlarını da kolayca kullanabilirsiniz. Bu işlem için kod editörü penceresinde nesne ve olay seçilir.

1. Kod editörü penceresinde Object kutusundan istediğiniz nesneyi seçin.
2. Procedure kutusundan istediğiniz prosedürü seçin.
3. Sub prosedürünüz hazır.

Fonksiyonlar

Visual Basic procedure'ları sadece Sub olarak düzenlenmez. Bir diğer procedure yapısı da Function olarak bilinen fonksiyonlardır. Bir procedure olarak düzenlenen fonksiyonlar belli bir işlemi kod olarak içeren ve bir değeri geri döndüren program parçalarıdır. Bir yordam olarak yaratılan, fonksiyonların Sqr, Val gibi Visual Basic'te hazır olarak bulunan fonksiyonlarla ilişkisi yoktur. Fonksiyonlar kullanım bakımından bir işlemi yapan ve istenildiği yerde kullanılabilen bir kod birimidir. Örneğin bir faiz fonksiyonu istenildiği program içinde istenilen yerde çağırılarak kullanılır.

Bir fonksiyon yordamının yapısı:

```
[Private |Public][Static] Function yordam adi (argümanlar) [As tip]
    -deyimler-
End Sub
```

Fonksiyonlar ayrı bir yordam olarak düzenlenirler. Fonksiyonların kullanımında argümanlar önemlidir. Fonksiyonlar argümanları alırlar, bir dizi

işlem yaparlar ve sonucu döndürürler. Sub yordamlar ile Function yordamlar arasında farklar şunlardır:

- Bir fonksiyon değer döndürür ve genellikle bir eşitliğin sağında kullanılır.
- Bir sub ise belli bir işi yapar. İşlemi tamamlar bir değer ya da parametre ile kullanım zorunluluğu yoktur.

```
Function UcretHes(deger1,deger2,deger3)
    -deyimler-
End Function
```

Yapisi: **Ucret=UcretHes(gun,baz,katsayi)**

Fonksiyon yordamlar bir değeri elde etmemizi sağlarlar. Fonksiyonlar değişken gibi veri tipine sahiptirler. Bu veri tipi, dönecek veri tipini belirler. Fonksiyonlar bir hesaplamanın parçası da olabilirler.

Toplam= İkramiye+UcretHes(gun,baz,katsayi)/360

Örneğin çapı verilen bir dairenin alanını hesaplayan bir fonksiyon ya da verilen dereceyi Fahrenheit' e çeviren bir fonksiyon gibi.

Örnek:Dereceyi Fahrenheit'e çeviren fonksiyon.

```
Function Fahrenheit(x)
    Fahrenheit = x*9/5+32
End Function
```

Fonksiyonun kullanımı:

```
Private Sub Command1_Click ( )
    Dim Deger As Integer
    Deger=InputBox(" ")
    MsgBox Fahrenheit(Deger)
End Sub
```

Yeni Bir Yordam Yaratma

Olay temelli geliştirilen kod yazma ortamında bir procedure yaratmak oldukça kolaydır.

- Bir genel yordam yaratmak.
- Bir olay yordamı yaratmak.

Microsoft Visual Basic 6.0

Genel bir yordam yaratmak için kod penceresinde genel bölümüne gidilir. Burada yordam yazılır.

```
Sub DosyaGuncelle ( )  
    -deyimler-  
End Sub
```

Ya da bir Function yordam:

```
Function Ikramiye (x, y)  
    -deyimler -  
End Function
```

Olay yordami yaratmak ise olusturulan uygulamada yer alan nesnelerin sahip olduklari olaylarin seçilmesiyle saglanir.

Yordamlarin Çağırılması

Olay temelli yordamlar bir olayla iliskilidir. Örneğin form açılır ve Load olayı oluşur. Bu nedenle Form1_Load adlı yordam çalışmaya başlar ya da bir düğmeye tiklanır ve Command1_Click yordamı çalışır. Olay yordamlarının çalıştırılmasının yani sıra diğer yordamların çalıştırılması için de çağırılma (call) işlemi kullanılır. Çağırma işlemi; program kodu içinde yordamın adının yazılmasıdır. Çağırma işlemi için istenirse Call deyiimi de kullanılır. Ancak Call deyiimi kullanılmadan da sadece yordamın adını yazmak yeterlidir.

Yapisi: Call yordam adı (parametreler)
ya da
Yordam adı parametreler

Örneğin bir yordamın içinde, yapılacak bir işlem için başka bir yordam çağırılır.

```
Private Sub ZeminRengi (ColorCode As Integer, MyForm As Form)  
    MyForm. BackColor = QBColor (ColorCode)  
End Sub
```

```
Private Sub Command1_Click()  
    Dim Renk As Integer  
    Renk = InputBox("hangi rengi istiyorsun ? (1-15 arası)")  
    sub çağır  
        ZeminRengi Renk, Form1  
End Sub
```

```
Private Sub Sonraki_ Click()  
    'Bir sonraki kayıda git  
    'eger EOF ise uyar ve sonuncu kayıda git.
```

```

With rst
    .MoveNext
    If .EOF Then
        Beep MsgBox "Sonuncu kayıt"
    .MoveLast
End If
End With
'alanlari doldurmak için bir yordam çağır
AlanlariDoldur
End Sub

```

Fonksiyon yordamların kullanımı ise zaten bir çağırma işlemidir.

```

ücret = UcretHes(gun,baz,katsayi)

```

Çağırma işlemi bir modül içerisindeki yordamlar arasında yapılır. Yani bir yordam aynı modül içindeki diğer bir yordamı çağırabilir. Bir yordamın diğer modüllerden de çağırılabilmesi için Public olarak tanımlanması gerekir. Diğer modüllerdeki yordamları çağırarak için kullanılacak söz dizimi yordamın form, standart modül ya da class modülde olmasına göre değişir.

Parametre Kullanımı

Bir procedure'dan diğer bir procedure 'e değer geçirmenin iki yolu vardır. Referans ve değer olarak. Genellikle referans biçimi kullanılır. Çünkü bu durumda çağırılan procedure gönderilen değeri kullanır. Kendisi bir değer ve ona bir bellek alanı ayırmaz.

```

Private Sub Command1_ Click()
    Dim A As Integer
    A = 10
    Ekle A
End Sub
Sub Ekle(x As Integer)
    'referans ile geçirme
    x = x+1
End Sub

```

Değer ile geçirmede ise ByVal ifadesi kullanılır. Geçirilen bu değere yerel bir bellek ayrılır ve karşılık gelen argümanı bu değere kopyalar.

```

Sub Ekle(ByVal x As Integer)
    'değer ile geçirme
    x = x+1
End Sub

```

Diger Formlardaki Yordamlari Çağirmek

Diger bir formun modülünde yer alan yordami çağirmek için formun adi ve yordam belirtilir.

```
Call Form1.DosyaGonder (argümanlar)
```

Class Modüllerindeki Yordamlari Çağirmek

Diger bir class modülünde yer alan yordami çağirmek için da Class modülünün adi ve yordam belirtilir.

```
Dim ClassModülü as New Class1  
ClassModülü.DosyaGönder
```

Standart Modüllerdeki Yordamlari Çağirmek

Diger bir standart modülde yer alan yordami çağirmek için ise modülün adi ye yordam belirtilir.

```
Modüle2.DosyaGönder (argümanlar)
```

Eger yordami adi uygulama için de tek (unique) ise modül adinin belirtilmesine gerek kalmadan yordamin adi yazilir.

Kendi-Kendini Çağirma

Procedure'lar degiskenleri için sinirli bir bellek alanina sahiptirler. Bir procedure kendini çağirdiginda daha fazla bellek alanina gereksinim duyulur. Iste procedure'larin kendilerini çağirma islemine öz-yineleme (recursive) procedure denir. Eger bir procedure kendisini sürekli çağirmaya devam öderse; eger herhangi bir kontrol konulmamissa o zaman hata verir.

Hata verecek olan yineleme:

```
Function RunOut(Maximum)  
    RunOut = RunOut(Maximum)  
End Function
```

Aşağıdaki procedure ise tipik olarak bir öz-yineleme işlemini gerçekleştirir. Fonksiyon istenilen faktöriyel sayısını hesaplamak için belli sayıda kendini çağırır.

```
Function Faktöryel(N)
    If N <= 1 Then ' sonuna ulasma
        Faktöryel=1
    Else ' N > 0 ise tekrar çağır.
        Faktöryel = Faktöryel (N - 1) * N
    End If
End Function
```

Öz-yinelemeli prosedürler etkin bir kullanım sağlarlar. Ancak bu prosedürlerin iyi bir şekilde test edilmesi gerekir. Özellikle bellek kullanımı bakımından test edilmeleri gerekir. Aksi takdirde kullanımda sorunlar oluşabilir. Bunun için:

- Gereksiz değişken tanımlamaları önlenmelidir.
- Variant tipli değişken dışında bir değişken kullanılmalıdır.

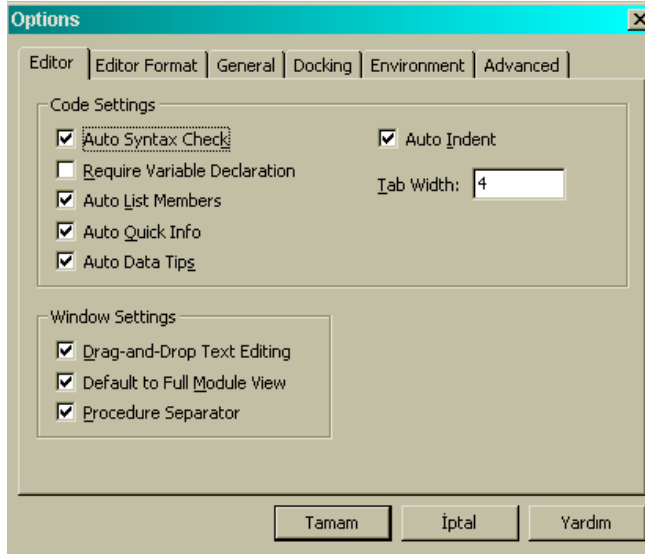
Kod Yazma Kuralları

Buraya kadar bir Visual Basic programının yapısını ve kodlanmasını temel olarak öğrendik. Bu konuda ise bir prosedürün kod editöründe yazılırken kurallar ve kullanılabilecek olanaklar yer almaktadır.

Otomatik Kod Tamamlama

Otomatik kod tamamlama program yazmayı sanki bir "boslukları doldurun" formu gibi kolaylaştırmıştır. Kod sözcükleri yazıldıkça özellikler, metotlar kendiliğinden bir liste olarak karşınıza gelir. Buradan istediğinizi seçebilir ve kodu yazmaya devam edebilirsiniz.

Otomatik tamamlama seçenekleri Options iletişim kutusunda yer alan Editör sekmesinden düzenlenir. Kod içinde kontrolün adı girildiğinde AutoListMembers özelliği bir liste ile olası özellikleri görüntüler. Özelliğin ilk harflerinden birkaç tane yazarak ya da listeden seçerek istenilen özellik seçilmiş olur. Seçilen eleman TAB tusuna basılarak program satırına alınır. Otomatik tamamlama özellikle kullanılan nesnenin özellikleri ve metotları bilinmiyorsa işe yarar.



Sekil 4.9 Visual Basic ortam düzenlemeleri

KOD EDITÖRÜ DÜZENLEME SEÇENEKLERİ

Kod Düzenlemeleri:

Auto Syntax Check	Bir kod satiri girildikten sonra otomatik olarak düzeltir.
Require Variable Declaration	Degiskenlerin tanımlanmasını zorlar. Bu seçenekte "Option Explicit" satiri modülün başına eklenir.
Auto List Members	Yazılan kod bileşenini tamamlar.
Auto Quick Info	Fonksiyonlar ve parametreleri hakkında
Auto Data Tips	Üzerinde bulunan değişkenin değeri listeler.
Auto Indent	Yazılan kod satırlarının üst satıra göre içeriden başlamasını sağlar.
Tab Width	Sekme (içeriden) başlama genişliğini ayarlar.

Pencere Düzenlemeleri:

Drag-and-Drop Text Editing	Kod penceresi, anlık ya da izleme pencerelerinde sürüklebirak işlemlerinin yapılmasını sağlar.
Default to Full Module View	Bir procedure ya da bütün procedure'ların görünmesini sağlayan görünüm düzeni.
Procedure Separator	Her bir procedure arasında çizgi olmasını sağlar.

Bu düzenleme Auto List Members özelliğinin seçilmemesiyle bu özellik ortadan kaldırılır. Ancak bu özelliğe yine kavusmak için CTRL +J tuslarına basılır. Auto Quick özelliği ise deyimlerin ve fonksiyonların sözdizimini gösterir. Bir Visual Basic deyiminin (örneğin MsgBox) adı yazıldığında sözdizimi aynı satırda hemen görülür.

Deyimin ya da fonksiyonların argümanları girildikçe bir sonraki kalın olarak görülür. Bu düzenleme Auto Quick Info özelliğinin seçilmemesiyle bu özellik ortadan kaldırılır. Ancak bu özelliğe yine kavusmak için CTRL +I tuslarına basılır.

Kod Penceresi Kısayolları

F5	Programı çalıştırma.
F7	Kod penceresini gösterir.
F2	Object browser'i gösterir.
CTRL+F	Bul
CTRL+H	Yer değiştir
F3	Bir sonrakini bul
SHIFT+F3	Bir öncekini bul
CTRL+ DOWN ARROW	Bir sonraki procedure
CTRL+UP ARROW	Bir önceki procedure
SHIFT+F2	Tanımlamayı göster
CTRL+PAGE DOWN	Bir alt ekrana atla

Microsoft Visual Basic 6.0

CTRL+PAGE UP	Bir yukari ekrana atla
CTRL+SHIFT+F2	Son konuma git
CTRL+HOME	Modülün basina git
CTRL+END	Modülün sonuna git
CTRL+RIGHT ARROW	Bir sözcük saga
CTRL+LEFT ARROW	Bir sözcük sola
END	Satirin sonuna
HOME	Satirin basina
CTRL+Z	Geri al
CTRL+Y	Bulunulan satin sil
CTRL+DELETE	Sözcüğü sil
TAB	Bir sekme içeri
SHIFT+TAB	Bir sekme disari
CTRL+SHIFT+F9	Bütün breakpoint'leri temizle
SHIFT+F10	Kisayol menüsünü listele

Menü Kisayollari

Print	CTRL+P
Undo	CTRL+Z
Paste	CTRL+V
Delete	DEL ya da DELETE
Find	CTRL+F
Find Next	F3

Replace	CTRL+H
Indent	TAB
Outdent	SHIFT+TAB
List Properties/Methods	CTRL+J
List Constants	CTRL+SHIFT+J
Quick Info	CTRL+I
Parameter Info	CTRL+SHIFT+I
Complete Word	CTRL+SPACEBAR
Definition	SHIFT+F2
Last Position	CTRL+SHIFT+F2
Object Browser	F2
Immediate Window	CTRL+G
Project Explorer	CTRL+R
Properties Window	F4
Step Into	F8
Step Over	SHIFT+F8
Run To Cursor	CTRL+F8
Quick Watch	SHIFT+F9
Toggle Breakpoint	F9
Clear All Breakpoints	CTRL+SHIFT+F9
Start	F5
Break	CTRL+BREAK
Shortcut menu	SHIFT+F10

Microsoft Visual Basic 6.0

New Project	CTRL+N
Open Project	CTRL+O
Save Form	CTRL+S
Save Form As	CTRL+A
Full Compile	CTRL+FS
Property Pages	SHIFT+F4
Add File	CTRL+D

Koda Açıklamalar Eklemek

Yazılan kod satırların hakkında (önce ya da sonra) bir açıklama yapmak iyi bir alışkanlıktır. Visual Basic'te açıklama yapmak için tek tirnak kullanılır.

' bu bir açıklamadır.

Dim A As Integer ' bu da bir açıklamadır.

Bu satırlar Visual Basic derleyicisi tarafından işletilmezler.

Sub bilgial()

' Bu yordamda ekrandan bilgi alınır.

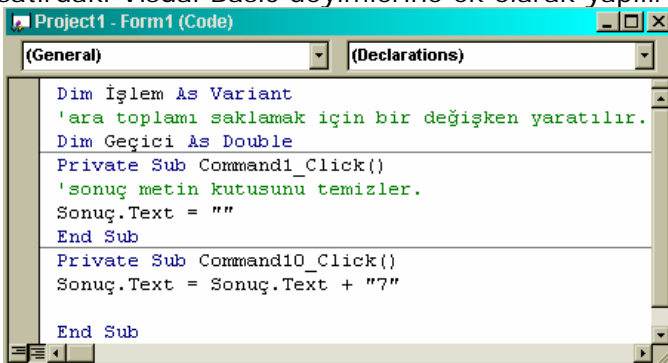
' bu asamada varsayılan değerler verilir.

' Text1.Text = "Ne haber" ' Mesaj

' bitir

End Sub

Yukarıdaki örneklerden de görüldüğü gibi açıklamalar tam bir satır olarak ya da bir satırdaki Visual Basic deyimlerine ek olarak yapılır.



Şekil 4.10 Kodlarla açıklamalara yer vermek

Uzun Bir Satiri Birçok Satıra Bölmek

Cobol ya da Fortran dilinde program yazarken (eski zamanlarda) bir program satirinin kaç karakter olacağı hatta satirdaki karakterlerin konumları bile önemliydi. Ancak şimdi böyle bir kısıtlama olmamasına rağmen uzun kod satirlarının takibi bakımından; belli bir yerinden bir alt satıra devam edilmesi için satir-devam karakteri olan alt tire (_) karakteri kullanılır. Böylece program satirlarının daha anlaşılabilir olması sağlanır.

```
Data1.RecordSource = _
    " SELECT * FROM Musteriler, Sipar, isler " _
    & " WHERE Musteriler.Kodu = Siparisler.Mkodu " _
    & " AND Siparisler.Bolge = ' ANADOLU ' "
```

Birçok Deyimin Bir Satir Olarak Birleştirilmesi

Genellikle bir Visual Basic deyimi bir satir olarak düzenlenir. Satir sonlarında herhangi bir sonlandırıcı karakter kullanılmaz. Ancak yine birden çok deyimi bir satir olarak düzenlemek istersiniz. Bu durumda satir sonu karakteri olarak iki nokta üst üste karakteri kullanılır.

Örnek: Birçok deyimin bir satir olarak birleştirilmesi

```
Text1 .Text = " Ne Haber " : Red = 255 : Text1.BackColor = Red
```

Visual Basic'te Adlandırma

Visual Basic ile program yazmada adlandırma (naming) işlemini çok yaparız. Bir değişkenin tanımlanması, bir Sub ya da Function'a ad vermek birer adlandırma işlemidir. Visual Basic'te adlandırma kuralları:

- Bir harf ile başlamalıdır.
- Nokta içermezler.
- 255 karakterden fazla olmazlar. Kontrol adları, form adları, class ve modül adları 40 karakteri geçmez.
- Bir Visual Basic deyimi adlandırmada kullanılmaz.

Bir Visual Basic deyimi örneğin MsgBox, bir ad olarak programın bir yerinde kullanılamaz. Çünkü bu deyimler Visual Basic'e özel sözcüklerdir. Bu sözcüklere kısıtlı sözcükler (restricted keywords) denir. Örneğin Len, Abs gibi sözcükler bir fonksiyon adı olduğu için bir değişkenin adlandırılmasında kullanılamazlar.

Aslında kısıtlı sözcüklerin programın adlandırılmasında kullanılması iyi bir davranış değildir. Çoğunlukla bir hataya neden olur.

`Msgbox.Visible = True` ' hata verir.

Ancak yine de istenirse kısıtlı sözcükler özel bir belirtimle kod içinde ad olarak kullanılabilir. Bir form ya da kontrole kısıtlı bir sözcüğün ad olarak verilebilmesi için köşeli parantez ya da hiyerarşik görünümde gösterilmesi gerekir.

`[Msgbox] . Visible = True` ' hata vermez.

Ya da;

`Musteri . Formu . Msgbox . Visible = True` ' hata vermez.

Bunun dışında köşeli parantezler diğer tip kütüphanelerden değişkenlerin kullanılmasında kısıtlı sözcüklerle karıştırılmamasını sağlarlar.

NOT: Visual Basic'te kodlama küçük harfle yapılır. Eğer değişkenlerin bir kısmı büyük harfle yazılırsa o zaman eslesen değişkenlerde büyük harfe çevrilir.

Deyimler

Deyimler (statements) programlama dilinin komutlarıdır. Bir işlemin yapılmasını sağlarlar. Deyimler nesnelerden bağımsız olarak yalnız başlarına yazılarak kullanılırlar. Örneğin Dim, Sub, MsgBox, Select Case, With, If .. Then birer Visual Basic deyimidir. Programda veri tanımlamalarının yani sıra belli işlemlerin yapılması için deyimler kullanılır. Deyimleri belli işlemleri yerine getirmesi bakımından şu şekilde sınıflandırabiliriz:

- Tanımlama deyimleri.
- Atama deyimleri.
- Aritmetik işlem deyimleri.
- Program denetimi deyimleri.
- Kullanıcı ile iletişim deyimleri.

Tanımlama deyimleri verilerin tanımlanmasını sağlar. Bunların başında Dim deyimisi gelir. Atama deyimleri ise değişkenlere değer verilmesini sağlar. Bir değişken tanımlandıktan sonra ona değer vermek gerektiğinde atama deyimisi kullanılır. Atama deyimisi bildiğimiz eşittir işaretidir.

```
Toplam = 2
Bakiye = 1
Ogrenci_sayisi = 100
Buyuk_Sayi = 9
```

Ayrıca değişkenler de birbirine atanabilir:

```
Toplam = Degerler_Toplami
Ortalama = Toplam / OgrSay
Adi Soyadi = "Ahmet Yilmaz"
Bosluk = InStr(AdiSoyadi," ")
Adi = Left(AdiSoyadi, Bosluk)
```

Yukarıdaki basit atama deyimleri dışında nesne ve özellikleri de atama deyimlerine konu olabilirler.

```
Ogrenci = Text1.Text
Text2.Text = "Ahmet Uzun"
```

Aritmetik İşlemler

Programlarda yaygın bir biçimde sayısal işlemler yaparız. Bu işlemler için aritmetik işlemler (arithmetic operators) kullanılır. Bir ortalama, bir ara toplam bir bakiyenin hesaplanması aritmetik işlemler gerektirir.

Toplama (Addition)	+
Çıkarma (Subtraction)	-
Çarpma (Multiplication)	*
Bölme (Division)	/
Tamsayı bölme (Integer division)	\
Mod (Modulus)	mod
Üs alma (Exponentiation)	^

Toplama ve çıkarma işlemi iki basit matematik işlemidir. Visual Basic programında çok sayıda sayıyı, ifadeyi birbiri ile toplayabilir ve çıkarabilirsiniz.

```
Toplam = Sayi1 + Sayi2
Fark = 5 - Eldekalın
Sonuc = Sayi1 - Sayi2
```

Toplama işleminde sayıların toplanma sırası önemli değildir. Ancak çıkarma işleminde bu sıra şu şekildedir. Önce ikinci sayı birinci sayıdan çıkarılır. Ardından kalandan üçüncü sayı çıkartılır. Böylece işlemler soldan sağa doğru gider.

```
Sonuc = 20 - 12 - 5
```

Microsoft Visual Basic 6.0

Yukarıdaki ifade de önce 20'den 12 çıkartılır. 8 kalır. Ardından 8'den 5 çıkartılır. Ve sonuç olarak 3 kalır. Uzun ya da karmaşık toplama ve çıkarma işlemleri parça parça yapılabileceği gibi parantezler içinde de yapılabilir.

$$\text{Sonuc} = 20 - (12 - 5)$$

$$\text{Sonuc} = (20 - 12) - 5$$

Yukarıdaki iki ifadenin de sonucu farklıdır. Çünkü önce parantezlerin içi hesaplanır. Çarpma işlemi de toplama işlemi gibi düzdür. Sayılar sırasıyla çarpılır ve toplam elde edilir.

$$\text{Sonuc} = \text{Sayi1} * \text{Sayi2}$$

$$\text{Sonuc} = \text{Sayi1} * \text{Sayi2} * \text{Sayi3}$$

Bölme işlemi değişik biçimlerde yapılabilir. Normal bölme işlemi (floating-point division olarak bilinir) bir sayının diğerine bölünmesidir. Sonuç genellikle noktalı bir sayıdır.

$$\text{Sonuc} = \text{Sayi1} / \text{Sayi2}$$

$$\text{Sonuc} = 10 / 6$$

Sonuç 1,66 çıkar. Tamsayı bölme işleminde ise bölüm sonucunun tamsayı kısmı döndürülür. Bu işlem \ isareti ile yapılır.

$$\text{Sonuc} = \text{Sayi1} \backslash \text{Sayi2}$$

$$\text{Sonuc} = 4 \backslash 3$$

Sonuç 1 olarak çıkar. Mod işlemi ise bölme işlemindeki kalanı verir.

$$\text{Sonuç} = \text{Sayi1} \text{ Mod } \text{Sayi2}$$

Üs işlemi kuvvet olarak da bilinir. Örneğin 2^3 ikinin üçüncü kuvvetini yani $2 \times 2 \times 2$ ya da 8 değerini verir. Üs işlemleri genellikle bilimsel ve mühendislik işlemlerinde kullanılır. Üs alma işlemi $^$ dir.

$$\text{Sonuç} = \text{Sayi1} ^ 3$$

Değişik üs kullanımları:

$$3 ^ 2 = 9 \quad \text{Karesi}$$

$$9 ^ {0.5} = 3 \quad \text{Karekökü}$$

$$2 ^ {-2} = 0.25 \quad \text{Negatif üs ile fraction elde edilir.}$$

İsleçlerin (Operatörlerin) Öncelikleri

Bu bölümde aritmetik işlemlerin uygulanmasından ve elde edilen sonuçları açıkladık. Aritmetik ifadelerde parantez kullanımı önceliği belirleyen asıl faktördür. Ancak parantez kullanılmıyorsa o zaman operatörlerin öncelikleri işlemlerin sonuçlarını etkiler. Öncelik sırası:

1. Üs alma (^)
2. Negatif (-)
3. Çarpma ve bölme (*; /)
4. Tamsayı bölme (\)
5. Mod işlemi (Mod)
6. Toplama ve çıkarma (+, -)

Bir ifade içinde bileşenler soldan sağa doğru işlenirler. Bütün alt gruplar hesaplandıktan sonra kalan soldan sağa doğru değerlendirilir. Örneğin iki test sonucunun ortalamasını almak için aşağıdaki deyim yazmak mümkündür:

Ortalama = Not1 + Not2 / 2

Ancak yukarıdaki işlemin sonucu yanlıştır. Bunun yerine şu ifade kullanılmalıdır:

Ortalama = (Not1 + Not2) / 2

Metotlar

Deyimlerin yani sıra Visual Basic'te programa hayat vermek için metotlar (methods) kullanılır. Metotlar nesnelere belli işlemleri yaptırmayı sağlarlar. Örneğin araba bir nesnedir. Onun hareketleri metotlarını oluşturur. İleri/geri hareket etmek arabanın metodudur. Bir arabanın davranışları ise şunlardır:

Araba . İleri
Araba . Geri
Araba . Dur

Dikkat edersek arabanın eylemleri sadece bir komut gibi çalışır. İleri deyim arabanın ileri hareketini gösterir. Bu deyime metot (method) denir.

Arabanın bir günlük gezisi:

```
Sub Araba_Basla()  
    Araba.İleri  
    Araba.Korna = "Daat"
```

Microsoft Visual Basic 6.0

```
Araba.Durumu = "Kirli"  
Araba.vites = "2"  
Araba.Dur  
End Sub
```

Yukarıdaki örnekte araba hareket eder, birtakim özellikler kazanır ve sonra durur. İşte arabaya ne yaptırılacaksa bir metod ya da bir özellikte yaptırılır. İşte bir sipariş programı yazacaksanız; form üzerindeki bilgi alanlarından yararlanırken veri girişi, hesaplama, kayıt vb bütün işlemleri birer olay olarak düşünüp, olayın gerçekleşmesinin ardından birtakim işlemler (metotlar) yapılır ve birtakim değerler kazanılır.

Özellikler

Özellikler (properties), nesnelerin büyüklüğü, rengi, adı, yazı tipi vb niteliklerini içeren bilgilerdir. Örneğin bir arabanın özellikleri, onun rengi vb sahip olduğu değerlerdir.

```
Araba . Renk = Red  
Araba . Fiyati = 10000000  
Araba . Yeni = True  
Araba . Durumu = Temiz
```

Araba nesnesinin renk özelliğinin değeri Red'mis. isterseniz başka renkleri de verebilirsiniz.

Arabanın bir günlük gezisi:

```
Sub Araba_Basla()  
Araba.Ileri  
Araba.Korna = "Daat"  
Araba.Durumu = "Kirli"  
Araba.vites = "2"  
Araba.Dur  
End Sub
```

Yukarıdaki örnekte araba hareket eder, birtakim özellikler kazanır ve sonra durur. İşte arabaya ne yaptırılacaksa bir metod ya da bir özellikte yaptırılır.

```
Set Picture1.Picture = LoadPicture ( " C : \WINDOWS\CLOUDS.BMP " )
```

Yukarıdaki deyim ile bir resim kutusuna (Picture Box), CLOUDS.BMP dosyası yüklenir (gösterilir). Aşağıdaki deyimle ise bu resim dosyası boşaltılır.

```
Set Picture1.Picture = LoadPicture ( "" )
```

Yukarıdaki örneklerde ise Picture özelliği kullanılmıştır.

With Deyimi

Visual Basic'te özellikle özelliklerin (properties) düzenlenmesinde kullanılan bir deyim vardır. With deyimi bir nesnenin ya da bir kullanıcı tanımlı değişkenin bir dizi deyim için bir kez belirtilmesini sağlar. With deyimi sayesinde prosedürler daha hızlı çalışırlar ve kod yazma zamanı azaltılır ve etkinleştirilir. Özellikle bir nesneye çok sayıda özellikler vereceksek With deyimi idealdir. Aşağıdaki örnek ile normal bir özellik atama deyimleri yerine daha kısa bir şekilde With deyimi kullanılmaktadır:

Uzun şekli:

```
Private Sub Command1_Click()  
    Combo1.AddItem "Çorbalar"  
    Combo1.AddItem "Tatlılar"  
    Combo1.AddItem "Yemekler"  
    Combo1.AddItem "İçkiler"  
    Combo1.FontBold = True  
    Combo1.Fontname = "Aria"  
End Sub
```

With deyimi ile olanı:

```
Private Sub Command1_Click()  
    With Combo1  
        .AddItem "Çorbalar"  
        .AddItem "Tatlılar"  
        .AddItem "Yemekler"  
        .AddItem "İçkiler"  
        .FontBold = True  
        .Fontname = "Aria"  
    End With  
End Sub
```

Diğer bir örnek ile bir metin kutusunun özellikleri verilmektedir:

```
Sub Bicimle()  
    With Text1  
        .Text = 30  
        .Font.Bold = True  
        .Interior.Color = RGB(255, 255, 0)  
    End With
```

```
End Sub
```

```
Sub Bicimle2()  
  With Text1  
    .Text = "Merhaba"  
    With .Font  
      .Name = "Arial"  
      .Bold = True  
      .Size = 8  
    End With  
  End With  
End Sub
```

Veritabanı uygulamalarında da With deyimi ile birçok işlem daha kolay yapılabilir.

Olaylar

Visual Basic programlarının çalışmasında olaylar (events) çok önemlidir. Olaylar genellikle bir işlemin başlatılması (tetiklenmesi) için kullanılırlar. Yaygın olarak kullanılan olay bir kontrole (örneğin bir komut düğmesine) tıklamaktır (click). Burada olay "click" tir.

Olay yordamları bir olayın (event) oluşmasına tepki olarak çalışırlar. Visual Basic nesneleri (formlar, metin kutuları, düğmeler, vb) belli olayları tanırlar. Böylece olay oluştuğunda; o olay için hazırlanmış olay yordamı çalışır.

Bir komut düğmesinin olayları:

Click	Tıklama
GotFocus	Odaklanma, üzerine gelme.
KeyDown	Bir tusa basmak.
MouseDown	Farenin bir tusuna basmak.

Olay yordamları, bir nesnenin alt tire ile olaylara bağlı olarak geliştirilir. Bu adlar Visual Basic kod editörü tarafından otomatik olarak ya da manuel olarak oluşturulur.

```
Nesne_Olay  
Command1_Click
```

Visual Basic olaylarını ikiye ayırmak mümkündür: Kullanıcı temelli olaylar, Nesne temelli olaylar. Komut düğmesine tıklamak (click) kullanıcı temelli bir olaydır. Ancak bir form_load olayı formun yüklenmesiyle otomatik olarak oluşur. Bu olay nesne temellidir. Aşağıdaki örnekte Text1 kontrolüne

odaklandığında (odaklanma olayı olustugunda) textbox'in zemin rengi düzenlenir.

```
Sub Tex1.GotFocus ()  
    Text1.BackColor = RGB(255,0,0)    j  
End Sub
```

```
Sub Tex1.LostFocus ()  
    Text1.BackColor = RGB(0,0,0)  
End Sub
```

Veri Elemanlarının Tanımlamaları

Bir programda geçici olarak verilerin temsili ve degerlerin saklanması için degiskenlere gereksinim duyulur.Örneğin çok sayıda degeri birbiriyle karsilastirmak ya da üzerlerinde çeşitli hesaplamalar yapmak için ekrandan alınan verileri; A, B, C gibi degiskenler olarak temsil etmek gibi. Iste bu nedenle birçok programlama dilinde olduğu gibi Visual Basic'te de geçici degerlerin saklanması için degiskenler (variables) kullanılır. Degiskenlerin mutlaka bir adi ve tipi vardır.

- Müsteri adini temsil için ADI degiskeni.
- Müsteri gurubunu temsil için GRUBU degiskeni.
- Alis fiyatı için AFIYAT degiskeni gibi.

Degiskenlerin yani sıra program içinde bir takım sabit degerlerinde temsili de gerekir. Örneğin, Pi sayısı 3. 14, KDV oranı 1.18 gibi.

Degiskenler

Degiskenler program içinde yer alan geçici veri alanlarını (diger bir deyişle bellekteki yerlerin adlarını) temsil ederler. Örneğin, programın çalışması sırasında iki degerin çarpımına gereksinim duydunuz. Daha sonra bu ara toplamı diger bir deger ile karsilastiracaksınız ve veritabanına yazacaksınız diyelim.Iste bu süreç içinde birçok degiskene gereksinim duyulur.

Degiskenlerin kullanimında; degiskenler tanımlanırlar, degerler alırlar (deger ataması). Ardından procedure'in ya da formun kapanmasıyla bellekten silinirler. Bu nedenle degiskenler için geçici (temporary) sifati kullanılır.

Degiskenlerin Tanımlanması

Bir degisken kullanılmadan önce tanımlanır. Bu tanımlama genellikle Dim deyimini ile yordamin basında yapılır:

Yapisi:

Dim Degisken [As Tip]

Dim deyimi tanımlama deyimidir. Genellikle modülün ya da procedure'in başında ver alır.

Degisken ise bilginin adini gösterir. Örneğin "Toplam" bir degisken adidir.

Tip ise degiskenin tipini gösterir. Bir degisken çerdigi veri bakımından farklı özelliklere sahip olabilir. Tamsayı ya da gerçek sayı gibi.

Bir degiskenin özellikleri:

- Bir harf ile baslar.
- En çok 255 karakterden oluşur.
- Nokta içermez.
- Kapsama alanı içinde tek olmalıdır.

Dim A As Integer

A=5

A=A+3

Bir degiskeni tanımlamak için değişik yollar da vardır:

1. Degiskeni bir formun, standart ya da class modülünün Declarations bölümünde tanımlamak. Bu tanımlama ile degisken formun ya da modülün her yerinden erişilebilir.

2. Bir degiskeni Public sözcüğü ile tanımlamak onu tüm uygulama tarafından kullanılabilir yapar.

3. Yerel bir degiskenin Static sözcüğü ile tanımlanması onun değerinin (içerisinin) procedure'in sonlanmasına rağmen sürmesine neden olur.

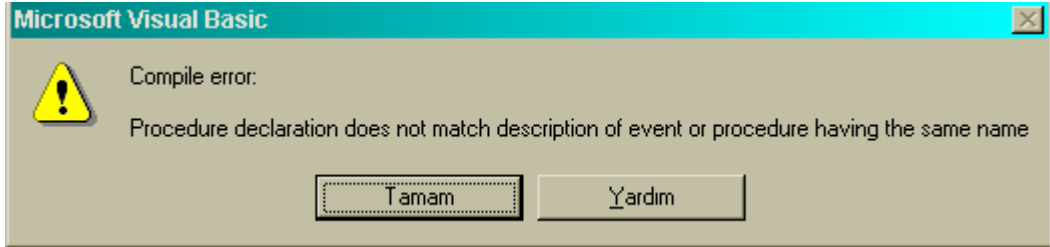
Genel olarak bir Visual Basic programı içinde kullanılacak olan degiskenlerin tanımlanması gerekir. Ancak eğer degisken (daha önce) tanımlanmadan kullanılırsa Visual Basic ona varsayım olarak Variant veri tipini yaratır. Variant bir degisken herhangi bir tip veriyi içerebilir.

Pratikte çok yararlı gibi görünen varsayım tanımlamaların görünmeyen yan etkileri olabilir. Bir Visual Basic programında Variant tanımlanmış varsayım tipli degiskenleri kullanmanın kötü tarafları vardır. Bunlar:

- Daha fazla bellek kullanımına neden olur.
- Variant veri tipi bazı veri işlem fonksiyonları için geçersiz olabilir.
- Değer atamalarda karışıklıklar olabilir.

Değişken Tanımlamada Hatalar

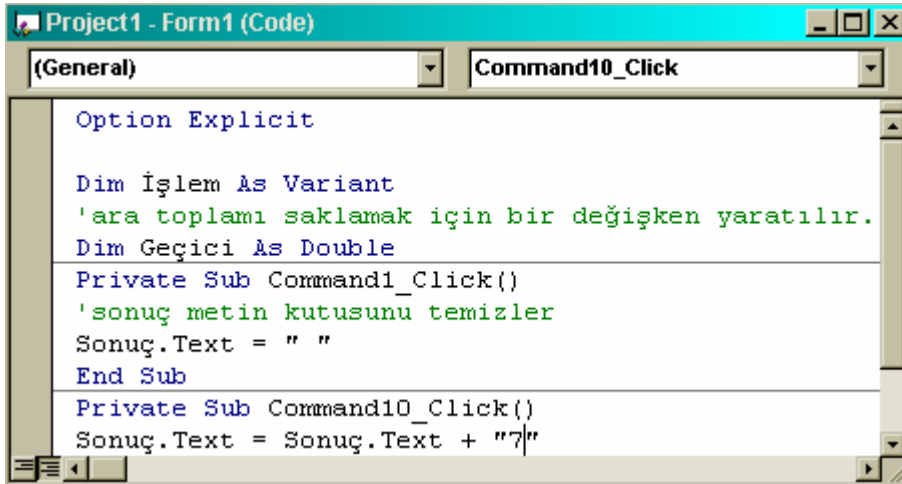
Bir değişken adı hatalı yazılmamalıdır. Bir değişken adı bir procedure adı ile aynı olmamalıdır.



Sekil 4.11 Bir değişkenin adı program içindeki bir nesne ile aynı olduğunda bu mesaj ekrana gelir.

Açık Tanımlama

Açık tanımlama (explicit declaration) program içinde (procedure'lar) kullanılan bütün değişkenlerin ad ve tip olarak tanımlanmasıdır. Tanımlama genellikle procedure başında ya da modül başında yapılır.



Sekil 4.12 Açık tanımlamaya zorlama

Açık Tanımlamaya Zorlama

Daha önce varsayım tanımlamalardan söz ettik. Bu tür tanımlamalara kapalı (implicit) tanımlama denir. Bununla birlikte değişkenlerin kapalı olarak tanımlanarak kullanılmasının çeşitli problemlere yol açabileceğini de belirttik. İşte bu nedenle değişkenlerin açık olarak tanımlanmaya zorlanması sağlanabilir. Bu tanımlamanın (zorlamanın) ardından kod içinde tanımlanmamış bir değişkenin kullanımını Visual Basic derleyici "tanımsız değişken" olarak yakalayacak ve size bildirecektir.

Değişkenleri açık olarak tanımlamak için: Bir form, class ya da standard modül'ün Declarations kesiminde şu deyimlere yer verilir:

Options Explicit

Yada;

Tools menüsünden Options komutu seçilir. Ardından Editör sekmesinden Require Variable Declaration seçeneği işaretlenir. Bu düzenlemenin ardından Option Explicit deyiimi proje içindeki bütün modüllere yerleştirilir.

Option Explicit deyimini düzenledikten sonra bir değişkeni tanımlamadan kullanmaya kalkarsanız Variable not defined hata mesajı alırsınız.

Kapalı Tanımlamalar

Bir değişkenin kullanılmadan önce procedure ya da modül başında tanımlanması genel olarak yapılması gereken şeydir. Ancak Visual Basic kapalı tanımlama olarak adlandırabileceğimiz bir yöntemle; tanımlanmadan kullanılan değişkenlere varsayım olarak Variant tipinde bir değişkenmiş gibi kullanılmasını sağlar. Örneğin procedure'in bir yerinde şu ifadeyi kullanmak daha önce bu değişkenlerin tanımlanmasını gerektirmez:

AraToplam = BirinciToplam + İkinciToplam

Küçük programlama işlemleri için değişken tanımlama zorunluluğunu ortadan kaldıran kapalı tanımlama (implicit declaration) büyük programlama işlerinde değişken adlarının yanlış yazılmasından ötürü problemlerin yaşanmasına neden olur. Bu nedenle açık tanımlama ve açık tanımlamaya zorlama işlemlerinin kesinlikle yapılması gerekir.

NOT: Özellikle çok sayıda değişkeni kullandığınız bir programda kapalı tanımlamalara gitmeyin. İki değişkenin birisi "KüçükDeger" diğer "KucukDeger" olarak yazılmamışsa ve sizde bu değişkeni farklı şekillerde yazarak kullanıyorsanız, programda hiçbir hata verilmeyecektir. Çünkü siz bir değişken

kullandiginizi sanirken, Visual Basic iki degisken kullanacaktır ve hesaplamalarda yanlisliklar olusacaktır. Bu durumda ciddi programlarda kapali tanimlamalara izin vermeyin!

"KüçükDeger" = "KucukDeger" ?
"Ödeme" = "Odeme" ?

Bir degiskenin kullanilmadan önce tanimlanmasi gerektigini birçok kez belirttik, ancak kapali bir tanimlama ile de degiskenlere belli tipler verilebilir. Bu islem için degisken tipleri belli karakterler kullanilir:

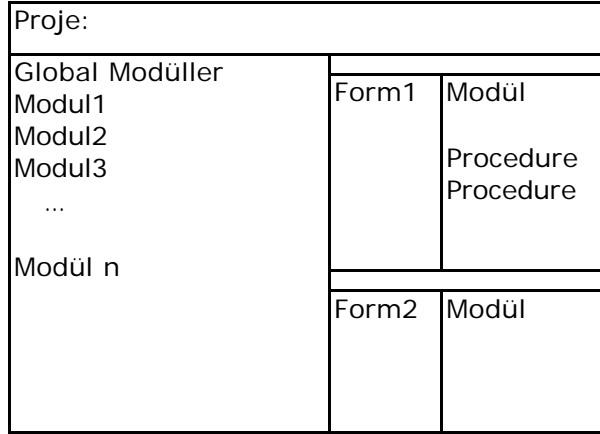
Saklanan degiskenlerin tipini belirten özel karakterler

Degisken tipi	Karakter
Integer	%
long	&
Single	!
Double	#
Currency	@
String	\$
Byte	Yok
Boolean	Yok
Date	Yok
Object	Yok
Variant	Yok

Sayi1% = 0
OrtalamaDeger% = 1
HesaplananOrt! = 10.1
Mesaj\$ = "Lütfen disketi A sürücüsüne takin"

Degiskenler Nerede Tanimlanacak?

Bir Visual Basic programinda çok sayida degisken kullanilir. Bu degiskenler çok sayida modülde ya da procedure'da tanimlanabilir. Peki hangi degisken nerede tanimlanacak? Bir degiskenin modül içinde mi yoksa procedure içinde mi tanimlanacagi kapsama alanı kurallarına göre kararlaltirilir.



Sekil 4.13 Modüllerin Kapsam Alanı

Kapsama Alanı

Nasıl telefonlarınız bir vericinin etki alanının dışına çıktığınızda hata sinyalleri veriyor, iste bir Visual Basic değişkeni de tanımlandıktan sonra kullanılırken belli bir kapsama alanı ile sınırlıdır. Sınır aşıldığında değişkenin etkisi kalmaz.

Bir procedure içinde tanımlanan degiskene local (yerel) degisken denir. Bu degisken bu procedure aktif olduğu zaman aktif olur. Buradan degiskenin procedure'in başlaması ve bitmesi arasında yaşadığını söyleyebiliriz. Böylece ortaya "kapsama alanı" konusu çıkar.

Kapsama alanı procedure değişkenin tanımlandığı yere bağlı olarak, procedure, form ve uygulama olmak üzere üç ayrı genişlikte gerçekleşir. Bildiğiniz gibi bir Visual Basic kodu mutlaka procedure olarak yazılıyordu. Ancak procedure'lar da bir modülün içindeydi; modüller de form, class ya da standart olabiliyordu. İste bir değişkeni bir modülün tanımlamalar kısmında tanımlamak o değişkenin o modül içinde yer alan bütün procedure'larda kullanılmasını (geçerli olmasını) sağlarken, bir procedure içinde tanımlanan değişken (yerel değişken olarak adlandırılır) sadece o procedure içinden kullanılabilir. Yani o procedure içinde geçerlidir. Baska bir procedure içinde diğer bir procedure içinde tanımlanan degiskene bir atama ya da onun değerine ulaşmak mümkün olmaz.

Yerel Değişkenler

Yerel değişkenler procedure-düzeyi değişkenlerdir. Bir procedure içinde tanımlanırlar, kullanılırlar. Procedure'in başlamasıyla bu değişkenler tanımlanır. Yerel değişkenler özellikle hesaplanan bir takım geçici değerlerin

kullanilmasini saglar. Genellikle Temp ya da Geçici gibi ekleri de bu yüzden alirlar.

TempToplam
GecTop
GecDeg1
AraToplam

Yerel degiskenler Dim ya da Static deyimi ile tanimlanirlar:

Dim AraToplam As Integer
Dim Mesaj As Variant

Geçici olarak tanimlanan yerel degiskenler procedure içinde yapılan ve kullanılan hesaplamalar için deildir. Eger bir degiskene uygulama boyunca diger zamanlarda da erisilecekse (degisken kullanılacaksa) o zaman Dim ile kullanilmasi daha iyi olur.

Yerel Degiskenlerin Static Olarak Tanimlanmasi

Static deyimi ile tanimlanan yerel degiskenler uygulama çalistigi sürece bellekte kalirlar ve istenildigi zaman kullanilirlar. Dim deyimi ile tanimlanan yerel degiskenler için sadece procedure çalistigi süre içinde kullanilirlar.

Static Degisken Tanimlama:

Static Degisken (As tip]
Static deyimi degiskenin bellek alaninin kalici olmasini saglar.
Degisken ise bilginin adini gösterir.

Static AraToplam As Integer
Static Mesaj As Variant

Bir Modül İçinde Kullanilan Degiskenler

Bir modül düzeyi degiskeni varsayim olarak modül içindeki bütün procedure'lerde geçerlidir. Modül-düzey degiskenler Private deyimi ile tanimlanirlar.

Private Aratoplam As Integer

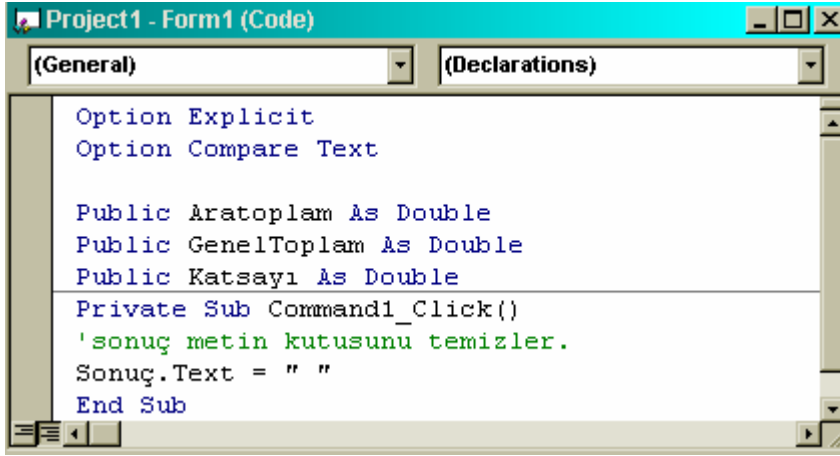
Modül düzeyinde Private ile Dim arasında bir fark yoktur. Ancak Private deyimi özellikle modül düzeyini vurgular.

Bütün Modüller İçinde Kullanilan Degiskenler

Bir modül düzeyi değişkenini bütün diğer modüllerde de geçerli kılmak için Public deyimi kullanılır. Bir public değişkenin değeri uygulama içindeki bütün modüllerde geçerlidir. Bu değişkenler bütün modüllerde kullanılabilir ya da değer atanabilir. Public değişkenler de diğer modül düzeyi değişkenler gibi modülün üst kısmındaki Declarations kesiminde tanımlanırlar.

Public Aratoplam As Integer

NOT: Bir Public değişken bir procedure içinde tanımlanamaz. Sadece modülün declarations kesiminde tanımlanır.



Sekil 4.14 Public değişkenler

Bütün Yerel Değişkenlerin Static Olarak Tanımlanması

Bir yordamdaki bütün yerel değişkenlerin Static yapılması için Static deyimi fonksiyonun başına konur.

Static Function Toplambul(miktar)

Bu durumda bütün yerel değişkenler; Static, Dim, Private gibi tanımlama deyimlerine bakılmaksızın Static olurlar. Böylece bu değişkenler uygulamanın sonuna kadar her zaman ve her yerden kullanılabilirler.

Aynı Değişken Adının Birçok Modülde Kullanılması

Farklı modüllerde bulunan Public değişkenler aynı adı kullanıyorlarsa bunların modül adları ile birlikte ayrı ayrı kullanmak mümkündür.

```
Birinci modül: Form1
Public Toplam As Double
```

```
Ikinci modül: Modül1
Public Toplam As Double
```

```
Modül1.Toplam    modül1 içindeki Toplam degiskeni.
Form1.Toplam     form1 içindeki Toplam degiskeni.
```

Örnek: Ayni Degisken Adinin Birçok Modülde Kullanilmasi.

Ayni degisken adinin birden çok modülde kullanilmasi için iki standart modül kullanalım. Bunun yanı sıra bir form yaratıp üzerinde üç tane komut düğmesi koyalım. Birinci modülde bir degisken tanımlayalım.

```
Public Deger As Integer
```

Ardından bir Test procedure'i ile Deg1 degiskenine bir deger verelim:

```
Sub Test()
    Deger = 1
End Sub
```

Ikinci degisken de yine Deger adinda olacak ancak ikinci modülde duracak:

```
Public Deger As Integer    ' Module2'deki deger
Sub Test()
    Deger = 2
End Sub
```

Üçüncü Deger degiskenini de Form modülü içinde tanımlayalım ve bir Test procedure'i ile deger atayalım:

```
Public Deger As Integer
Sub Test ()
    Deger = 3
End Sub
```

Ardından üç komut düğmesine farklı modüllerde bulunan degiskenlerin degerlerini gösteren deyimleri yazalım:

```
Private Sub Command1_Click ()
    Module1.Test
    'Modül1 içindeki Test procedure'ini çalıştırır.
    MsgBox Module1.Deger
```

Microsoft Visual Basic 6.0

```
'Module1' in içindeki Degeri gösterir.  
End Sub  
  
Private Sub Command2_Click()  
    Module2.Test  
    MsgBox Module2.Deger  
End Sub  
  
Private Sub Command3_Click()  
    Test ' Form1 içindeki Test procedure.  
    MsgBox Deger  
End Sub
```

NOT: Üçüncü durumda Test ve Deger degerlerinin modül ile birlikte gösterilmedigine dikkat edin. Eger birçok degisken ayni adi paylasiyorsa Visual Basic onun en yerel olanini kullanir. Buna digerlerini "gölgeleme" denir.

Degisken ve Procedure Adlariinin Ayni Olmasi

Private ya da Public olarak tanimlanan modül düzeyi degiskenler procedure adlariyla çalistiklarinda soruna yol açarlar. Modül içindeki bir degisken procedure adi ile ayni adi alamaz. Ancak Public procedure'lar, Type degiskenler ve diger modüller içindeki degiskenlerle ayni adi alamazlar.

Public ve Local Degiskenler

Ayni degisken adlariinin kullanimina farkli kapsamlarda da karsilasilir. Örneğin Aratoplama adli bir Public degisken tanimlanir. Ardindan da bir procedure içinde yine Aratoplama adli bir yerel degisken tanimlanir.

Bu tanimlamanin ardindan procedure içinde Aratoplama adli degiskene erismek istendiginde yerel degiskene erisilir. Procedure disindan erisim ise Public degiskene olacaktir. Ayrica modül düzeyindeki degiskene procedure içinden erisilmek istenirse modülün adi ile birlikte kullanılabilir.

```
Public Aratoplama As Integer  
  
Sub Test()  
    Dim Aratoplama As Integer  
    Aratoplama = 2  
    ' Aratoplama degiskenine 2 degeri verilir.  
    MsgBox Form1.Aratoplama  
    ' Form1.Aratoplama degeri 1.  
End Sub
```

```
Private Sub Form_Load ()
    Aratoplam = 1 ' Form1.Aratoplama 1 degeri ver.
End Sub
```

```
Private Sub Command1_Click()
    Test
End Sub
```

:

Yasam Süresi

Kapsama alanina benzer ama farkli bir kavram da yasam süresidir (lifetime). Yasam süresi bir degiskenin degerini sürdürdüğü (geçerli oldugu) süreyi belirtir. Public ye modül düzeyindeki degiskenlerin degeri uygulama boyunca korunur. Yerel degiskenlerin degeri ise procedure'in çalışması ile baslar ve procedure'in sona ermesiyle sona erer. Ancak yerel degiskenler Static deyimi ile tanımlanarak uygulama boyunca korunurlar.

Static UzunYasayanDegisken As Integer

Static Degisken Tanımlama:

Static Degisken [As tip]

Static deyimi degiskenin bellek alaninin kalici olmasini saglar.

Degisken ise bilginin adini gösterir. Örneğin "Toplam" bir degisken adidir.

Tip ise degiskenin tipini gösterir. Bir degisken içerdigi veri bakımından farkli özelliklere sahip olabilir. Tamsayı ya da gerçek sayı gibi.

Örneğin aşağıdaki yordam bir önceki toplam degere yeni toplam degeri eklenir.

```
Function ToplamBul (Miktar)
    Static Satadet
    Satadet = Satadet + miktar
    ToplamBul = Satadet
End Function
```

Eğer Satadet, Static yerine Dim olarak tanımlansaydı toplam degeri eski toplam degerine eklenmezdi ve ToplamBul fonksiyonu çağırılan degeri geri döndürürdü. Ancak aynı sonuç Satadet degiskeninin modül düzeyinde tanımlanmasıyla da olabilirdi.

Sabitler

Degiskenlerin aksine sabitler (constant), sahip olduklari degerler ile kullandigimiz bilgi tanimlamalaridir.Özellikle belli (bilinen) degerlerin kullanimini saglar. Kodlamayi ve kodun okunabilirliğini kolaylastirir.

Bir sabit su biçimde tanimlanir:

[Public | Private] Const adi [As tip] = Ifade ya da deger

Örnek:

```
Const Pi = 3.14159265358979
Const Katsayi = 2.5
Public Const MaxOgrSay = 999
Const AcilisTar = # 1/1/98 #
Const ElemanAdi = "Onur Türkyilmaz"
```

Sabitlerin degeri bir sayi ya da karakter olabilecegi gibi bir ifade de olabilir.

```
Const Oran = AitOran * 2
```

Kullanici tanimli sabitlerin kapsamı da yine degisken kapsamı kurallarına benzer:

- Bir procedure içinde tanımlanan sabit o procedure içinde kullanılır.
- Bir sabitin bütün procedure'lar için geçerli olması için modülün Declarations kesiminde tanımlanması gerekir.
- Bir sabitin uygulama boyunca geçerli olması için sabit bir Standart modülün Declarations kesiminde tanımlanır. Ayrıca Const deyiminden önce Public deyimi kullanılır.

Sabitler aynı zamanda degerleri bilgisayarın ana belleğinde saklamanın diğer bir yoludur. Bir sabit tanımlandıktan sonra degeri degistirilemez. Eger program yolu ile degistirilirse o zaman Visual Basic hata verir.

Sabit kullanmanın yararları:

- Uzunca tanımlanan verilerin kolayca kullanılmasını sağlar.
- Belli oranların, degerlerin kolayca kullanılmasını sağlar.
- Programın degisik versiyonlarını tanıtan sabitler sayesinde programın kodlarından daha kolay yararlanılır.

Sabit Tanımlamalarında Çözülemez İfadeler

Bir sabit tanımlanırken diğer bir sabitin ona göre tanımlanması hataya yol açar.

Örnek:

Birinci modülde:

```
Public Const Deger1 = Deger2 + 7
```

İkinci modülde:

```
Public Const Deger2 = Deger1 / 2
```

Yukarıdaki örnekte iki sabit birbirlerini içeren bir tanımlamaya sahiptir. Bu durumda bir "circular" hata oluşur. Bu nedenle sabitler genellikle bir modül olarak hep bir arada tanımlanmalıdırlar.

Option Deyimleri

Options deyimleri form ve modüllerin başında kullanılır. Options deyimleri değişkenlerin ve dizilerin tanımlamalarını (kullanımını) etkiler. Bir form, class ya da standart modülün Declarations kesiminde, modül içinde kullanılacak değişkenlerin daha önceden tanımlanması zorunluluğunu getirmek için şu deyimlere yer verilir:

Options Explicit

Ya da; Tools menüsünden Options komutu seçilir. Ardından Editör sekmesinden Require Variable Declaration seçeneği işaretlenir. Bu düzenlemenin ardından Option Explicit deyimini proje içindeki bütün modüllere yerleştirilir.

Option Explicit deyimini düzenledikten sonra bir değişkeni tanımlamadan kullanmaya kalkarsanız Variable not defined hata mesajı alırsınız.

Option Compare Deyiminin Kullanılması

Option Compare deyimini string(karakter bilgi) işlemlerinde yapılacak olan karşılaştırmayı düzenler.

Yapısı: Option Compare { Binary | Text | Database }

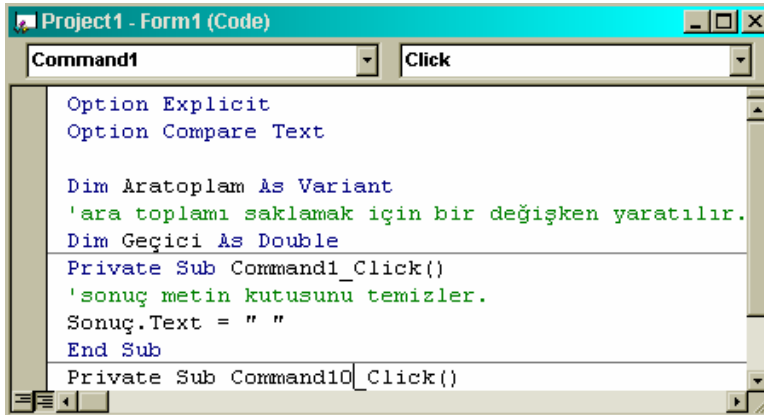
Option Compare deyimi string bilgilerin nasıl (Binary, Text, ya da Database) karşılaştırılacağını belirtir. Varsayılan karşılaştırma yöntemi Binary'dir. Option Compare Binary karşılaştırma karakterlerin ikili değerlerine göre yapılır:

A < B < E < Z < a < b < e < z < Å < Ê < Ø < à < ê < ø

Option Compare Text karşılaştırma yönteminde ise büyük küçük harf duyarlı olmayan (case-insensitive) sıralama uygulanır. Bu yöntemde sistemin ayarları kullanılır.

(A=a) < (Å= à) < (B=b) < (E=e) < (Ê= ê) < (Z=z) < (Ø= ø)

Option Compare Database yöntemi ise sadece Microsoft Access'de uygulanır. Bu yöntemde veritabanı kimlik bilgisine göre sıralama yapılır.



Sekil 4.15 Options deyimleri

Option Base Deyimi

Option Base deyimi ise isteğe bağlı olarak kullanılabilir. Deyimin amacı dizi tanımlamalarında dizinin alt sınırını değiştirmektir. Normalde bir dizi sıfırdan başlayarak tanımlanır. Ancak Option Base 1 deyimi ile 30 ile 40 arası dizinleri (indeksi) arası bir dizi yaratılabilir:

Modül tanımlama kesimi:

```
Option Base 1
Dim Dizil (3 To 45) As Integer
Dim Aylar (1 To 12) As String
```

Daha fazla bilgi için bakınız: Bu bölümde yer alan: "Diziler".

Veri Tipleri

Bir değişken tanımlanırken değişkenin adı ve tipi (type) belirtilir. Genellikle varsayım tip olan Variant bir veri kullanılmayacağı zaman önce değişken tanımlanır ve tipi belirtilir. Değişkenin tipi onun içereceği bilgi türünü ifade eder.

Veri Adı	Tipi
Adi Soyadi	Alfabetik
Ücreti	Sayısal
Evlilik Durumu	Evet/Hayır

Visual Basic ile tanımlanan değişken tipleri şunlardır:

Tip	Bellek alanı	Değer aralığı
Integer	2 bayt	-32,768 to +32,767
Long	4 bayt	(yaklaşık) +/- 2 milyar
Single	4 bayt	+/- 1E-45 to 3E38
Double	8 bayt	+/-5E-324ta 1.8E308
Currency	8 bayt	+/- 9E14
String	1 karakter bir bayt	Sabit uzunluklu veriler için 65,400 karakter. 2 milyar karakter dinamik veriler için
Byte	1 bayt	0 to255
Boolean	2 bayt	True ya da False
Date	8 bayt	1/1/100 to 12/31/9999
Object	4 bayt	N/A
Variant	16 bayt +1 bayt her karakter için	N/A

Bir değişkenin tipi o değişkenin değeri içereceği veriyi ve bellekte nasıl tutulacağını gösterir. Bir değişken tanımlandığında ona bir veri tipi verilir. Bütün değişkenler bir veri tipi ile ne tür bir veriyi içerdiklerini belirtirler. Böylece bir değişkenin tipini belirtmek onun daha etkin kullanımını sağlar. Örneğin bir müşteri adı String tipinde tanımlanırken, müşterinin numarası Integer tanımlanabilir.

Değişken tipini tanımlamak için Dim, Private, Public, Static gibi deyimler kullanılır. Ardından tip belirtilir. Tipler Currency, Double, String gibi deyimlerle tanımlanır.

Private I As Integer
Dim Toplam As Double

```
Static ismi As String  
Public Odeme As Currency
```

Bir tanımlama deyimi ile çok sayıda değişken de tanımlanır:

```
Private I As Integer, Toplam As Double  
Private ismi As String, Odeme As Currency
```

Sayısal Veri Tipi

Visual Basic'te sayısal veri tipleri şunlardır: Integer, Long, Single, Double, Currency.

Long (long integer), Single (single-precision floating point), Double (double-precision floating point), ve Currency. Sayısal veri tipi variant ve diğer veri tiplerine göre daha az veri saklama alanına gereksinim duyulur.

Eğer bir değişken 100 ile 300 arasında bir değer alıyorsa onu Integer olarak tanımlayabilirsiniz. Eğer değer ondalık noktası varsa 3,5 gibi; o zaman Long tanımlamak gerekir. Örneğin döngü değişkenleri genellikle Integer tanımlanır.

Single ve Double veri tipleri ise noktadan sonra dört rakama sahip onbes basamaklık bir sayının kullanılmasını sağlar.

Byte Veri Tipi

0 ile 255 arasında bir değer içerir. Eğer değişken ikili (binary) veri içeriyorsa o zaman Byte dizi olarak tanımlanabilir.

```
Dim SiraNo As Byte
```

String Veri Tipi

Karakterlerden oluşan veriler String olarak tanımlanır. Örneğin müşteri adı gibi bir bilgi String olarak tanımlanmalıdır. String veriler sayıları içerebilir ancak sayısal işleme giremezler.

Değişken uzunluklu bir string tanımlama:

```
Bilgi As String
```

Değişken (değişir) uzunluklu bir string tanımlamak için sadece bilginin (değişkenin) string olduğu belirtilir.

Private S As String

String degiskenlere deger atamak için çift tirnak karakteri yaygın olarak kullanılır:

S = " Ugur Degirmenci "

Varsayım olarak bir string degisken uzunluktur. Böylece string deger istenildiği kadar uzun sözcükleri içerebilir. Bir diğer string tanımlaması ise sabit uzunluklu degiskenlerdir.

Sabit uzunluklu bir string tanımlama:

String * uzunluk

Sabit uzunluklu string tanımlamak için string tanımında asterisk karakteri ile uzunluk (kaç karakter) belirtilir. Örneğin 30 karakter uzunluğunda bir veri tanımlamak istersek şu şekilde tanımlarız:

Dim Adi As String * 30

Eğer 30 karakterden daha az deger atanırsa Adi degiskenine boşluk eklenerek 30 karaktere tamamlanır. Eğer 30 karakterden daha uzun bir veri atanırsa o zaman Visual Basic fazlalıkları keser. Sabit uzunluklu bir string boşlukla tamamlandığı için Trim ve Rtrim fonksiyonları ile boşluklardan kurtulunabilir.

NOT: Standart modüllerde yer alan sabit uzunluklu string degiskenler Public ya da Private olarak tanımlanmalıdır. Form ve Class modüllerinde ise Private olarak tanımlanmalıdır.

String verilere sayısal degerler atanabilir. Ancak bu degiskenlerle sayısal işlem yapılmaz. Ancak sayısal bir degeri olan string degisken sayısal bir degiskene atanabilir ya da dönüştürülebilir:

```
Private Sub Command1_Click
Dim TamsayiA As Integer
    Dim StrB As String
    StrB = "500"
    TamsayiA = StrB
' String deger tamsayi degiskene atanir.
End Sub
```

String Verilerle Çalışma

String işlemlerinde birleştirme operatörü (concatenation operator) yaygın olarak kullanılır. Birleştirme işlemi iki ya da daha çok string veriyi birleştirmeyi sağlar. Birleştirme işlemi & (ampersand) simgesi ile yapılır. Bu simge aracılığıyla bir string veri diğeri ile birleştirilebilir. Örneğin kodu ile adını birleştirmek gibi.

Birleştirme işlemi:

YeniString = String1 & String2 & String3 ...

Birleştirme işleminin ardından uzun string, YeniString'e atanır.

AdivAdresi = "Ali " & "Uzun"

Aşağıdaki örnekte iki string bilgi birleştirilmiştir:

```
Adi$ ="Ayse"  
Soyadi$="Yilmaz"  
Adresi$="1234 Sakak No 1"  
Toplam$ = Adi$ & " " & Soyadi$ & " " & Adresi4$
```

Boolean Veri Tipi

Eğer bir değişken sadece true/false, yes/no gibi iki durumlu bir değeri içeriyorsa değişken Boolean olarak tanımlanır. Boolean değişkenlerin varsayımlı değeri False'dir.

Örnek:

```
Dim Kaldi_Geçti As Boolean  
Dim Cinsiyet As Boolean  
Dim Bayrak As Boolean
```

Date Veri Tipi

Tarih ve zaman değerleri Date veri tipiyle tanımlanan değişkenlerle saklanır. Bunun yanı sıra Variant veri tipi de tarih ve zaman verisi içerebilir.

```
Dim Tarih As Date
```

Object Veri Tipi

Object veri tipi 32-bit { 4 byte) olarak saklanır ve bir nesnenin adresini içerir.

```
Dim objDb As Object
Set objDb = OpenDatabase ("c: \my documents\adresler.mdb")
```

Bir object degiskenleri özel olarak tanımlanırlar ve bu class'lar Object Browser'da listelenir.

Örnek: Nesne tanımlama

```
Dim xl As Object
Set xl = GetObject ("c: \my documents\butce.xls")
```

Veri Tiplerinin Dönüştürülmesi

Visual Basic çok sayıda dönüştürme fonksiyonuna sahiptir. Bu fonksiyonlar degerlerin tiplerinin birbirlerine çevrilmesini sağlar.

Fonksiyon	Dönüştürür
Cbool	Boolean
Cbyte	Byte
Ccur	Currency
CDate	Date
Cdbl	Double
CInt	Integer
CLng	Long
CSng	Single
CStr	String
CVar	Variant
CVErr	Error

NOT:

Bir veri tipindeki deger bir baska tipe çevrilecegi zaman karsi tarafın tipine uygun olmalıdır. Örneğin Long bir deger Integer bir degiskene atanacaksa Long deger Integer deger sinirlarında olmalıdır.

```
Ucret = Ccur(Text1.Text)
```

Variant Veri Tipi

Variant veri tipi bütün verileri (her türdeki) saklayabilen bir veri tipidir. Eger herhangi bir veri tipini Variant tipli bir degiskene atiyorsanız herhangi bir dönüşüme gereksinim duyulmaz. Bir Variant veri içerigine bakılmaksizin 16 byte alan kaplar. Bununla birlikte Variant veri tipi, tipleri tanımlanmayan degiskenlerin varsayım tipini olusturur. Variant degisken içindeki verinin tipini öğrenmek için VarType fonksiyonu kullanilir.

VarType Fonksiyonu

Bir degiskenin tipini belirten bir tamsayi deger döndürür.

Yapisi:

VarType(degisken)
Degisken bilgisi variant bir degerdir.

Dönen deger:

Sabit	Deger	Açıklama
vbEmpty	0	Empty (bos), deger verilmemis
vbNull	1	Null (geçerli veri yok)
vbInteger	2	Integer (tamsayi)
vbLong	3	Long integer
vbSingle	4	Single-precision floating-point sayi
vbDouble	5	Double-precision floating-point sayi
vbCurrency	6	Currency (para birimi)
vbDate	7	Date (tarih)
vbString	8	String
vbObject	9	Object
vbError	10	Hata degeri
vbBoolean	11	Boolean (ikili) deger
vbVariant	12	Variant
vbDataObject	13	Veri erisim nesnesi
vbDecimal	14	Decimal deger
vbBvte	17	Byte deger
vbUserDefinedType	36	user-defined tip içeren veri tipi
vbArray	8192	Array (dizi)

Asagidaki örnekte degiskenlerin tipi belirlenmektedir:

Dim TamsayiDegisken, StringDegisken, TarihDegisken, Kontrol


```
TamsayiDegisken = 123
StringDegisken = " Cem Yazici "
TarihDegisken = #22/06/1983#
Kontrol = VarType (TamsayiDegisken] ' 2 döner.
Kontrol = VarType (TarihDegisken) ' 7 döner.
Kontrol = VarType (StringDeger) ' 3 döner.
```

Asagidaki örnekte ise ekrandan alınan bir degerin tipi belirlenir:

```
Private Sub Command1_Click()
    Dim Defter, Kontrol
    Deger = InputBox("bir deger girin")
    Kontrol = VarType(Deger)
    Select Case Kontrol
        Case 0 : MsgBox "Bos"
        Case 1 : MsgBox "Null"
        Case 2 : MsgBox "Tamsayi"
        Case 3 : MsgBox "Long Tamsayi"
        Case 4 : MsgBox "Single-precision floating-point sayi"
        Case 5 : MsgBox "Double-precision floating-point sayi"
        Case 6 : MsgBox "Para birimi"
        Case 7 : MsgBox "Tarih"
        Case 8 : MsgBox "String"
        Case 9 : MsgBox "Nesne"
        Case 10 : MsgBox "Hata"
        Case 11 : MsgBox "ikili"
        Case 12 : MsgBox "variant"
        Case 13 : MsgBox "data access object-veritabani nesnesi"
        Case 13 : MsgBox "onlu sayi"
        Case 14 : MsgBox "Byte"
        Case 15 : MsgBox "User-defined veri içerir"
    End Select
End Sub
```

Örneğin bir variant degiskene bir onlu sayi atanırsa Variant degisken içinde bu deger Double olarak yer alır.Eger büyük ve duyarlı bir sayiya gereksinim duyulmuyorsa o zaman Variant veri Single ya da Integer veriye dönüştürülür.

```
If VarType(VarDeg) = Then X = CSng(X)
' Sigle'a çevirme
```

Variant İçinde Saklanan Sayısal Veriler

Bir Variant değişken sayısal değerler de içerebilir. Visual Basic sayıları en az ver kaplayacak biçimde Variant değişkenlerde saklar. Örneğin eğer bir tamsayı değer bir variant değişkende saklanırsa sayı Integer olarak saklanır. Eğer sayı daha büyükse sayıya göre Long ya da Double değer kullanılır. Variant değişken içindeki sayısal değerle hesaplama yapılabilir. Bunun dışında diğer özel tiplerin variant olarak saklanması için Visual Basic'in sahip olduğu çok sayıda fonksiyon kullanılır. Örneğin değerin Currency tipine dönüştürülmesi için CCur fonksiyonu kullanılır:

```
Haftalik = CCur(Saat * Saatücreti)
```

IsNumeric fonksiyonu bir Variant değişkenin içerdiği değerin geçerli bir sayısal değer olup olmadığını belirlemek için kullanılır.

```
Do  
    Sayi = InputBox(" Bir sayı girin ")  
Loop Until IsNumeric(Sayi)  
MsgBox " Sayının karakökü: " & Sqr(Sayi)
```

Variant İçinde String Değer Saklama

String değerlerin Variant değişkenler içinde saklanmaları bir tip sorunu yaratmaz. Ancak artı (+) karakterlerinin kullanılması bazen belirsiz durumlar yaratabilir. Örneğin iki Variant değişken de sayısal değer içeriyorsa o zaman + operatörü iki değerin toplanmasını sağlar. Eğer iki variant değişkenin içeriği de String ise o zaman iki değerin + ile bir araya getirilmesi onların karakter olarak birbirine eklenmesini sağlar. Ancak eğer değerlerden birisi sayı ise Visual Basic sayıyı String olarak dönüştürmeye çalışır. Eğer başarılı olursa iki değeri toplar. Başarılı olmazsa "Type mismatch error" hatası verir. Bu nedenle String işleminde içeriğe bakılmaksızın başarılı olmak için & operatörü kullanılmalıdır.

Örnek: String birleştirme

```
Sub Form_Click()  
    Dim X, Y  
    X = "4"  
    Y = "5"  
    Print X + Y, X & Y  
    X = 4  
    Print X + Y, X & Y
```

Yukarıdaki procedure su sonuçları üretir:

45	45
9	45

Variant İçinde Saklanan Tarih ve Zaman Değerleri

Variant değişkenler Date ve Time değerleri de saklarlar. Birçok fonksiyon tarih ve zaman değeri üretir. Ayrıca tarih ve zaman üzerinde aritmetik işlemler de yapılabilir. Bu nedenle variant değişkenler içinde yer alan tarih ve zaman verilerinin durumuna bakalım:

Örnek: Tarih ve zaman bilgilerinin kullanımı

```
Private Sub Form_Click()
    Dim suan, geçengün, geçensaat, geçendakika
    'değerler variant olarak tanımlanır.
    suan = Now ' su anki tarihi ve zamanı verir.
    geçengün = Int(DateSerial (Year (suan) + 1, 1, 1) - suan)
    geçensaat = 24 - Hour(suan)
    geçendakika = 60 - Minute(suan)
    Print geçengün & " yılda geçen gün."
    Print geçensaat & " günde geçen saat."
    Print geçendakika & " saatte geçen dakika."
End Sub
```

Tarih ve zaman verileri üzerinde matematik işlemler de yapılabilir. Toplama ve çıkarma işlemi ile günler ve saatler toplanabilir ve çıkarılabilir. Variant değişkenlere saklanan tarih değerleri January 1, 0100 ile December 31, 9999 arasındadır. Tarih ve zaman değerlerinin kullanımında sayı karakteri de kullanılabilir. Aşağıdaki örnekte geçerli tarih değerleri yer almaktadır:

```
Tarih = #3 - 6 - 98 13:20#
Tarih = #March 21, 1998 3:45am#
Tarih = #Apr - 2 - 98#
Tarih = #12 January 1999#
```

Bir variant değişkenin içerdigi değerin geçerli bir tarih değeri olup olmadığı **IsDate** fonksiyonu ile anlaşılar. Ardından **Cdate** fonksiyonu ile değer tarihe çevrilir. Aşağıdaki örnekte bir metin kutusu içinde yer alan tarih verisi test edilmekte ve kullanılmaktadır:

```
Dim Tarih, geçengün
If IsDate (Text1.Text) Then
    Tarih = CDate(Text1.Text)
    geçengün = DateSerial (Year (Tarih) + _
```

```
1, 1, 1) - Tarih
Text2.Text = geçengün & " yıl içinde geçen gün."
Else
MsgBox Text1.Text & " geçersiz bir tarih."
End If
```

Empty Degeri

Bazen bir degiskene bir deger atanip atanmadigini bilmek isteyebilirsiniz. Bir variant degiskene bir deger atanmadan önce Empty (bos) degerine sahiptir. Empty degerinin anlami sifir degildir. Empty degeri Null ya da bos string " " anlamindadir. Bir degiskenin Empty degeri **IsEmpty** fonksiyonu ile belirlenir.

```
If IsEmpty (Toplam) Then Toplam = 0
```

Bir variant Empty degerine sahip oldugu zaman; degisken ifadeler içinde kullanılabilir. Ve degeri ifadeye göre 0 ya da sifir uzunluklu string olarak algılanir. Degiskene bir deger (sifir, null ya da sifir uzunluklu string} atandigi zaman Empty degerini yitirir.

Null Degeri

Bir variant verinin içerebilecegi bir diger özel veri de Null'dir. Null degeri özellikle veritabani uygulamalarında bilinmeyen ya da olmayan veriyi ifade eder. Null verisinin özellikleri sunlardir:

- Ifadelerin bir kısmi Null ise tamami Null olarak degerlendirilir.
- Bir Null degeri geçirmek, bir Null degeri içermek genellikle fonksiyonun Null degerini döndürmesini saglar.

Null degerler "Null" deyim ile atanirlar:

```
Toplam1 = Null
```

Bir variant degiskenin Null degere sahip olup olmadigini test etmek için **IsNull** fonksiyonu kullanilir. Ayrıca degiskenlere Null degeri vermek için açık olarak Null degerini atamak gerekir.

```
If IsNull (Toplam1) And IsNull(Toplam2) Then
K = Null
Else
K = 0
End If
```

Null degeri, Variant disinda bir degiskene atanirsa o zaman hata olurur. Variant deger döndüren bütün fonksiyonlardan Null degeri döndürülebilir.

Veri Tiplerinin Degisken Adlariyla Kullanimi

Degiskenlerin tiplerinin normalde degisken adlariyla hiçbir ilgisi yoktur. Ancak daha kontrollü olmak bakimindan degisken adlari tiplerine göre de verilebilir.

Yaygin olarak kullanılan kisaltmalar

Degisken Tipi	Kisaltma	Örnek
Array	a	aAylar
Currency	cur	curTutar
Double	dbl	dblToplam
Integer	int	intSira
String	str	strAdiSoyadi
Variant	var	varTarih

Diziler (Arrays)

Her zaman degiskenlerin tümünü tek bir adla ve tek bir içerikle kullanmak kısıtlayıcı olabilir. Bu nedenle diziler (arrays) geliştirilmiştir. Bir dizi aynı tipte ve aynı adı paylaşan bir grup degisken demektir. Diziler birçok degiskene aynı adla ulaşmayı sağlayan bir grup veri yapısıdır. Bir indeks numarası ile dizi içindeki elemanlara ulaşılır. Dizi içindeki elemanlar aynı tipteki verilerdir; örneğin haftanın günleri ya da iller gibi.

Ocak
Subat
Mart
Nisan
...
...
Aralık

Aylar Dizisi

Öğrencilerin notlarını toplamak ve ortalamasını almak istiyorsunuz?Yüzlerce öğrencinin notlarını tek tek toplamak ve üzerinde işlem yapmak yerine onları bir grup olarak tanımlamak dizilerin kullanımını ortaya koyar. Dizilerin en önemli yanı özellikle döngüler ile birlikte kullanılmasıdır. Böylece çok sayıda işlem kısa zamanda ve bir blok kod içinde yapılabilir.

Dizi tanımlama:

Dim dizidegiskeni (eleman sayısı) As tipi
Dizi degiskeni dizinin adini gösterir.

Eleman sayısı dizinin eleman sayisini gösterir. Dizilerin eleman sayılan sıfırdan baslar. Bes elemanlı bir dizi: sıfırdan başladığı için eleman sayısı 4 olarak belirtilir.

'yedi elemanlı günler dizisi

Dizi_Günler (6) As String

'Dizinin indeksi 0'dan 6' ya kadar degisir

Dizi_Günler(0) = "Pazartesi" 'dizinin birinci elemani

Dizi_Günler(1) = "Salı" 'dizinin ikinci elemani

Dizi_Günler(2) = "Çarsamba" 'dizinin üçüncü elemani

Dizi_Günler(3) = "Perembe" 'dizinin dördüncü elemani

Dizi_Günler(4) = "Cuma" 'dizinin besinci elemani

Dizi_Günler(5) = "Cumartesi" 'dizinin altinci elemani

Dizi_Günler(6) = "Pazar" 'dizinin yedinci elemani

Bunun disinda diziler birden çok boyutu ile matematikten bildigimiz matrislere de karsilik gelen veri yapılarıdır.

Dim YillarSatislar(10, 5) As String

Dizilerin kullaniminda bir diger konu dizilerin boyutlarinin (eleman sayılarının) degistirilmesi ile ilgilidir. Dinamik dizilerin büyüklükleri degistirilebilirken, sabit büyüklükteki dizilerin büyüklükleri sonradan degistirilemezler.

Bir Dizinin Tanımlanması

Bir diziye tanımlamak için üç yol vardır. Bu tanımlamalar dizi elemanları için gereksinim duyulan kapsama alanına göre kullanılabilir.

- Public dizi tanımlama
- Modül düzey dizi tanımlama
- Yerel dizi tanımlama

Dim aylar(11) As String ' 12 elemanlı dizi.

Diziler tanımlandıktan sonra belli işlemler yapılır. Bunlar:

- Veri doldurma
- Verilere erisme

Veri doldurmak ya da erismek için döngü yapisi kullanilir. Döngü kontrol degiskenleri dizi elemanlarinin indekslerini temsil ederek sirayla elemanlarin kullanilmasini saglar. Asagidaki örnekte 11 elemanli bir diziye; 0 dan baslayarak 10'a kadar deger girilmektedir.

```
For i = 0 to 10
    Dizi(i) = Inputbox ("dizinin" & i & "elamanini girin");
Next i
```

Bir dizi içindeki veri elemanlari ayni veri tipine sahiptir. Dizideki elemanlar her türde olabilirler. Eger veri tipi variant ise dizi elemanlari farkli tiplerde olabilir. Dizi tanimada bir diger konu da dizinin kapsamidir. Kapsam ayni degiskenlerde oldugu gibi dizi elemanlarinin kullanabileceklerialani da gösterir:

- Public bir dizi yaratilacaksa Declarations kesiminde Public deyimi kullanilir.
- Modül düzeyinde dizi yaratilacaksa modülün Declarations kesiminde Private deyimi kullanilir.
- Yerel bir dizi yaratilacak o zaman Procedure içinde Private deyimi kullanilir.

Visual Basic'te iki tür dizi vardır:

- Sabit-eleman sayili dizi
- Dinamik dizi

Sabit Eleman Sayili Dizi

Sabit eleman sayili (sabit büyüklükteki) dizilerin eleman sayilari sabittir. Daha sonra eleman sayilari degistirilemez.

```
Dim Günler (6) As String
Dim Aylar (11) As String
Dim Notlar (3) As Integer
```

Dinamik Diziler

Bazen bir dizi yaratirken eleman sayisini tam olarak kestiremeyiz. 20 ya da 30 olabilir. Iste bir dizinin eleman sayisini sonradan degistirebilme özelligine dinamik diziler denir. Dinamik bir dizinin boyutlari istenildigi zaman degistirilebilir. Dinamik dizilerin eleman sayilarinin degistirilebilmesi ayni zamanda bellegin de etkin olarak kullanilmasini saglar. Böylece dizi önce daha küçük olarak tanimlanir. Ardindan büyütülür.

Dinamik bir dizi tanımlamak:

Bir dinamik dizi tanımlamak için modül düzeyinde Public ya da Dim deyimi kullanılarak dizi tanımlanır. Eger dizi procedure düzeyinde kullanılacaksa zaman procedure düzeyinde Static ya da Dim deyimi kullanılır.

Bir dinamik dizi bos olarak tanımlanabilir:

```
Dim DinamikDizi ()
```

Ardından istenildiği zaman (daha sonra) dizinin boyutları değiştirilebilir. Bu işlem için ReDim deyimi kullanılır:

```
ReDim DinamikDizi (10)
```

ya da daha sonra;

```
ReDim DinamikDizi (10+x)
```

ReDim deyimi sadece bir procedure içinde kullanılır. Dim vs Static deyimlerinden farklı olarak ReDim deyimi işletilebilen bir deyimdir. ReDim deyimi uygulamanın çalışma zamanında dizinin yeniden boyutlandırmasını sağlar. ReDim deyiminde sabit olarak dizi eleman sayısı ya da dizinin alt ve üst sınırı belirtilir.

```
ReDim DinamikDizi (4 to 12)
```

Örneğin Matris1 adlı dinamik dizi modül düzeyinde tanımlanır:

```
Dim Matris1() As Integer
```

Ardından procedure içinde dizinin yeniden boyutlandırılması sağlanır:

```
Sub YenidenHesapla()  
...  
...  
...  
ReDim Matris1 (9, 9)  
End Sub
```

ReDim deyiminin her kullanılışında dizideki bütün değerler önce yok olur. Ardından Visual Basic bütün elemanları sıfırlar. Bu işlemde variant değerlere Empty, sayısal değerler sıfır, string değerler sıfır uzunluklu verilerle doldurulur. Bu nedenle dizinin veri kaybı olmadan eleman sayısını artırmak

için ReDim deyimi ile Preserve deyimi kullanilir. Ayrica dizinin üst sinirini göstermek için de **UBound** deyimi kullanilir.

ReDim Preserve DinamikDizi (UBound (DinamikDizi) + 1)

Yukaridaki deyim ile dinamik dizinin eleman sayisi artirilir.

Çok Boyutlu Diziler

Bir grup veri yapisi olan diziler belli sayida elemani içermenin yani sıra birden çok boyuta da sahiptirler. Bunun anlami çok sayida bilginin bir arada kullanilabilmesidir.

Dim aylar_satislar (11,4) As String

Yukaridaki dizi ile 12 elemanli satislarin yani sıra bes ürünün satis bilgileri de takip edilir.Bu durumda dizi tanimlamalari:

aylar_satislar (0,0) ' Birinci ayda birinci ürünün satis rakami

aylar_satislar (0,1) ' Birinci ayda ikinci ürünün satis rakami

aylar_satislar (0,2) ' Birinci ayda üçüncü ürünün satis rakami

Ocak	50000
Şubat	50000
Mart	50000
Nisan	3000
▪	3000
▪	3000
Aralık	3000

Sekil4.15 İki boyutlu dizi

Dizilerin Alt ve Üst Sinirlariyla Tanimlanmasi

Diziler bazen verilere daha uygun olarak kullanilmak istenirler. Örneğin bir matris sekinde belli bir boyutta verilerin kullanilmasi gibi.

Yillar_Satislar (1990 to 2000, 1 to 5) As Double

Yukarıdaki tanımlanan bes tane ürünün 1990 yılından 2000 yılına kadar olan satış bilgilerini tutar. Yine çok boyutlu dizilerinde alt ve üst sınırlarıyla daha açık tanımlanması mümkündür. Örneğin aşağıdaki dizi 10 X 10 toplam yüz elemanlı ve iki boyutlu bir dizidir:

Static Matris(9, 9) As Double

Boyutlardan birisi ya da ikisi alt boyutlarıyla tanımlanabilir:

Static Matris(1 To 10, 1 To 10) As Double

Diger Dizileri İçeren Diziler

Faklı veri tiplerinden oluşan dizileri içeren bir Variant tipli bir dizi yaratılabilir. Aşağıdaki örnekte önce iki dizi yaratılmaktadır. Bu dizilerden birisi string diğeri tamsayı değerleri içermektedir. Ardından üçüncü bir dizi yaratılır ve bu dizileri içine alır.

```
Private Sub Command1_Click()  
    Dim TamsayiX As Integer  
    ' sayaç tanımlanır.  
    ' tamsayı dizi tanımlanır.  
    Dim SayaçA (5) As Integer  
    For TamsayiX = 0 To 4  
        SayaçA(TamsayiX) = 5  
    Next intX  
    'string dizi tanımlanır.  
    Dim SayaçB(5) As String  
    For TamsayiX = 0 To 4  
        SayaçB(TamsayiX) = "Merhaba"  
    Next TamsayiX  
    Dim DiziX(2) As Variant  
    ' yeni iki elemanlı dizi tanımlanır.  
    DiziX(1) = SayaçA()  
    ' Diziyi diğer dizi ile doldur.  
    DiziX(2) = SayaçB()  
    MsgBox DiziX (1) (2)  
    ' Dizinin elemanlarını göster.  
    MsgBox arrX (2) (3)  
End Sub
```

Dizilerde Option Base Deyimi Kullanimi

Option Base deyimi modülün genel kısmında kullanılır ve bir dizinin indeksinin alt sınırını belirler.

Yapisi: Option Base {0 | 1}

Bir dizinin indeksinin alt sınırı varsayım olarak 0' dir. Eger bu deger degistirilecekse o zaman 1 degeri kullanilir.

Option Base deyimi içinde sadece bulunduğu modülde dizilerin alt sınırlarını etkiler. Asagidaki örnekte dizilerin indeks degeri degistirilmektedir:

```
Option base 1 ' Dizi indeksleri 1 degerini alır..
Dim Alt
Dim Dizi (1 To 10, 5 To 15, 20 To 30)
' dizi degiskenleri tanimlanir.
Dim Dizi2(10)
Alt= Lbound (Dizi, 1) ' 1 döndürür.
Alt= Lbound (Dizi, 3) ' 20 döndürür.
Alt= Lbound (Dizi2)
' Option Base deyiminde göre 0 ya da 1 degeri döndürür.
```

LBound fonksiyonu **UBound** fonksiyonu ile birlikte de kullandir. **UBound** ise dizinin büyüklüğünü ve boyutlarının ve üst limitini belirler. **UBound** fonksiyonunu **LBound** fonksiyonu ile birlikte kullanilir. Lbound fonksiyonu dizinin bir boyutunun alt limitini gösterir.

```
Dim A ( 1 To 100, 0 To 3, -3 To 4)
```

Yukaridaki deyime göre Ubound fonksiyonu asagidaki degerleri döndürür:

```
UBound(A, 1)    100
UBound(A, 2)     3
UBound(A, 3)     4
```

Asagidaki örnekte dizinin belirtilen boyutunun en büyük degeri hesaplanmaktadır:

```
Dim Üst
Dim Dizi (1 To 10, 5 To 15, 20 To 30)
' dizi degiskenleri tanimlanmaktadır.
Dim Dizi2(10)
Üst= UBound(Dizi, 1) ' 10 döndürür.
Üst= UBound(Dizi, 3) ' 30 döndürür.
```

Üst= UBound(Dizi2) ' 10 döndürür.

Kullanici Tanimli Veriler

Kullanici tanimli veri (user defined data type) farkli tiplerdeki verileri içeren özel bir birlesik veri tipidir. Bu degiskenlere tip degiskenleri de denir. Özellikle bir birim veri tanimi içinde farkli tipteki degiskenleri kullanmak istiyorsak kullanici tanimli veri tipini kullanabiliriz. Kullanici tanimli veri tipi yaratmak için **Type** deyimi kullanilir. Bu deyim modülün Declarations kesiminde yer alır. Kullanici tanimli veri tipleri Private ya da Public olarak tanimlanabilirler.

Private Type MyDataType

Yada; Public Type MyDataType

Örneğin bir öğrenci kaydı ya da bir müşteri kaydı bilgisi kullanici tanimli degisken olarak tanimlanabilir:

```
' Tanimlamalar
Private Type Öğrenci
    AdıSoyadı As String
    Bölümü As String
    Vize1 As Integer
    Vize2 As Integer
    Final As Integer
    Durumu As String
End Type
```

Kullanici tanimli degiskenlere deger atamak için özelliklerin kullanimina benzer biçimde bir yöntem kullanilir:

Öğrenci.AdıSoyadı = "ahmet uzun"

Bunun disinda eger yapıları aynıysa kullanici tanimli degiskenler birbirlerinde de atanabilirler.

Öğrenci = YedekÖğrenci

Bir Dizi İçeren Kullanici Tanimli Tipler

Bir kullanici tanimli degisken sabit uzunluklu bir diziyi de içerebilir.Örneğin Öğrenci tip degiskeni içinde notlar dizisi.

```
' Tanimlamalar
Private Type Öğrenci
```

```
AdiSoyadi As String
Bölümü As String
Notlar (9) As Integer
Durumu As String
End Type
```

Ayrıca bir dinamik dizi de tip değişkeni içinde yer alabilir:

```
' Tanımlamalar
Private Type Öğrenci
    AdiSoyadi As String
    Bölümü As String
    Notlar () As Integer
    Durumu As String
End Type
```

Tip değişkeni içindeki dizi elemanlarına aynı kullanıcı tanımlı verilerin elemanlarına erişildiği gibi erişilir. Bu elemanların gösterimi için şu yöntem kullanılır:

```
Dim DiğerÖğrenciler As Öğrenci
ReDim Öğrenci.Notlar (3)
Öğrenci.Notlar (0) = 45
```

Bunların dışında kullanıcı tanımlı değişkenlerin dizileri de oluşturulabilir:

```
Dim OkulÖğrencileri (100) As Öğrenci
OkulÖğrencileri (0).Bölümü = "Mühasebe"
OkulÖğrencileri (0).Notlar (0) = 50
```

Kullanıcı-Tanımlı Değişkenlerin Procedure'lara Geçirilmesi

Kullanıcı tanımlı değişkenler de bir argüman olarak procedure'lara geçirilirler.

```
Sub ÖğrenciYerleştir(Ogr As Öğrenci)
    Ogr.AdıSoyadı = Text1.Text
    Ogr.Bölümü = Text2.Text
    Ogr.Durumu = Text3.Text
End Sub
```

İç İç Olan Veri Yapıları

Veri yapıları birbirlerinin içine girebilirler. Diğer bir deyişle bir kullanıcı tanımlı değişken diğer bir kullanıcı tanımlı değişkenin elemanı olabilir.

Tanımlamalar:

```
Type Notlar
    Adi As String
    Degeri As Integer
End Type
Private Type Öğrenci
    AdiSoyadi As String
    Bölümü As String
    Notu (9) As Notlar
    Durumu As String
End Type
Type DriveInfo
    Type As String
    Size As Long
End Type
Type SystemInfo
    CPU As Variant
    Memory As Long
    DiskDrives(26) As DriveInfo
    Maliyet As Currency
    AlisTarihi As Variant
End Type
```

Kullanımlari:

```
Dim Sistemler(100) As SystemInfo
Sistemler(1).DiskDrives(0).Type = "Floppy"
Dim Okul (100) As Öğrenci
Okul (0).Öğrenci(0).Notu(0).Adi = "Türkçe"
```

Numaralandırma

Grup verileri islerken basvurulacak bir diger kullanım ise numaralandırma (enumerations) dir. Numaralandırma sabitlerin belli adlarla temsil edilmesini saglar. Örneğin ayların sayılar yerine adlarıyla kullanılması gibi. Numaralandırma **Enum** deyimi ile modülün **Declarations** kesiminde yapılır. Numaralandırma tipi Private ya da Public olabilir.

```
Private Enum Numaralandırma
```

```
Yada; Public Enum Numaralandırma
```

Varsayım olarak; numaralandırmanın ilk sabitinin değeri sıfırdır. İzleyen değerler birer artarak devam eder. Aylar örneğine bakarsak; Ocak ayının numarası 0'dır. Şubat ayı 1, Mart ayı 2 olarak devam eder.

```
Public Enum Aylar
    Ocak
    Şubat
    Mart
    Nisan
    Mayıs
    Haziran
    Temmuz
    Ağustos
    Eylül
    Ekim
    Kasım
    Aralık
End Enum
```

Numaralandırma işleminin ardından sabitlerin sayısal değeri şu şekilde kullanılabilir:

```
Dim IsAylar As Aylar
IsAylar = Mart
If IsAylar < Nisan Then
    ' burada nisan ayının değeri 3'tür.
```

Yukarıdaki örnekte gördüğümüz gibi numaralandırma sabitlere bir sıra değeri vermiş ve onların bu değere göre karşılaştırılmasını sağlamıştır.

Program Denetimi

Program içinde tanımlama deyimleri, atama deyimleri yer alır. Ancak programın işleyişinde "eğer böyleyse şöyle yap" gibi sapmalar ya da kararlar vermek gerekebilir. İşte bu durumda programın işleyişini, akışını denetleyen "denetleme deyimleri" kullanılır. Denetleme deyimleri standart yapılar sayesinde istenilen bir seçimi, sapmayı ya da kararın alınmasını sağlar.

Program Denetimi Yapıları:

- Karar (sapma)
- Döngü

Klasik (geleneksel) programlamada programın akışı siradan gider. Diğer bir deyişle programın işletimi deyimlerin sırasıyla yürütülmesiyle devam eder. Bir karar, döngü ya da bir sapma deyimini ile karşılaşılması durumunda programın işletim sırasını değiştirir. Karar yapıları sayesinde elde edilen değere göre farklı işlemlerden birisi yapılır. Örneğin "gelir sundan büyükse kesinti oran şu şekilde...". Ana karar yapıları şunlardır:

Karar yapıları:

- If...Then...Else
- Select Case

Karar yapılarının yanı sıra yaygın olarak kullanılan bir diğer yapı da döngü (loop) türü. Döngüler özellikle programın etkinliği bakımından önemli bir programlama tekniğidir.

Döngü Yapıları:

- Do...Loop
- For...Next

Döngüler

Program denetiminde yaygın olarak kullanılan mekanizmalardan birisi de döngülerdir. Döngüler işlemlerin yinelenmesi anlamına gelir. Örneğin dosyanın sonuna kadar kayıtların birer birer okunması ve her okunan kayıt üzerinde belli işlemlerin yapılması gibi. Çünkü burada yapılacak işlem aynıdır ama dosyada yer alan bütün kayıtlar için uygulanır. Visual Basic'te iki ana tip döngü vardır:

- Kosullu döngüler
- Sayaçlı döngüler

Kosullu döngüler belli bir koşul yerine gelinceye kadar ya da belli bir koşul olduğu sürece boyunca yineleme işlemini sürdürürler. Sayaçlı döngüler ise işlemleri belirtildiği kadar yinelerler.

Döngü yapıları:

Do...Loop: Bir koşula göre döngü.

For...Next: Belli bir sayıda döngü (sayaçlı).

Bir sayaçlı döngü For...Next döngüsü olarak bilinir. For deyimini ile döngünün sayacı tanımlanır. Next deyimini ile yinelemek blok sona erer.

Döngünün her dönüşünde sayaç degiskeni bir sayi artirilir. Böylece bir sayidan digerine kadar islemler sürüdürölür.

Do...Loop Deyimi

Bir blok deyimi verilen kosul dogru (True) oldugu sürece isletir.

Yapisi:

```
Do [( While | Until) kosul1]
    [ ifade blogu]
[Exit Do]
    [ ifade blogu]
Loop
```

Bir blok, deyimi verilen kosul saglanincaya kadar isletir.

Yapisi-2:

```
Do
    [ifade blogu]
[Exit Do]
    [ ifade blogu]
Loop [(While | Until) kosul1]
```

Do...Loop deyiminde kullnılan elemanların islevleri sunlardır:

Eleman	Islevi
Do	Do..Loop kontrol yapisinin ilk deyimi.
While	Iken: Belirtilen kosul dogru oldugu sürece döngünün isleyeceğini belirtir.
Until	'e kadar: Kosul dogru oluncaya kadar döngü isletilir.
kosul	Sayisal ya da karakter bir ifade; dogru (0 hariç) ya da yanlış (0 ya da Null) degeri degerlendirir. ifade blogu Do ile Loop arasında; kosul saglanincaya kadar isletilecek olan program satirlari.
Exit Do	Döngüden çıkisi saglar.Bir If-Then deyimi ya da diger kosullarda döngüden ayrilmayi saglar.Istenildigi kadar kullanılabilir.
Loop	Do..Loop döngüsünü bitirir.

Do döngülerinin temelinde kosul (condition) yatar. Bir kosul True ya da False olabilen bir ifadedir. Bu bir fonksiyon (örneğin EOF), özellik ya da bir iki degerin bir iliski operatörü ile karsilastirilmesi yapılabilir.

İliski operatörleri

Operatör	Anlami
<	Küçüktür
<=	Küçük esittir
>	Büyüktür
> =	Büyük esittir
<>	Esit degildir
=	Esittir
And	Ve
Or	Veya - Ya da
Not	Degil

Örnek: Bir döngü kurma

```
Do While Ücret < 10000 ' Ücret 10000 den küçük olduğunda
    Toplam = Toplam + Ücret
Loop
```

İki temel Do döngüsü vardır: Do While Döngüsü, Do Until Döngüsü. Do...While döngüsünde While deyiminde belirtilen koşul doğru olduğu süre boyunca döngü yinelenir. Kosulun sağlanmadığı anda program döngüyü atlayarak Loop deyiminden sonraki deyimlere geçer.

Örnek: Bir Veritabanını İşleme

```
Do While Not Tablo.EOF
    Tablo.Edit
    If Bakiye > 100000 Then
        Durumu = "ödeme mektubu gönder"
    Else
        Durumu = "ekstre gönder"
    End If
    Tablo.Update
    Tablo.MoveNext
Loop
```

While deyiminde koşul doğru olduğu sürece döngü sürüdürülür. Until deyiminde ise koşul doğru olmadığı (yanlış olduğu süre) süre boyunca döngü yinelenir. Do...Until döngülerinde ise Until deyiminde belirtilen koşul doğru

oluncaya kadar döngü yinelenir. Diğer bir deyişle koşul yanlış (False) olduğu sürece döngü yinelenir. Do...Until döngüler, yaygın olarak veritabanı işlemlerinde kullanılır. Bir döngü ile tablonun ilk kaydından son kaydına kadar okunması sağlanır.

Örnek: Dosya sonuna kadar okuma

```
Dim Vt As Database, Tablo As Recordset
Set Vt = DBEngine.Workspaces(0).Databases ("C: \my
documents\adresler.MDB")
Set Tablo = Vt.OpenRecordset ("Ogrenciler", dbOpenTable)
Tablo.Movefirst
Do Until Tablo.EOF
    YzListe.AddItem Tablo! ("Ogrenciler")
    Tablo.MoveNext
Loop
Tablo.close
vt.Close
```

NOT:

Do While döngüsü içinde bir de Loop Until deyişi konulmaz. Bu durum hataya neden olur.

Aşağıdaki örnekte dosya sonuna kadar işlem yapılır. Örnekte MYFILE dosyası text bir dosyadır.

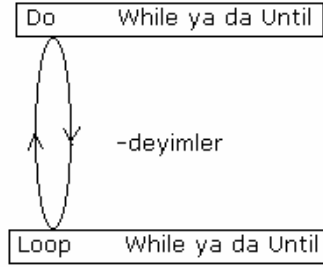
```
Dim InputData
Open "MYFILE" For Input As #1 ' input açılan dosya.
Do While Not EOF(1) ' end of file kontrolü.
    Line Input #1, InputData ' data oku.
    Debug.Print InputData ' Immediate window'a yaz.
Loop
Close #1 ' dosya kapat.
```

Aşağıdaki örnekte ise veriler; dosyanın sonundan başına doğru (ters sırada) okunur.

```
MyRecordset .moveLast
Do While not MyRecordset.EOF
    Print MyRecordset.fields.GetValue("myfield")
    MyRecordset.movePrevious
Loop
```

Döngünün Basinda ve Sonunda Denetim

Do... Loop döngüsünün basi ve sonu vardır. While ve Until deyimleri (iken ve 'e kadar) kendi anlamlarını sürdürmekle birlikte döngü yapısının basında ve sonunda kullanılmasına göre değişik anlamlar tasirlar.



Sekil4.16 While ve Until deyimlerinin döngünün basında ve sonunda kullanılması

Kosulların DO döngüsünün basında yer alması denetimin döngünün basında yapılması anlamına gelir. Döngü basında kullanılan Until deyimi koşul False olduğu süre boyunca döngünün islemesini ve koşul testinin döngünün basında her seferinde yapılmasını sağlar. While deyimi ise yine döngü basında koşulun True olması durumunda geçişe izin verir.

NOT:

Do... Until yerine While..Wend deyimleri ile de döngü yaratılabilir. Ancak daha esnek oldukları için While Wend döngüleri yerine Do... Until döngüleri önerilir.

Örnek: Giriş Kontrolü .Aşağıdaki örnekte tipik olarak bir giriş kontrolü yapılmaktadır.

Kullanıcı belirtilen sayıları girmediği takdirde program kontrolü bu alanı geçmemektedir:

```
Dim Yanit As String
Do ' döngü başı
    Yanit = InputBox ("1,2,3 sayılarından birisini girin . ")
Loop Until Yanit >= 1 And Yanit <= 3
'döngü sonu
```

Örnek: Dosya sonuna kadar okuma

```
Dim SAYAC, OGR1SAYAC, OGR2SAYAC As Integer
Dim Vt As Database
Dim Tablo as Recordset
```

```
Set Vt = DBEngine .Workspaces (0).Databases ( "C: \DATA\OKUL.MDB")
Set Tablo = Vt.OpenRecordset ( "OGRENCIHAREKET")
Do
    If Tablo!OGR = "OG01" Then OGR1SAYAC = OGR1SAYAC + 1
    If Tablo!OGR = "OG02" Then OGR2SAYAC = OGR2SAYAC + 1
    SAYAC = SAYAC + 1
    Tablo.MoveNext
Loop Until Tablo.EOF
MsgBox SAYAC & " " & OGR1SAYAC & " " & OGR2SAYAC
```

Bir Kosul Dogru Oluncaya Kadar Döngü

Bir Do...Loop Until döngüsündeki kosulu kontrol etmek için Until deyimini kullanmanın iki yolu vardır. Birincisinde kosul döngüye girmeden önce kontrol edilir. İkincisinde kosul döngünün sonunda kontrol edilir; bu yöntemde döngü birkez dönmüş olur.

Giriste kontrol:

```
Sub IlkUntil()
    Sayaç = 0
    Sayi = 20
    Do Until Sayi = 10
        Sayi = Sayi - 1
        Sayaç = Sayaç + 1
    Loop
    MsgBox "Döngü " & Sayaç & " kere döndü."
End Sub
```

Çıkışta kontrol:

```
Sub SonUntil()
    Sayaç = 0
    Sayi = 1
    Do
        Sayi = Sayi + 1
        Sayaç = Sayaç + 1
    Loop Until Sayi = 10
    MsgBox "Döngü " & Sayaç & " kere döndü."
End Sub
```

While ve Until Sistemi:

While Yapisi:

```
Sayaç = 1
Do While Sayaç <= 10
    MsgBox sayaç
    Sayaç = Sayaç + 1
Loop
Do
    Yanıt=InputBox ( "Parolayı girin")
Loop While Yanıt <> "XXX"
```

Until Yapisi:

```
Sayaç = 1
Do Until Sayaç > 10
    MsgBox sayaç
    Sayaç = Sayaç + 1
Loop
Do
    Yanıt=InputBox ( "Parolayı girin")
Loop Until Yanıt <> "XXX"
```

Döngüden Çıkis

Bir Do. ..Loop döngüsünden kosulsuz çıkmak için Exit Do deyimi kullanılır.

Yapisi: Exit Do

Asagidaki örnekte bir If ...Then ...Else deyimi içinde kosul test edilerek döngüden çıkılmakta böylece sonsuz döngü engellenmektedir:

```
Sub ÇikisÖrneği()
    Sayaç = 0
    Sayi = 9
    Do Until Sayi = 10
        Sayi = Sayi - 1
        Sayaç = Sayaç + 1
        If Sayi < 10 Then Exit Do
    Loop
    MsgBox "Döngü " & Sayaç & " kere döndü."
End Sub
```

NOT: Bir sonsuz döngüyü durdurmak için ESC ya da CTRL+BREAK tuşları kullanılır.

For...Next Deyimi

Bir grup deyimi belli sayıda çalıştırarak (tekrar ederek) bir döngü oluşturur. Örneğin birden ona kadar (on kere) su işlemi yap şeklinde. Özellikle bir döngü denetim değişkeninin kullanıldığı (sayaç) bu döngü yapısı iç içe döngülerin de yapılmasını sağlar. For...Next döngüleri özellikle dizilerin işlenmesinde de yaygın olarak kullanılır.

Yapısı:

```
For sayaç = başlangıç To bitis [ Step artırım ]
    [ifade bloğu]
[Exit For]
    [ifade bloğu]
Next [sayaç]
```

For...Next deyiminde kullanılan parametreler şunlardır:

Parametre	Açıklama
For	For..Next döngüsüne başlar.
sayaç	Döngünün kaç kere tekrar edeceğini belirten sayısal değer.
başlangıç	Sayacın ilk değeri.
To	Başlangıç ve bitis değerlerini artırmak için kullanılan sözcük.
bitis	Sayacın son değeri.
Step	Artırım sayısı. Belirtilmezse 1 sayılır.
Exit For	For..Next döngüsünü sona erdirir. Artırım değerini sayaca ekler.
Next	For..Next döngüsünü sona erdirir. Artırım değerini sayaca ekler.

Artırım değeri döngünün işleyişini aşağıdaki biçimde kontrol eder:

Artırım değeri	Döngünün işletilmesi koşulu
Pozitif ya da 0	sayaç <= bitis
Negatif	sayaç >= bitis

Döngü birkez işledikten sonra artırım değeri sayaç değerine eklenir. Eğer herhangi bir artırım değeri kullanılmadıysa o zaman sayaç bir artırılır.

Sayaç değeri bitiş değerinden büyük olduğunda döngü sona erdirilir ve programın kontrolü Next deyiminden itibaren devam eder.

NOT:

Eğer döngünün başlangıç değeri bitiş değerinden büyükse döngü çalışmaz. Ancak böyle bir döngü Step deyiminin düzenlenmesiyle yapılabilir. Next deyiminin ardında sayaç değişkenini belirtmek zorunda değilsiniz. Ancak özellikle iç içe döngülerde belirtmekte yarar var.

NOT:

Sayaç değişkenine döngü içinde yeniden atama yapmayın. Eğer yaparsanız sonsuz bir döngüye yol açabilirsiniz.

For..Next yapısı iç içe (nested) de kurulabilir. İç içe döngünün işletilmesinde; dıştaki bir döngünün bir sefer dönmesi içteki döngünün kendi sayısı kadar dönmesi anlamına gelir. Genellikle sıralama vb. gibi birbiriyle ilgili iç işlemleri olan döngülerde kullanılır:

İç-içe Döngü:

```
For I = 1 To 10
    For J = 1 To 10
        For K = 1 To 10
            ...
        Next K
    Next J
Next I
```

Örnek: Birden ona kadar sayıların toplamını alır:

```
Sub Button31_Click()
    Dim Toplam As Double
    Toplam = 0
    For I = 1 To 10
        Toplam = Toplam + I
    Next I
    MsgBox " 1'den 10'a kadar sayıların toplamı: " & Toplam
```

Bubble Sort (Köpük Sıralama)

Verilerin sıralanması programlamada sık karşılaşılan bir konudur. Çok sayıda sıralama yöntemi (algoritması) vardır. Bunlardan birisi de köpük sıralamadır.

Birden çok sayinin siralanmasi için geleneksel yöntemlerden birisi de köpük siralamadır. Köpük siralamadaki algoritma şöyledir: Siralanacak elemanlar bir dizi (array) olarak düzenlenir. Ardından birinci elemanla ikinci eleman birbiriyle karsilastirilir. Eger ikinci eleman birinciden küçükse birinciyle yer degistirilir. Bu islem böyle devam eder:

```
For r = 1 to n-1
    For J = r+1 to n
        if x(j) < x(r) Then Swap (x(j), x(r))
    Next J
```

```
Next r
```

Yukardaki ham kodda görüldüğü gibi köpük siralama iç-içe iki döngü ile yapılmaktadır. Distaki döngü r'den n-1 'e kadar dönerken, içteki döngü r+1 'den n'e kadar dönmektedir.

For...Next döngüsünde döngü degiskeni, döngünün kullaniminda önemli bir rol oynar. Kullanimlardan genellikle döngü degiskeni ile bir dizinin elemani ya da kayıt sayisi gibi degerler temsil edilir:

```
For i = 1 to 7
    Günler(i) = " "
Next i
```

Asagidaki döngü 10 kez çalışacak ve bilgisayardan 10 kez ses çıkartacaktır:

```
Sub Beeps()
    For x = 1 To 10
        Beep
    Next x
End Sub
```

Asagidaki kod ise kullanilabilir ekran yazi tiplerini listeler:

```
Private Sub Form_Click()
    Dim I As Integer
    For i = 0 To Screen.FontCount
        Print Screen.Fonts(i)
    Next
End Sub
```

Step Deyiminin Kullanimi

Step deyimi döngü degiskenin (sayacin) belirtilen degerde artmasini ya da azalmasini saglar.Örneğin döngü degiskenin 2,4,6 diye gitmesi.

Microsoft Visual Basic 6.0

```
Sub IkininToplami()  
    For j = 2 To 10 Step 2  
        Toplam = Toplam + j  
    Next j  
    MsgBox "Toplam: " & Toplam  
End Sub
```

Adımlama azalan biçimde de yapılabilir:

```
Sub Toplama1()  
    For Sayi = 16 To 2 Step -2  
        Toplam = Toplam + Sayi  
    Next Sayi  
    MsgBox "Toplam: " & Toplam  
End Sub
```

Dizileri İşlemek İçin Döngülerin Kullanılması

Özellikle çok boyutlu dizilerin elemanlarına erismek, onlara değer atamak için For... Next döngüleri idealdir. Aşağıdaki örnekte iki boyutlu bir dizinin elemanları sıfırlanmaktadır:

Örnek:

```
Dim I As Integer, J As Integer  
Static MatrisA (1 To 10, 1 To 10) As Double  
For I = 1 To 10  
    For J = 1 To 10  
        MatrisA( I , J) = 0  
    Next J  
Next I
```

Örnek: Bir ListBox içindeki çift elemanları silme

```
For i = 0 To List1.ListCount  
    For j = i + 1 to List1.ListCount  
        If List1.List(i) = List1.List(j) Then List1.RemoveItem i  
    Next j  
Next i
```

For...Next Döngülerinden Çıkis

Bir For...Next deyiminden çıkmak için Exit For devimi kullanılır.

Yapisi: Exit For

Tamsayı Degiskenler Ile Daha Hizli Döngüler

Integer degiskenler Variant bir tipli bir degiskene göre daha az bellek yeri harcarlar. Küçük bir programda bu fark o kadar önemli olmayabilir. Ancak büyük programlarda ya da yapılan işlem sayısı arttıkça bu önem kazanır:

```
Dim HizliSay As Integer ' Birinci durum. Integer kullan.  
For HizliSay = 0 to 32766  
Next HizliSay  
Dim YavasSay As Variant ' İkinci durum. Variant kullan.  
For YavasSay = 0 to 32766  
Next YavasSay
```

Yukarıdaki örneklerden birinci olanı daha hızlı çalışır. Bununla birlikte HizliSay degiskeni 32,767'yi geçerse hata oluşur. Bunun önlemek için HizliSay degiskeni Long olarak tanımlanabilir.

Karar Yapilari

Program denetimi yapılarından birisi de karar vermektir. Kararlar özellikle belli koşullara göre yapılacak işlemlerin seçilmesini sağlarlar.Örneğin değişik ücret düzeylerindeki insanların değişik vergi oranlarıyla vergilendirilmesi.

Visual Basic'te karar yapıları olarak If...Then...Else ve Select...Case deyimleri kullanılır. If...Then...Else deyimi belli bir deyimi ya da bir blok deyimi bir koşula bağlı olarak işletmeyi sağlar. Koşulun doğru olması True, yanlış olması False anlamına gelir. Select Case deyiminde ise özellikle bir karar degiskeninin aldığı değerlere göre değişik işlemlerin yapılması sağlanır.

Karar yapılarının temelinde değerler arasındaki ilişkiler yatar. İlişkiler değerler arasındaki operatörlerle kurulur.

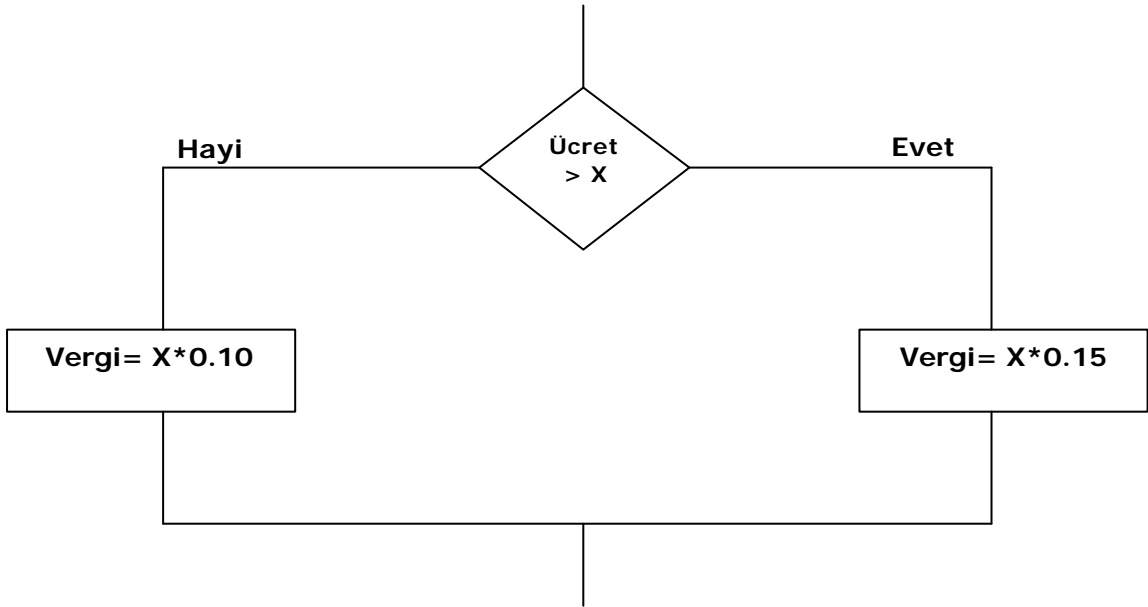
İlişki operatörleri

Operatör	Anlami
<	Küçüktür
<=	Küçük esittir
>	Büyüktür
>=	Büyük esittir
<>	Esit değildir
=	Esittir

And	Ve
Or	Veya - Ya da
Not	Değil

If...Then...Else Deyimi

Klasik olarak kullanılan karar mekanizmasıdır. Eğer koşul sağlanıyorsa (if) bunu yap; yoksa sunu yap (else) gibi. Aşağıdaki akış şemasında; Ücret değişkenine bağlı olarak gelir hesabının iki ayrı yolla yapılması yer almaktadır:



Sekil 4.17 If deyimi ile karar verme

If deyimi belli bir ifadenin değerine göre (koşul) bir deyimin isletilmesini sağlar. If deyiminin basitten karmaşığa doğru değişik kalıpları vardır:

Yapı: (1)

If koşul1 Then işlem

Yapı: (2)

```
If koşul Then
    [islemi1]
Else
    [islem2]
End If
```

Yapi: (3)

```

If kosul1 Then
    [islem1]
[ElseIf kosul2 Then
    [islem2] ]
...
[Else
    [islem-n]
End If

```

If-Then-Else yapısının birinci kullanım biçiminde genellikle basit ifadelerle karşılık gelen ifadeler çözülür. Kosul doğruysa (True) Then'den sonraki deyim işletilir, değilse bu deyimler atlanır. İkinci kullanım biçiminde ise; daha karmaşık hesaplamalara ve kararlara karşılık gelen ifadeler çözülür. Bu yapıda kosul doğruysa islem1 yerine getirilir, değilse Else'den sonra belirtilen işlemler yerine getirilir. If-Then-Else yapısı içinde kullanılan parametreler:

Parametre Açıklama

If	If yapısını başlatır.
Kosul	Yerine getirilecek ifade; Doğru ya da Yanlış sonucunu, verecek sayısal ya da karakter ifade.
Then	Kosul sağlandığında yapılacak işlemi belirtir.
Else	Kosulun sağlanmadığı durumlarda yapılacak işlemleri belirtir. Else deyimi kullanılmazsa; bir sonraki deyim işletilmesine geçilir.
Else If	Kosul1'in sağlanmadığı durumlarda diğer bir kosulu belirten sözcük.
End If	If-Then-Else yapısını bitirir.

If bloğunun işleyişinde; önce kosul test edilir. Eğer kosul sağlanıyorsa (True) o zaman Then'den sonra belirtilen deyimler işletilir. Eğer kosul sağlanmıyorsa (False) o zaman Else kısmında bulunan işlemler yerine getirilir.

Else If'li olan If yapısında ;If bloğun işleyişinde önce kosul test edilir. Eğer kosul sağlanıyorsa (True) o zaman Then'den sonra belirtilen deyimler işletilir. Eğer kosul sağlanmıyorsa ya da sağlanırsa o zaman Else If de bulunan kosul ele alınır. Eğer bu kosul doğru ise onun Then kısmı işletilir. Else If kosullarında herhangi biri sağlanmıyorsa o zaman; izleyen Else deyimi işletilir.

Kosulun ardından birden çok deyim kullanabilir: Bu deyimler aynı satırda bulunmalıdır ve aralarında iki nokta olmalıdır:

```
If Deger1 > 3 Then Deger1 = Alacak + Borç : Deger1 + Alacak
```

Örnek: Ekrandan alınan üç sayının en büyüğünü bulan program. Aşağıdaki örnekte; Ekrandan Inputbox ile alınan üç sayının en büyüğü bulunur:

```
Private Sub Command1_Click()  
    Dim Sayi1, Sayi2, Sayi3 As Double  
    Sayi1 = InputBox ("Birinci degeri giriniz")  
    Sayi2 = InputBox ("Ikinci degeri giriniz")  
    Sayi3 = InputBox ("Üçüncü degeri giriniz")  
    If Sayi1 > Sayi2 Then  
        If Sayi1 > Sayi3 Then MsgBox "Sayi1 Enbüyük"  
        Else If Sayi3 > Sayi2 Then MsgBox "Sayi3 Enbüyük"  
        Else  
            MsgBox "Sayi2 Enbüyük"  
        End If  
    End Sub
```

Kosul Doğruysa Deyimin İletilmesi

En basit biçimden bir If..Then yapısı koşulun doğru olması durumunda deyimın çalıştırılmasını sağlar. Burada If..Then deyimini ve iletilecek deyim bir satır olarak düzenlenir.

```
Sub Tarihal()  
    Tarih = #12/2/98#  
    If Tarih < Now Then Tarih = Now  
End Sub
```

Eğer daha fazla karar verilecekse If...Then...Else yapısı kullanılır. Bu yapı çok satırlı olarak düzenlenir:

```
Faiz = Inputbox("faiz ?")  
If Bakiye > 1000000 Then  
    Oran = 0.07  
Else  
    Oran = 0.05  
End If
```

Kosul Dogru Degilse Deyimin Isletilmesi

If...Then...Else deyimi iki blok halinde kullanilarak; kosul dogru ise bunlari, degilse sunlari diye kodlanabilir:

```
Sub Mesajver (Deger as Long)
  If Deger = 0 Then
    Etiket.ForeColor = vbRed
    Etiket.Font.Bold = True
    Etiket.Font.Italic = True
  Else
    Etiket.ForeColor = vbBlack
    Etiket.Font.Bold = False
    Etiket.Font.Italic = False
  End If
End Sub
```

Birinci Kosulun False Olmasinin Ardindan Ikinci Bir Kosulu Test Etme

If...Then...Else yapisinin çok sayıda kullanimindan birisi de birinci kosulun False olmasinin ardindan ikinci bir kosulu test etmektir. Asagidaki örnekte ise göre ücret belirlenmektedir:

```
Function Ikramiye(Performans, Ücret)
  If Performans = 1 Then
    Ikramiye = Ücret * 0.1
  ElseIf Performans = 2 Then
    Ikramiye = Ücret * 0.09
  ElseIf Performans = 3 Then
    Ikramiye = Ücret * 0.07
  Else
    Ikramiye = 0
  End If
End Function
```

Örnek: Su parasi hesabi

Su parasi hesabi ya da KDV kesintisi hesabi ya da benzer fiyatlandirmalar belli bir tarifeye göre bir hesaplamanin yapilarak bir islemin yapılmasını ve sonucun hesaplanmasını gerektirirler. Örnekte asagidaki tablo kullanilarak ekrandan girilen mektreküp degerlerine göre ne kadar TL'lik su harcandigini hesaplayan bir program yazilacaktır.

Tüketim	Metreküp Fiyatı
0-30	20.000 TL
31-100	40.000 TL
101-sonsuz	100.000 TL

```
Private Sub Command0_Click()  
    Dim Tüketim As Double  
    Dim Para As Double  
    'Deger alma  
    Tüketim = InputBox("Tüketim Degerini girin")  
    If Tüketim > 0 And Tüketim < 31# Then  
        Para = Tüketim * 20000  
    ElseIf Tüketim > 30# And Tüketim < 101# Then  
        Para = (30# * 20000#) + ((Tüketim - 30#) * 40000#)  
    ElseIf Tüketim > 101# Then  
        Para = (30# * 20000#) + (70# * 40000#) + ((Tüketim - 100#) *  
100000#)  
    End If  
    MsgBox Str(Para)  
End Sub
```

If..Then..Else Yapisindan Çikis

Istenildiginde If..Then...Else yapisi içinde çikilabilir. Bu islem için Exit If deyimi kullanilir.

Yapisi: Exit If

Select Case Deyimi

If..Then...Else gibi karar verilmesini; durumlara göre farkli islemlerin yapılmasını saglar. Select Case yapisi bir kosul ya da degisken için olasi degerlerin degerlendirilmesini saglar. Diger bir deyişle bir ifadenin degerine göre bir ya da daha çok deyim grubunun (ifade blogunun) isletilmesini saglar. Select Case yapisi çok sayıda islemin bir test degerine göre isletilmesini saglar. Select Case yapisinin temelinde bir test degeri vardır. Bu degerin degisik seçenekleri ve islem bloklari yer alır.

Yapisi (basit):

```
Select Case test degeri  
    Case deger1  
        deyim blok 1  
    Case deger2  
        deyim blok 2
```


End Select

Yapisi:

```
Select Case test ifadesi
  [Case ifade listesi-1
    [ifade blogu-1] ]
  [Case ifade listesi-2
    [ifade blogu-2] ]
  [Case Else
    [ifade blogu-n] ]
End Select
```

Select Case deyiminin parametreleri şunlardır:

Parametre	Açıklama
Select Case test ifadesi	Select Case karar yapısının başlangıcı. Sayısal ya da karakter bir ifade. Test ifadesi, ifade listesindeki ifadeler uyarsa (aynisi ise); ifade bloğunda yer alan deyimler işletilir.
Case	İfade listesindeki ifadenin test ifadesi ile uyusması durumunda işletilecek deyimleri içeren bir deyim grubu.
ifade listesi	Virgülle ayrılmış ifade ve değerler.
ifade blogu	Bir ya da daha çok Visual Basic deyimini.
Case Else	Case ile belirtilen ifadelerden hiçbirisinin test ifadesi ile uyusmaması durumunda işletilecek deyimleri belirtir.
End Select	Select Case yapısını sona erdirir.
To	Bir degerden-bir degere veri alanini gösterir.
Is	Bir karsilastirma isleci ile birlikte kullanilir.

Select Case yapısı içindeki ilk deyim Select Case'dir. Ardından bir değeri olan test değişkeni yer alır. Bu değer bir sayı, string ya da fonksiyon olabilir. Ardından koşulların belirtildiği Case deyimleri yer alır. Case deyimini test değeri ile karşılaştırılacak değeri belirtir. Eğer Case ile belirtilen değer test değerine eşitse o zaman belirtilen deyim ya da deyim blogu işletilir. End Case deyimini ile Select Case yapısı sona erdirilir.

Örnek: Ücret hesabi

```
Ücret = Val(Text1.Text)
Select Case Ücret
  Case 1000 to 1100
```

Microsoft Visual Basic 6.0

```
Vergi = Ücret * 0.20
Case 1101 to 1200
    Vergi = Ücret * 0.25
Case 1201 to 1300
    Vergi = Ücret * 0.30
Case Else
    Vergi = Ücret * 0.40
End Select
```

Özellikle ücret bordrosu gibi değişik fiyatlandırma işlemlerinde Select Case kullanımı idealdir.

```
IsteneMiktar = Val (Text1 .Text)
Select Case IsteneMiktar
    Case Is < 0
        MsgBox "Siparis sifir olamaz", vbExclamation
        Exit Sub
    Case 1, 2, 3
        Indirim = 0
    Case 4 To 9
        Indirim = 0.05
    Case 10 To 49
        Indirim = 0.1
    Case Is > 50
        Indirim = 0.15
End Select
```

Yukarıdaki örnekte siparis miktarının belli değerlerine göre belli işlemler yapılmaktadır. Ancak bu değerlerin dışındaki bir değer olduğunda ne olacak? Bu durumda Else Case durumu kullanılır. Else Case durumu Case koşullarının dışındaki koşulu belirtir.

```
IsteneMiktar = Val (Text1 .Text)
Select Case IsteneMiktar
    Case Is < 0
        MsgBox "Siparis sifir olamaz", vbExclamation
        Exit Sub
    Case 1, 2, 3 Indirim = 0
    Case 4 To 9 Indirim = 0.05
    Case 10 To 49 Indirim = 0.1
    Case Is > 50 Indirim = 0.15
    Case Else
        Indirim = 0.20
End Select
```

Case deyiminde çok sayıda değer ve belli bir değer aralıkları da kullanılabilir:

Case 10 To 20, 40 To 80, 125, Is > Deger1
Case "AYSE" To "MEHMET"

NOT:

Tek bir degiskenin olasi degerlerine göre degisik seçeneklerin degerlendirilmesi söz konusu ise o zaman Select Case yapisi tercih edilmelidir. Birden çok degiskenin aldigi degisik degerlere göre bir karar yuaglı çözülecekse If yapisi daha uygun olabilir.

Örnek: Klavyeden girilen karakterin "küçük mü/büyük mü?" testi. Asagidaki örnekte; kullanicidan alınan deger aralık olarak kontrol edilmekte ve küçük ya da büyük harf mi, yoksa rakam mi oldugu belirlenmektedir.

```
Sub Button31_Click()  
    Dim Mesaj, Deger ' Degiskenler  
    Deger = InputBox ("Bir harfe ya da rakama basiniz. ")  
    Select Case Deger  
        Case "A" To "Z"  
            Mesaj = "A ile Z arasinda bir harfe bastiniz"  
        Case "a" To "z"  
            Mesaj = "a ile z arasinda bir harfe bastiniz"  
        Case 0 To 9  
            Mesaj = "0 ile 9 arasinda bir rakama bastiniz"  
        Case Else  
            Mesaj = "Istenmeyen bir harfe bastiniz"  
    End Select  
    MsgBox Mesaj  
End Sub
```

Select Case deyimi ElseIf ile kullanılan If..Then...Else yapısına bir alternatiftir. Özellikle bir degiskenin aldigi çeşitli degerlere göre degisik islemlerin yapılmasını sağlar. Select Case deyimi ilk basta degerlendirme yapar. Ardından degisik kosullara göre degisik islemlerin yapılmasını sağlar.

```
Function Ikramiye (Performanse, Ücret)  
    Select Case Performans  
        Case 1  
            Ikramiye = Ücret * 0.1  
        Case 2, 3  
            Ikramiye = Ücret * 0.09  
        Case 4 To 6  
            Ikramiye = Ücret * 0.07  
        Case Is > 8  
            Ikramiye = 100  
        Case Else
```

Microsoft Visual Basic 6.0

```
        Ikramiye = 0
    End Select
End Function
```

Örnek: Su Hesabi. If yapısında örneklenen su hesabi aslında Case yapısında daha iyi uyar. Örnekte aşağıdaki tablo kullanılarak ekrandan girilen metreküp değerlerine göre ne kadar TL'lik su harcandığını hesaplayan bir program yazılacaktır.

Tüketim	Metreküp Fiyatı
0-30	20.000 TL
31-100	40.000 TL
101-sonsuz	100.000 TL

```
Private Sub Command1_Click()
    Dim Tüketim As Double
    Dim Para As Double
    'Değer alma
    Tüketim = InputBox("Tüketim Değerini girin")
    Select Case Tüketim
        Case 0 To 30#
            Para = Tüketim * 20000
        Case 31 To 101#
            Para = (30# * 20000#) + ((Tüketim - 30#) * 40000#)
        Case Else
            Para = (30# * 20000#) + (70# * 40000#) + ((Tüketim - 100#)
* 100000#)
    End Select
    MsgBox Str(Para)
End Sub
```

NOT:

Select Case yapısında; iç-içe Select Case deyimi de kullanılabilir. Bu durumda her bir Select Case deyiminin kendine ait bir End deyimi olmalıdır.

Aşağıdaki örnekte ise ekrandan alınan bir değerin tipi belirlenir:

```
Private Sub command1_Click()
    Dim Deger, Kontrol
    Deger = InputBox("bir değer girin")
    Kontrol = VarType (Deger)
    Select Case Kontrol
        Case 0: MsgBox "Bos"
        Case 1: MsgBox "Null"
        Case 2: MsgBox "Tamsayı"
        Case 3: MsgBox "Long Tamsayı"
```

```
Case 4: MsgBox "Single-precision floating-point sayı"  
Case 5: MsgBox "Double-precision floating-point sayı"  
Case 6: MsgBox "Para birimi"  
Case 7: MsgBox "Tarih"  
Case 8: MsgBox "String"  
Case 9: MsgBox "Nesne"  
Case 10: MsgBox "Hata"  
Case 11: MsgBox " ikili"  
Case 12: MsgBox "variant"  
Case 13: MsgBox "data access object-veritabanı nesnesi"  
Case 14: MsgBox "onlu sayı"  
Case 17: MsgBox "Byte"  
Case 36: MsgBox "User-defined veri içerir"  
End Select  
End Sub
```

Aşağıdaki örnekte ise metin bilgiler karşılaştırılarak seçim yapılmaktadır:

```
Select Case İşlem  
Case "Topla": Sonuç.Text = Val(Sonuç.Text) + Geçici  
Case "Çıkar " : Sonuç.Text = Geçici - Val(Sonuç.Text)  
Case "Böl": Sonuç.Text = Geçici / Val(Sonuç.Text)  
Case "Çarp": Sonuç.Text = Val(Sonuç.Text) * Geçici  
End Select
```

Select Case Yapısından Çıkış

Select yapısından çıkmak için Exit Select deyimi kullanılır.

Yapısı: Exit Select

GoTo Deyimi

Bir procedure içinde belirtilen bir satıra (kosulsuz olarak) sarmayı sağlar.

Yapısı:

GoTo (etiket | satır numarası)

GoTo deyiminin parametreleri:

Parametre

Açıklama

GoTo

Belirtilen yere sarmayı sağlar.

Microsoft Visual Basic 6.0

etiket	İsletilecek program satirlarini belirtir. Etiket bilgisi alfabetik bir karakter ile baslamalıdır ve iki nokta üst üste (:) ile bitmelidir. Etiket bilgisi ençok 40 karakter olabilir.
satir numarası	İsletilecek program satirlarini belirtir. Satir numarası ençok 40 karakter olmak üzere 0 ve 9 rakamlarından oluşur. Satir numarasının ardından : isaretinin kullanılmasına gerek yoktur.

Goto deyimi ve GoSub deyimleri programda gerekmedikçe kullanılmaması gereken komutlardır. Bu komutlar program kontrolünü (okunmasını, hataların bulunmasını vb.) zorlastırırlar.

NOT:

Programcının Goto ve Gosub deyimlerinin yerine Do..Loop, For..Next, If..Then ve Select..Case gibi yapıları kullanması daha İyi olur.

Örnek: Goto Deyimi

```
Sub Button31_Click()  
    Dim Sayı 'Degiskenler tanımlanır  
    Sayı = InputBox ("1-100 arasında üir sayı giriniz.")  
    GoTo İstem 'diger bir kisima sapma  
    ' bu kisimda yer alan deyimler atılanır.  
İstem: ' bir program kisimi  
    Sayı = Sayı / 10  
    MsgBox "Girilen sayının onda biri :" & Sayı  
End Sub
```

Exit Deyimi

Do...Loop, For...Next loop, döngüleri, bir Function procedure ya da bir Sub procedure'dan çıkışı sağlar.

Yapısı:

Exit { Do | Function | Sub }

Exit deyiminin kullanım biçimleri şöyledir:

Deyim

Açıklama

Exit Do

Loop döngüsünden çıkar. Exit Do deyimi eger bir iç döngü (nested) içinde kullanılırsa o zaman bir önceki düzeydeki döngüye döner.

Exit For	For..Next döngüsünden çıkisi sağlar. Exit For deyimi eger bir iç döngü (nested) içinde kullanilrsa o zaman bir önceki düzeydeki döngüye döner.
Exit Function	Fonksiyonu sona erdirir ve program kontrolünü ; fonksiyonu çağiran deyimden sonraya geçirir.
Exit Sub	Sub procedure'i sona erdirir ve program kontrolünü; procedure'i çağiran deyimden sonraya geçirir.

NOT:Exit deyimi ile End deyimi karistirilmamalıdır. Exit deyimi bir fonksiyondan, bir procedure'da ve bir döngü içinden çıkisi sağlar. End deyimi ise Visual Basic procedure'ini dolayisi ile bir programini sona erdirebilen bir komuttur.

```
Sub Button31_Click()
    Dim Yanit 'Degisken tanimlanir.
    Do
        Yanit = InputBox("1,2,3 sayilarindan birisini girin.")
        If Yanit >= 1 And Yanit <= 3 Then ' Tamam
            MsgBox "Tamam"
            Exit Do ' Döngüden çık.
        End If
        Loop 'döngüye devam
    End Sub
```

End Deyimi

Bir Visual Basic procedure'ini ya da blogunu sona erdirir.

Yapisi:

```
End [ { Function | If | Select | Sub | Type}]
```

End deyimi asagidaki biçimlerde kullanilabilir:

Deyim	Açıklama
End Function	Function'u sona erdirir. End Function deyimi bir fonksiyon olarak fonksiyonunun sonuna eklenir.
End If	If-Then blogunu sona erdirir.
End Select	Select Case blogunu sona erdirir.
End Sub	Sub procedure'ini sona erdirir. Yeni bir Sub procedure'ina baslanacagi zaman otomatik olarak Sub procedure'in sonuna eklenir.
End Type	Kullanici tanimli veri tanimini sona erdirir.
End	Bir Visual Basic programini sona erdirir.

Operatörler

Operatörler (işleçler) hem karar vermede ilişkileri oluştururlar hem de aritmetik işlemlerde kullanılırlar. Birçok operatörün bir arada kullanılması durumunda işlem sırası operatörlerin öncelik sırasına göre düzenlenir. Ancak ifadelerde parantezlere yer vermek işlem sırasının programcı tarafından kontrol edilmesini sağlar.

Operatörler ve öncelikleri

Aritmetik		Karsılaştırma
Mantıksal		
Üs alma (^)	Esitlik (=)	Not
Olumsuz (-)	Esitsizlik (<>)	And
Çarpma ve bölme (* , /)	Küçüktür (<)	Or
Tamsayı bölme (\)	Büyüktür (>)	Xor
Modülo Aritmetik (Mod)	Küçük ya da esittir (<=)	Eqv
Toplama ve çıkarma (+, -)	Büyük ya da esittir (>=)	Imp
String birleştirme(&)	Like	Is

İfade içindeki operatörlerin esit öncelikli olmaları durumunda; ifade soldan sağa doğru değerlendirilir.

^ Operatörü

Bir değerin ya da ifadenin belirtilen sayı kadar kuvvetini alır.

Yapısı: Sonuç = Sayı ^ Üs

Örnek: Operatörün kullanımı

Deger = 3 ^ 3 ' deger'in degeri 9 olur.
Deger = 3 ^ 3 ^ 3 ' deger'in degeri 27 olur.
Deger = (-3) ^ 3 ' deger'in degeri -9 olur.

+ Operatörü

İki sayıyı toplamak için kullanılır.

Yapısı: Sonuç = Ifade1 + Ifade2

Ifadelerin yerinden herhangi bir deger bulunabilir.

Örnek: Operatörün kullanımı

Sonuç = 2 + 2 'sonuç 4 olur.

+ operatörü iki string verinin birlestirilmesi için de kullanilir. Iki string bilginin birlestirilmesinde daha açık olmak için & operatörü kullanılmalidir. Ancak istenirse + operatörü de kullanılabilir.

+ operatörünün kullaniminda; eger toplanan verilerden herhangi birisi Variant veri tipine sahipse o zaman toplama yerine birlestirme islemi yapilir.

Eger Islem

Iki ifade de sayisal ise (Byte, Boolean, Integer, Double, Date, Currency) Toplama Long, Single,

Iki ifade de String ise Birlestirme (Concatenate)

Ifadelerden birisi sayisal, digeri Variant ise Toplama

Ifadelerden birisi String, digeri Variant ise Birlestirme

Iki ifade de Variant ise Ikisinin de içeriği String ise Birlestirme islemi, aksi takdirde toplama islemi. Ifadelerden birisi sayisal, digeri String ise "Type mismatch" hatasi

Ifadelerden birisi Null ise Null

Örnek: Operatörün kullanımı

Sonuç = 2 + 2 'Sonuç'un degeri 4 olur.

Sonuç = "2" + 2 'Sonuç'un degeri 4 olur.

Sonuç = "2" + "2" 'Sonuç'un degeri 22 olur.

Örnek: Hesap makinesi; hesap makinesi programinda rakam tuslarına bastıkça rakamı üretmek için + operatöründen yararlanilir:

```
Private Sub Command10_Click()  
    Sonuç.Text = Sonuç.Text + "7"
```

```
End Sub
```

```
Private Sub Command11_Click()  
    Sonuç.Text = Sonuç.Text + "8"  
End Sub
```

Is Operatörü

İki nesne değişkeninin karşılaştırılmasını sağlar.

Yapısı: Sonuç = nesne1 Is nesne2

Eğer iki nesne de aynı nesneyi işaret ediyorsa o zaman sonuç True olur. Eğer değilse False sonucu doğar.

Örnek: Operatörün kullanımı

```
Set Nesne1 = Nesne2  
Sonuç = Nesne1 Is Nesne2 ' Sonuç değeri 0 olur.
```

Like Operatörü

İki String değeri karşılaştırmak için kullanılır.

Yapısı: Sonuç = String Like String ifade
Like deyimi iki string ifadeyi karşılaştırır. Eğer iki ifade birbirine uyuyorsa sonuç True olarak, uymuyorsa False olarak çıkar. Like deyiminin çalışması Modüllerin Declaration kısmında yazılan **Option Compare** devimine bağlıdır. Varsayılan karşılaştırma karakterlerin ikili (binary) değerinin sıralamasına göre yapılır. Buna **Option Compare Binary** denir.

İkili karşılaştırma: $A < B < Z < a < b < z$

Eğer **Option Compare Text** seçilirse o zaman küçük ve büyük harfe duyarlı karşılaştırma işlemi yapılır.

Text karşılaştırma: $(A=a) < (B=b) < (Z=z)$

Örnek: Karşılaştırma işlemi

```
Sonuç = "ahmet" Like "a*" ' True döndürür.  
Sonuç = "H" Like "[A-Z]" ' True döndürür.  
Sonuç = "XYZ" Like "X?Z" ' True döndürür.  
Sonuç = "WY£" Like "X?Z" ' False döndürür.
```

5-FONKSİYONLAR

Tüm programlama dillerinde olduğu gibi, Visual Basic'te de çeşitli işlemleri gerçekleştirmek için fonksiyonlar kullanılır.

Visual Basic programlama dili grafik elamanların yanı sıra metotlar, deyimler, olaylar ve fonksiyonlarla birlikte kullanılır. Fonksiyonlar belli işlemleri yaparlar ve bir değeri döndürürler. Bu nedenle bir değere esitlenerek kullanılırlar.

Fonksiyonlar yaptıkları işlemlere göre belli sınıflara ayrılırlar. Bunlar;

1. String Fonksiyonlar
2. Matematiksel Fonksiyonlar
3. Değişken Kontrol Fonksiyonları
4. Disk İşlem Fonksiyonları
5. Dosyalama fonksiyonları
6. Format Dönüşüm Fonksiyonları
7. Yükleme Sonlandırma Fonksiyonları

A. STRING FONKSİYONLAR

Tüm programlama dillerinde olduğu gibi, Visual Basic'te de çeşitli işlemleri gerçekleştirmek için fonksiyonlar kullanılır. Aşağıda matematikle ilgili olan fonksiyonlar verilmiştir.

Asc() Fonksiyonu

String bir değişkenin ilk karakterinin veya verilen herhangi bir karakterin ASCII kodunu verir.

Kullanımı;

Sonuc = Asc (Karakter)

Karakter : ASCII kodu bulunacak karakter veya string değişken

Sonuc : Argüman olarak verilen karakterin ASCII kodunun karşılığı olan integer sayı

Örnek;

```
Private Sub Form_Load()  
Dim A_Kodu  
A_Kodu = Asc("A") ' A_Kodu = 65.  
A_Kodu = Asc("a") ' A_Kodu = 97.  
A_Kodu = Asc("Apple") ' A_Kodu = 65.  
End Sub
```

Chr() Fonksiyonu

Bu fonksiyon Chr fonksiyonunun tersine ASCII kode verilen karakteri kendisini verir.

Kullanimi;

Sonuc = Chr (ASCII kod)

ASCII Kod : Kendisi bulunacak karakterin ASCII kodu

Sonuc : ASCII kodu verilen karakterin saklandığı sonuç değeri

Örnek;

```
Private Sub Form_Load()  
Dim Char  
Char = Chr(65) ' Char = A.  
Char = Chr(62) ' Char = >.  
End Sub
```

IsArray() Fonksiyonu

Argüman olarak verilen değişkenin dizi olup olmadığını verir. Eğer değişken dizi ise dönüş değeri True, değilse False olur.

Kullanimi;

Sonuc = IsArray (Değişken)

Değişken : Dizi kontrolü yapılacak değişken

Örnek;

```
Private Sub Form_Load()  
Dim MyArray(1 To 5) As Integer  
Dim a, MyCheck  
MyCheck = IsArray(MyArray) ' MyCheck = True.  
MyCheck = IsArray(a) ' MyCheck = False  
End Sub
```

Instr() Fonksiyonu

Verilen string değişken veya bilgi içinden bir başka string veya karakteri arar. Eğer aradığınız karakter veya string aranan string içinde yoksa fonksiyon 0 değerini üretir, fakat varsa aranan string veya karakterin bulunan string içinde kaçınca karakter olduğunun değerini üretir. Fonksiyon ayrıca büyük/küçük harf ayrımı yapmaktadır. Fonksiyonun ilk argümanı olan başlangıç değeri verilmezse bu değer 1 olarak kabul edilir.

Kullanımı;

Sonuc = Instr (Baslangiç , String , Aranan)

Baslangiç : Aramaya stringin kaçinci karakteriden baslanacaginin degeridir

String : İçinde arama yapılacak string veya karakter

Aranan : String içinde aranan karakter veya string

Sonuc : Aranan stringin sirasini veren sonuç degeridir

Örnek;

```
Private Sub Form_Load()  
    Dim TaranacakString, Bulunacak, MyPos  
    TaranacakString = "Veli Akçakaya"  
    Bulunacak = "A"  
    MyPos = Instr(1, TaranacakString, Bulunacak)  
    'MyPos = 6  
End Sub
```

Len () Fonksiyonu

Argüman olarak verilen string veya Variant bir degiskenin karakter sayisini yani uzunlugunu verir. Argüman olarak verilen degisken variant veya string degisken yapisindan farkli bir yapıya sahip ise Visual Basic'in 6 numarali (Overflow) hatasi ortaya çıkar ve program hemen sonlandırilir.

Kullanımı;

Sonuc = Len (Degisken)

Degisken : Uzunlugu yani kaç karakterden olustugu bulunacak degisken

Sonuc : Argüman olarak verilen degiskenin uzunlugu

Örnek;

```
Private Sub Form_Load()  
    Dim vrt_Degis, Str_Degis, i, j  
    vrt_Degis = 653819  
    Str_Degis = "Veli AKÇAKAYA"  
    i = Len(vrt_Degis) ' i = 6  
    j = Len(Str_Degis) ' j = 13  
End Sub
```

Left(), Left\$() Fonksiyonu

String degiskenin en solundaki belirtilen sayidaki karakterleri verir.

Kullanimi;

Sonuc = Left (Degisken, Uzunluk)

Degisken : Belirtilen karakteri seçilecek olan degisken

Uzunluk : Seçilecek karakter sayısı

Örnek;

```
Private Sub Form_Load()  
Dim AnyString, MyStr  
AnyString = "Hello World"  
MyStr = Left(AnyString, 1) ' MyStr = "H".  
MyStr = Left(AnyString, 7) ' MyStr = "Hello W".  
End Sub
```

Lcase(), Lcase\$() Fonksiyonu

Argüman olarak verilen string degiskenin içerdigi tüm büyük harfleri küçük harfe dönüştürür. Çogu programlama dilindeki ayni görevi gören fonksiyon gibi bu fonksiyonda türkçe karakterleri küçük harfe çeviremez.

Kullanimi;

Sonuc = Lcase (Degisken)

Degisken : Küçük harfe dönüştürülecek string karakter

Örnek;

```
Private Sub Form_Load()  
Dim UpperCase, LowerCase  
UpperCase = "Hello World "  
LowerCase = LCase(UpperCase)' LowerCase = "hello world "  
End Sub
```

Ltrim,Trim, Rtrim Fonksiyonu

Bu komutlar string degisken içindeki en sol ve en sagdaki bosluklari kaldırır.

Trim : En sag ve soldaki bosluk karakterlerini kaldırır

Ltrim : En soldaki bosluk karakterlerini kaldırır

Rtrim : En sagdaki bosluk karakterlerini kaldırır

Kullanimi;

Sonuc = Trim (Degisken)

Degisken : Bosluk karakterleri kaldırılacak degisken

Örnek;

```
Private Sub Form_Load()
```

```

Dim Str, TrimS
Str = " Veli "
TrimS = LTrim(Str) 'TrimS = "Veli "
TrimS = RTrim(Str) ' TrimS = " Veli"
TrimS = LTrim(RTrim(Str)) ' TrimS = "Veli"
TrimS = Trim(Str) ' TrimS = "Veli"
End Sub

```

Mid, Mid\$ () Fonksiyonu

Argüman olarak verilen string degisken içinden baslangıç karakterinden itibaren belirtilen sayıdaki karakterleri sonuç degeri olarak üretir.

Kullanimi;

Sonuc = Mid (String, Baslangıç, Sayı)

String : Seçilecek karakter sayısı

Baslangıç : Seçilecek string degisken içindeki baslangıç karakteri

Sayı : Seçilecek karakter sayısı

Sonuc : Fonksiyonun ürettiği string yapıdaki sonuç degeri

Örnek;

```

Private Sub Form_Load()
Dim MyString, FirstWord, LastWord, MidWords
MyString = "Mid Function Demo"
FirstWord = Mid(MyString, 1, 3) 'FirstWord = "Mid"
LastWord = Mid(MyString, 14, 4) 'LastWord = "Demo".
MidWords = Mid(MyString, 5) 'MidWord = "Function Demo".
End Sub

```

Right(), Right\$() Fonksiyonu

String degiskenin en sagındaki belirtilen sayıdaki karakterleri verir.

Kullanimi;

Sonuc = Right (Degisken, Uzunluk)

Degisken : Belirtilen karakteri seçilecek olan degisken

Uzunluk : Seçilecek karakter sayısı

Örnek;

```

Private Sub Form_Load()

```

String degiskenin en sagındaki belirtilen sayıdaki karakterleri verir.

```
Dim AnyString, MyStr
AnyString = "Hello World"
MyStr = Right(AnyString, 1) ' MyStr = "d"
MyStr = Right(AnyString, 7) ' MyStr = "o World"
End Sub
```

Space, Space\$ () Fonksiyonu

Bu komutta yukarida açıklanan Tab ve Spc komutu gibi istenilen sayıda bosluk bırakır. Yukarıdaki fonksiyonlar sadece print komutu ile kullanılabilirken ve sonuç değeri üretmezken Space fonksiyonunun sonuç değeri bir string degiskene atanabilir ve her yerde kullanılabilir.

Kullanimi;

Sonuc = Space (Sayi)

Sayi : Birakılacak bosluk karakteri sayisi

Sonuc : Sayi değeri kadar bosluk degerinden üretilmiş string degisken

Örnek;

```
Private Sub Form_Load()
Dim a As String
a = "Veli"
Show
Print a; Tab(10); "Akçakaya"
Print a; Spc(10); "Akçakaya"
End Sub
```

String, String\$ () Fonksiyonu

Bu fonksiyon argüman olarak verilen karakterden veya bu karakterin ASCII kodundan veya String degiskenin ilk karakterinden belirtilen sayi kadar çoğaltarak string bir degisken olusturur.

Kullanimi;

Sonuc = String (Sayi, Karakter)

Sayi : Olusturulacak yeni degiskenin karakter sayisi

Karakter : Çoğaltılacak karakter veya ASCII kodu

Sonuc : Sayi kadar karakterden olusmus string degisken

Örnek;

```
Private Sub Form_Load()
Dim MyString
MyString = String(5, "*") ' Returns "*****".
```



```
MyString = String(5, 42) ' Returns "*****".  
MyString = String(10, "ABC") ' Returns "AAAAAAAAAA".  
End Sub
```

StrComp() Fonksiyonu

Argüman olarak verilen String1 ve String2 degiskenlerini olusturan karakterlerin ASCII kodlarini dikkate alarak soldan saga dogru karakter karakter kiyaslar. Fonksiyonun kiyas degeri 1 olarak verilirse fonksiyon karsilastirma sirasinda büyük/küçük harf ayrimi yapmaz. Bu deger 0 veya hiç verilmezse küçük/büyük harf ayrimi yapılır. Kiyaslama islemi kiyaslanan, kiyaslanan karsilikli karakterlerin birbirlerinden farkli olmasına kadar veya String1 veya String2 degiskenlerinden birine ait NULL karakterine rastlanincaya kadar devam eder. Farkli bir veya NULL karakterine rastlanildiginda islem sona erer ve kiyaslamanin sonucunu üreten bir tamsayi üretilir. Sonuç degerine göre Stringlerin hangisinin büyük oldugu asagida verilmistir.

```
Sonuc = 1 ise String1 > String2  
Sonuc = 0 ise String1 = String2  
Sonuc = -1 ise String1 < String2
```

Kullanimi;

Sonuc = Strcomp (String1, String2, Kiyas)

String1 : Karsilastirilacak ilk degisken

String2 : Karsilastirilacak ikinci degisken

Kiyas : Karsilastirmanin kiyas sarti.

Örnek;

```
Private Sub Form_Load()  
Dim MyStr1, MyStr2, MyComp  
MyStr1 = "ABCD"  
MyStr2 = "abcd"  
MyComp = StrComp(MyStr1, MyStr2, 1) ' MyComp = 0  
MyComp = StrComp(MyStr1, MyStr2, 0) ' MyComp = -1  
MyComp = StrComp(MyStr2, MyStr1) ' MyComp = 1  
End Sub
```

Tab,Spc () Fonksiyonu

Print komutu ile kullanılan bu fonksiyonlar belirtilen sayida bosluk karakteri bırakır.

Kullanimi;

Tab (Sayi) veya Spc(Sayi)

Sayi :Bırakılacak boşluk karakteri sayısı

Örnek;

```
Private Sub Form_Load()  
Dim a As String  
a = "Veli"  
Show  
Print a; Tab(10); "Akçakaya"  
Print a; Spc(10); "Akçakaya"  
End Sub
```

Ucase(), Ucase\$() Fonksiyonu

Argüman olarak verilen string değişkenin içerdigi tüm küçük harfleri büyük harfe dönüştürür. Bu fonksiyondada türkçe karakterler büyük harfe dönüştürülemezler.

Kullanımı;

Sonuc = Ucase (Değişken)

Değişken : Büyük harfe dönüştürülecek string karakter

Örnek;

```
Private Sub Form_Load()  
Dim UpperCase, LowerCase  
UpperCase = "Hello World "  
LowerCase = UCase(UpperCase)' LowerCase = "HELLOWORLD"  
End Sub
```

B. MATEMATİKSEL FONKSİYONLAR

Abs() Fonksiyonu

Verilen sayının mutlak değerini hesaplar.Sayı Integer veya Double olabilir. Bu fonksiyonunu ürettiği değer yine Integer veya Double dir. Komutun kullanım şekli aşağıda verilmiştir.

Kullanımı;

Sonuc=Abs(Sayı)

Sayı : Mutlak değeri bulunacak Integer veya Double değişken tipinde bir sayı

Sonuc : Fonksiyonun ürettiği Integer veya Double değişken tipinde bir sayı.

Örnek;

```
Private Sub Form_Load()
    Dim a, b As Double
    a = -5.66
    b = Abs(a)
End Sub
```

Atn() Fonksiyonu

Verilen sayının arkatanjantini hesaplar. Eger sayi yerine integer bir degisken tipinde deger vererseniz program 0 degerini üretir.

Kullanimi;

Sonuc=Atn(Sayi)

Sayi : Arkatanjanti hesaplanacak Integer veya Double degisken tipinde bir sayi.

Sonuc : Fonksiyonun ürettiği Integer veya Double degisken tipinde bir sayi.

Örnek;

```
Private Sub Form_Load()
    Dim pi
    pi = 4 * Atn(1)
End Sub
```

Cos() Fonksiyonu

Verilen açının Cosinüsünü hesaplar. Derece cinsinden olan bir açının cosinüsünü hesaplamak istediginizde önce Radyan'a çevirmeniz gerekir. Bunuda açiyi Pi sayısı ile çarpim 180 bölerek yapabilirsiniz.

Kullanimi;

Sonuc=Cos(Açı)

Açı : Cosinüsü alınacak açı.

Sonuc : Bu fonksiyonun ürettiği Integer veya Double deger.

Örnek;

```
Private Sub Form_Load()
    Dim aci, aci1
    aci = 360
    aci1 = (aci * 3.14) / 180 'radyan cinsine çevirdik
    aci = Cos(ac1)
End Sub
```

Exp() Fonksiyonu

Exp fonksiyonu kendisine argüman olarak gönderilen sayinin Exponansiyelini yani matematikteki $e(2,718282)$ sayisinin sayi kuvvetini verir. Sayi degeri en fazla 709 olabilir. Eger businir asilrsa 6 numarali RunTime(OverFlow) hatasi meydana gelir ve programin akisi sonlandirilir.

Kullanimi;

Sonuc=Exp(Sayi)

Sayi : 2,718282 sayisinin kuvveti olacak sayi.

Sonuc : Fonksiyonun ürettiği Double tipinde bir degisken

Örnek;

```
Private Sub Form_Load()  
For b = 0 To 4  
Show  
Print (Exp(b) - Exp(-1 * b)) / 2  
Next b  
End Sub
```

Fix() Fonksiyonu

Küsüratli sayilari kendine en yakin büyük tamsayiya tamamlar ve tamsayi kismini alır.

Kullanimi;

Sonuc=Fix(Sayi)

Sayi : Tamsayi kısmi alınacak ondalik sayi

Sonuc : Fonksiyonun ürettiği Integer tipinde bir sayi

Örnek;

```
Private Sub Form_Load()  
Dim a  
a=5,555  
Show  
Print Int(a)  
End Sub
```

Hex() Fonksiyonu

Eger Assembly dili ile ugrasmis veya ugrasiyorsaniz Hexadecimal sistemin ne oldugunu çok iyi bilirsiniz. Bu dilde program yazarken verilerinin kendileri degil Hexadecimal karsiliklari yazilir.HexeDecimal sayi sistemi 16 lik

sistem demektir. Yani 9'dan sonra 10 yerine A ,11 yerinede B harfi gelir.

Kullanimi;

Sonuc=Hex(Sayi)

Sayi : HexeDecimal karsiligi alinacak sayi.

Sonuc : Fonksiyonun ürettiği veriant tipinde bir sayi

Örnek;

```
Private Sub Form_Load()  
Dim a  
For a = 0 To 100  
Show  
Print Hex(a)  
Next a  
End Sub
```

Int() Fonksiyonu

Küsüratli sayilari kendine en yakin küçük tamsayiya tamamlar ve tamsayi kismini alır.

Kullanimi;

Sonuc=Int(Sayi)

Sayi : Tamsayi kısmi alinacak ondalik sayi

Sonuc : Fonksiyonun ürettiği Integer tipinde bir sayi

Örnek;

```
Private Sub Form_Load()  
Dim a  
a=5,555  
Show  
Print Int(a)  
End Sub
```

Log() Fonksiyonu

Pozitif sayilarin dogal logaritmasini alır. Sayi Negatif verilirse 5 numarali runtime hatasi olusur ve programdan çikilir.

Kullanimi;

Sonuc=Log(Sayi)

Sayi : Dogal logaritmasi alinacak pozitif sayi

Sonuc : Fonksiyonun ürettiği Integer veya Double tipinde bir sayi

Örnek;

```
Private Sub Form_Load()  
Dim a  
a=5  
Show  
Print Log(a)  
End Sub
```

Oct() Fonksiyonu

Sayıları Octal sisteme çevirir. Eger sayi yerine string bir ifade verilirse program 0 degerini üretir.

Kullanimi;

Sonuc=Oct(Sayi)

Sayi : Octal sisteme çevrilecek Integer veya Double sayi

Sonuc : Fonksiyonun ürettiği Integer tipinde bir sayi

Örnek;

```
Private Sub Form_Load()  
Dim a  
a = 555,898  
Show  
Print Oct(a)  
End Sub
```

Rnd() Fonksiyonu

Rasgele sayi üretir. Belirtilen aralıkta rasgele sayi üretir. Programın her defasında farklı sayi üretmesi için Rnd Komutundan önce Randomize komutunun kullanılması gerekmektedir. Eger bu komut kullanılmaz ise program her defasında aynı sayıları üretir.

Kullanimi;

Sonuc = Int (Aralık * Rnd)

Yukarıda Int komutu üretilen sayiyi kendisine en yakın küçük tamsayıya tamamlar. Eger bu komutu kullanılmaz ise sonuç double tipinde bir sayi olacaktır. Int yerine Fix komutuda kullanılabilir.

Aralık : Fonksiyonun hangi aralıkta değer üreteceğidir.

Sonuc : Fonksiyonun ürettiği Integer veya Double tipinde bir sayı

Örnek;

```
Private Sub Form_Load()  
Dim Tamsayi: Dim Ondaliksayi  
Randomize  
Tamsayi = Int(50 * Rnd)  
Ondaliksayi = (50 * Rnd)  
Show  
Print Tamsayi  
Print Ondaliksayi  
End Sub
```

Sgn() Fonksiyonu

Ifadenin pozitif, negatif veya 0 olma durumunu inceler. Eğer sayı pozitif ise fonksiyon 1, negatif ise -1 ve 0 ise 0 değerini üretir.

Kullanımı;

Sonuc = Sgn(Sayı)

Sayı : Durumu bulunacak Integer veya Double sayı

Sonuc : Fonksiyonun ürettiği Integer tipinde bir sayı

Örnek;

```
Private Sub Form_Load()  
durum = Sgn(155) ' Üretilen değer 1.  
Show  
Print durum  
durum = Sgn(-5333.58) ' Üretilen değer -1.  
Print durum  
durum = Sgn(0) ' Üretilen değer 0.  
Print durum  
End Sub
```

Sin () Fonksiyonu

Verilen açının Sinüsünü hesaplar. Derece cinsinden olan bir açının sinüsünü hesaplamak istediğinizde önce Radyan'a çevirmeniz gerekir. Bunuda açıyı Pi sayısı ile çarpım 180 bölerek yapabilirsiniz.

Kullanımı;

Sonuc=Sin(Açı)

Açı : Sinüsü alınacak açı.

Sonuc : Bu fonksiyonun ürettiği Integer veya Double değer.

Örnek;

```
Private Sub Form_Load()  
Dim aci, aci1  
aci = 360  
aci1 = (aci * 3.14) / 180 'radyan cinsine çevirdik  
aci = Sin(ac1)  
End Sub
```

Sqr() Fonksiyonu

Pozitif sayıların kaekökünü hesaplar. Eger sayı negatif verilir ise program 5 numaralı hatayı oluşturur ve sonlandırılır.

Kullanımı;

Sonuc = Sqr(Sayı)

Açı : Karekökü alınacak sayı.

Sonuc : Bu fonksiyonun ürettiği Integer veya Double değer.

Örnek;

```
Private Sub Form_Load()  
Show  
Print Sqr(9)  
Print Sqr(4)  
Print Sqr(25)  
End Sub
```

Tan() Fonksiyonu

Verilen açının Tanjantini hesaplar. Derece cinsinden olan bir açının tanjantini hesaplamak istediğinizde önce Radyan'a çevirmeniz gerekir. Bunuda açıyı Pi sayısı ile çarpım 180'e bölerek yapabilirsiniz.

Kullanımı;

Sonuc=Tan(Açı)

Açı : Tanjanti alınacak açı.

Sonuc : Bu fonksiyonun ürettiği Integer veya Double değer.

Örnek;

```
Private Sub Form_Load()  
Dim aci, aci1
```



```
aci = 360  
aci1 = (aci * 3.14) / 180 'radyan cinsine çevirdik  
aci = tan(aci1)  
End Sub
```

Val() Fonksiyonu

Alfa numerik yani hem sayi hemde string bilgi içeren degiskenlerde soldan baslayara her hangi bir harfe gelinceye kadar olan sayilari sonuç degeri olarak üretir. Bosluk karakterinin herhangi bir etkisi yoktur.

Kullanimi;

Sonuc = Sqr(Veriant Degisken)

Sayi : Alinacak sayinin bulundugu Veriant degisken

Sonuc : Bu fonksiyonun ürettiği Integer veya Double deger.

Örnek;

```
Private Sub Form_Load()  
Show  
Print Val("2457") ' Deger 2457.  
Print Val(" 2 45 7") ' Deger 2457.  
Print Val("24 ve 57") ' Deger 24.  
End Sub
```

C.DEGISKEN KONTROL FONKSIYONLARI

IsDate() Fonksiyonu

Istenilen degiskenin Date (Tarih) formatinda olup olmadigini kontrol eder. Eger degisken Date formatinda ise fonksiyon True, degilse False degerini üretir. Fonksiyon veriant degiskenlerde False degerini üretir.

Kullanim;

Sonuc = IsDate (Degisken)

Sayi : Formatı kontrol edilecek herhangi bir degisken.

Sonuc : Fonksiyonun ürettiği Boolean degisken.

Örnek;

```
Private Sub Form_Load()  
Dim Dveriant  
Dim DDate As Date  
Dim DInteger
```

```
Show
Print IsDate(Dvariant) 'False
Print IsDate(DDate) 'True
Print IsDate(DInteger) 'False
Dim DInteger
End Sub
```

IsNull() Fonksiyonu

Istenilen degiskenin NULL olup olmadigini kontrol eder. Herhangi bir degiskeni fonksiyon ile kontrol ettirdiginizde eger degisken NULL ise program True, degilse False degerini üretir.

Kullanım;

Sonuc = IsNull (Degisken)

Sayi : Formatı kontrol edilecek herhangi bir degisken.

Sonuc : Fonksiyonun ürettiği Boolean degisken.

Örnek;

```
Private Sub Form_Load()
Dim D1: Dim D2: Dim D3
D1 = Null: D3 = Null
Show
Print IsNull(D1) 'True
Print IsNull(D2) 'False
Print IsNull(D3) 'True
End Sub
```

Lbound() Fonksiyonu

Çok boyutlu dizilerde numarası verilen dizilerin başlangıç degerlerini bu fonksiyon ile öğrenebilirsiniz.

Kullanım;

Sonuc = LBound (Dizi Adi , Sirasi)

Dizi Adi : Çok boyutlu dizi.

Sirasi : Çok boyutlu dizinin hangisinin kontrol ettirileceğinin degeri. Eger Sıra dizide tanımlı degilse program 9 numaralı hata ile sonlandırılır.

Sonuc : Fonksiyonun ürettiği Integer degisken.

Örnek;

```
Private Sub Form_Load()
Dim Dizi(1 To 10, 5 To 15, 10 To 20)'Dizi tanımlanıyor
```

```

Dim Dizi1(10)
Show
Print LBound(Dizi, 1) ' Deger 1.
Print LBound(Dizi, 3) ' Deger10.
Print LBound(Dizi1) ' Deger 0
End Sub

```

TypeName() Fonksiyonu

Verilen degiskenin türünün adini verir. Örneğin degisken Integer olarak tanımlanıp bu fonksiyon ile kontrol ettirilirse sonuç degeri olarak "Integer" string bilgisini üretir.

Kullanım;

Sonuc = TypeName(Degisken Adi)

Degisken Adi : Degisken tipi öğrenilecek herhangi bir degisken.

Sonuc : Fonksiyonun ürettiği String degisken.

Örnek;

```

Private Sub Form_Load()
Dim NullVar, MyType, StrVar As String
IntVar As Integer
CurVar As Currency
Dim ArrayVar(1 To 5) As Integer
NullVar = Null ' Assign Null value.
MyType = TypeName(StrVar) ' Sonuç "String".
MyType = TypeName(IntVar) ' Sonuç "Integer".
MyType = TypeName(CurVar) ' Sonuç "Currency".
MyType = TypeName(NullVar) ' Sonuç "Null".
MyType = TypeName(ArrayVar) ' Sonuç "Integer()".
End Sub

```

VarType() Fonksiyonu

Bu fonksiyonda TypeName fonksiyonu gibi verilen degiskenin türünü belirler. Ancak fonksiyon yukarıdaki fonksiyondan farklı olarak degiskenin adini değil kodunu verir.

Kullanım;

Sonuc = VarType (Degisken Adi)

Degisken Adi : Degisken tipi öğrenilecek herhangi bir degisken.

Sonuc : Fonksiyonun ürettiği Integer degisken.

Asagidaki sayfada degiskenlerin kod tablolari verilmistir. Bu degerleri kullanarak degiskenlerin tiplerini öğrenebilirsiniz.

Degisken	Dönüsü	Degisken	Dönüsü
Empty	0	String	8
Null	1	Object	9
Integer	2	Error	10
Long	3	Boolean	11
Single	4	Variant	12
Double	5	DataObject	13
Currency	6	Decimal	14
Date	7	Byte	17
		Array	8192

Örnek;

```
Private Sub Form_Load()  
Dim IntVar, StrVar, DateVar, MyCheck  
IntVar = 459  
StrVar = "Merhaba"  
DateVar = #2/12/69#  
MyCheck = VarType(IntVar) ' Deger 2.  
MyCheck = VarType(DateVar) ' Deger 7.  
MyCheck = VarType(StrVar) ' Deger 8.  
End Sub
```

D. DISK ISLEM FONKSIYONLARI

ChDir() Fonksiyonu

Dizin degistirmek için kullanılan bir fonksiyondur. Aktif olan dizin bu fonksiyon ile degistirilir.

Kullanimi;

Chdir "Dizin Adi"

Dizin Adi : Aktif hale getirilecek dizin.

Örnek;

```
Private Sub Form_Load()  
ChDir "c:\WINDOWS\SYSTEM"  
End Sub
```

ChDrive Fonksiyonu

Aktif olan sürücüyü degistirir.

Kullanimi;

ChDrive "Sürücü Harfi"

Sürücü Harfi : Seçeceginiz sürücünün adi

Örnek;

```
Private Sub Form_Load()  
ChDrive "A"  
End Sub
```

CurDir Fonksiyonu

O anda aktif olan dizini verir. Eger sürücü verilmezse aktif sürücüdeki aktif dizin, verilirse verilen sürücüdeki aktif dizini verir.

Kullanimi;

CurDir veya CurDir "Sürücü Harfi"

Sürücü Harfi : Aktif dizini bulacaginiz sürücü

Örnek;

```
Private Sub Form_Load()  
Dim MyPath  
MyPath = CurDir  
MyPath = CurDir("C")  
MyPath = CurDir("D")  
End Sub
```

Dir Fonksiyonu

Belirtilen dizindeki verilen dosya isimlerini arar.

Kullanimi;

dosya = Dir("Dosya Adi",özelligi)

Dosya : Eger dosya bulunursa dosyanin içerigine dosya adi atanir,eger bulunamazsa dosya'nin içeriği bos " " olur.

Dosya Adi : Aranacak dosyanin adi

Özelligi : Aranacak dosyanin özellikleri

Örnek;

```
Private Sub Form_Load()  
MyName = Dir("c:\", vbDirectory)  
Do While MyName <> ""  
MyName = Dir 'Bir sonraki dizin okunur.  
Show  
Print MyName 'Okunan dizin form üzerine yazdirildi.  
Loop  
End Sub
```

Environ Fonksiyonu

Windows'un veya DOS'un bazı Path'leri vardır. Bunlari bu fonksiyon ile anlayabiliriz. Örneğin bunlardan biride Prompt'tur.

Kullanimi;

Sonuc = Environ(Degeri)

Degeri : Path'i bulacaginiz özelligin degeri. Bu deger 1-8 arasinda olmalidir. 0 olursa hata olusur.

Sonuc : Verilen degere göre fonksiyonun ürettiği Path'lar

Örnek;

```
Private Sub Form_Load()  
Dim EnvString, a  
For a = 1 To 8  
EnvString = Environ(a)  
Show  
Print EnvString  
Next a  
End Sub
```

FileDateTime Fonksiyonu

Belirtilen dosyanin olusturulma tarih ve saatini verir.

Kullanimi;

Sonuc = FileDateTime ("Dosya Adi")

Sonuc : Fonksiyonun ürettiği dosyanin olusturulma tarih ve saatinin saklandigi degisken

Dosya Adi : Ne zaman olusturulduğunu bulacaginiz dosyanin tam adi

Örnek;

```
Private Sub Form_Load()  
Dim Adi  
Adi = FileDateTime("c:\autoexec.bat")  
MsgBox Adi  
End Sub
```

FileLen Fonksiyonu

Belirtilen dosyanın bayt türünden disk üzerinde kapladığı alanı verir.

Kullanımı;

Sonuc = FileLen ("Dosya Adi")

Sonuc : Dosyanın disk üzerinde kapladığı alanın atandığı Integer değişken

Dosya Adi : Uzunluğu bulunacak dosyanın tam adı

Örnek;

```
Private Sub Form_Load()  
Dim uzunluk  
uzunluk = FileLen("c:\autoexec.bat")  
MsgBox uzunluk  
End Sub
```

FileCopy Fonksiyonu

Dosyaları bir yerden başka bir yere kopyalar

Kullanımı;

FileCopy "Kaynak Dosya", "Hedef Dosya"

Kaynak Dosya : Kopyalanacak dosyanın kaynak dizini ile tam adı

Hedef Dosya : Kopyalanacak dosyanın hedef dizini ve yeni veya eski adı

Örnek;

```
Private Sub Form_Load()  
FileCopy "c:\mesaj10.txt", "c:\belgelerim\mesaj10.txt"  
End Sub
```

Get Attr Fonksiyonu

Disk üzerinde istenen dosyaya ait arşiv,gizli,sistem vb. bilgileri verir.
Her özelliğin bir kodu vardır. Bunlar aşağıda tablo halinde verilmiştir.

Kullanimi;

Sonuc = GetAttr ("Dosya Adi")

Sonuc : Dosyanın özelliğine göre üretilen kodun saklandığı değişken

Dosya Adi : Özelliği bulunacak dosyanın tam adı

GetAttr Fonksiyonu'nun dönüş değerleri ve karşılıkları

Constant	Value	Description
Normal	0	Normal Dosya
ReadOnly	1	Sadece Okunabilir Dosya
Hidden	2	Gizli Dosya
System	4	Sistem Dosyası
Directory	16	Klasör
Archive	32	Yedek dosya

Örnek;

```
Private Sub Form_Load()  
  
    Dim ozelik  
  
    ozelik = GetAttr("c:\autoexec.bat")  
  
    MsgBox ozelik  
  
End Sub
```

Kill Fonksiyonu

Belirtilen dosyayı disk üzerinden tamamen siler

Kullanimi;

Kill ("Dosya Adi")

Dosya Adi : Silinecek dosyanın tam adı

Örnek;

```
Private Sub Form_Load()  
    Kill ("C:\Deneme.Bak")  
End Sub
```


MkDir() Fonksiyonu

Yeni ve içi bos bir dizin olusturur.

Kullanimi;

Mkdir "Dizin Adi"

Dizin Adi : Yeni olusturulacak dizinin adi

Örnek;

```
Private Sub Form_Load()  
Mkdir "c:\deneme"  
End Sub
```

Name Fonksiyonu

Belirtilen dosya yada dizinlerin isimlerini degistirir ve istersek bu dosyalari bir yerden baska bir yere tasiyabilir.

Kullanimi;

Name "Eski Dosya Adi","Yeni Dosya Adi"

Eski Dosya Adi : Ismi degistirilecek dosyanin tam yolu ile eski adi

Yeni Dosya Adi : Ismi degistirilecek dosyanin tam yolu ile yeni adi

Örnek;

```
Private Sub Form_Load()  
Name "c:\mp3" As "c:\belgelerim\mp3"  
End Sub
```

Rmdir Fonksiyonu

İçi bos dizinleri silmek için kullanılır. İçi dolu dizinleri silemeyiz.

Kullanimi;

Rmdir "Dizin Adi"

Dizin Adi : İçi bos silinecek dizin

Örnek;

```
Private Sub Form_Load()  
Rmdir "c:\deneme"  
End Sub
```

Yukaridaki örnekte C sürücüsü içinde bulunan ve gizli olmayan tüm dizin ve dosyalari listeledik.

E.DOSYALAMA FONKSİYONLARI

Bilgisayar üzerinde dosya oluşturmak, oluşturulmuş olan dosyaları okumak için kullanılan , kısacası dosyalama komutları aşağıda verilmiştir.

Open Fonksiyonu

Disk üzerinde yeni bir dosya açar.

Kullanımı;

Open "Dosya adı" For Açılış Modu As #Dosya Numarası

Dosya Adı : Dosyanın disk üzerinde hangi ada sahip olacağını buraya yazın

Açılış Modu : Dosyanın hangi modda açılacağını belirleyen bölümdür

D.Numarası : Dosyanın program içinde hangi numara ile açılacağı burada belirlenir

Örnek;

```
Private Sub Form_Load()  
    Open "Deneme" For Output As #1  
End Sub
```

Yukarıdaki örnekte Deneme adında yeni bir dosya açıldı.

Write ve Print Fonksiyonu

Daha önceden Open deyimi ile açılan dosyalara bilgi yazdırmak için kullanılır. Write ile Print arasındaki fark; Write deyiminin dosyaya yazdırılan bilgiler arasına virgül koyması ve Print'ın bu virgülleri koymamasıdır.

Kullanımı;

Write # Dosya Numarası , Değişkenler

Print # Dosya Numarası , Değişkenler

Değişkenler : Dosyaya yazdırılacak değişkenler

D.Numarası : Open deyimi ile açılan dosya numarası olmalı

Örnek;

```
Private Sub Form_Load()  
    Open "Deneme" For Output As #1  
    Write #1, Ad  
    Print #1, Soyad  
End Sub
```

Yukarıdaki örnekte Deneme dosyasına Ad ve Soyad değişkeni içerisindeki bilgiyi yazdırdık.

Input Fonksiyonu

Daha önceden Open deyimi ile açılan ve Write veya Print komutlari ile dosyaya yazdirilan bilgileri sirasi ile okumak için kullanilir.

Kullanimi;

Input # Dosya Numarasi , Degiskenler

Degiskenler : Dosyadan okunana bilgilerin atanacagi degiskenler.

D.Numarasi : Open deyimi ile açılan dosyanin numarasi

Örnek;

```
Private Sub Form_Load()  
Open "Deneme" For Input As #1  
Input #1, Ad  
End Sub
```

Yukaridaki örnekte Deneme dosyasindan okunan bilgileri ad degiskenine atadik.

Close Fonksiyonu

Open deyimi ile açılan dosyalari kapatmak için kullanilir.

Kullanimi;

Close #Dosya numarasi

D.Numarasi : Kapatilacak dosyanin açilis numarasi

Örnek;

```
Private Sub Form_Load()  
Open "Deneme" For Input As #1  
Close #1  
End Sub
```

Yukaridaki örnekte Deneme dosyasi açilip kapatildi.

FileAttr Fonksiyonu

Open deyimi ile açilmis olan dosyanin hangi modda açildigini gösterir. Fonksiyonun ürettiği deger açilis modunun kendisi degil kod numarasidir. Bu kod numaralari asagida verilmistir.

Dosyalarin Açilis Modu Kod Tablosu

Modu	Açilis Kodu
------	-------------

Microsoft Visual Basic 6.0

Input	1
Output	2
Random	4
Append	8
Binary	32

Kullanimi;

Sonuc = FileAttr (Dosya Numarasi, Dönüs tipi)

D.Numarasi : Açilis modü öğrenilecek dosyanın Açilis numarasıdır.

Dönüs tipi Açilis modünün kodunu elde etmek için bu deęer sürekli 1 olur.

Sonuc : Açilis modünün kod numarası olan dönüs deęeridir.

Örnek;

```
Private Sub Form_Load()  
Open "TESTFILE" For Append As #1  
MsgBox FileAttr(1, 1)  
Close 1  
End Sub
```

FreeFile Fonksiyonu

Dosyanın açilis numarasını en düşük seviyede seçer. Yani açılmamış olan dosya numarasının en küçük olanını seçer. Aşağıda bunlar örnek tablo ile açıklanmıştır.

FreeFile Fonksiyonuna ait örnek tablo;

- 1.Dosya 'nın açilis numarası 1 olsun
- 2.Dosya 'nın açilis numarası 2 olsun
- 3.Dosya 'nın açilis numarası 5 olsun
- 4.Dosya 'nın açilis numarası 6 olsun
- 5.Dosya 'nın açilis numarası 9 olsun
- 6.Dosyanın dosya numarasına FreeFile fonksiyonun sonuç deęerini atarsak, bu deęer 3 olur.

Kullanimi;

Sonuc = FreeFile

Sonuc : Kullanılmamış olan en düşük dosya numarası.

Örnek;

```
Private Sub Form_Load()  
Dim f  
f = FreeFile
```

```
MsgBox f  
End Sub
```

Loc Fonksiyonu

Open deyimi ile açılmış olan dosyada okunan/yazılan yani işlem yapılan pozisyonu verir.

Kullanımı;

Sonuc = Loc(Dosya Numarasi)

D.Numarasi : İşlem gören alanın bulunacağı dosya

Sonuc : Dosyanın işlem gören pozisyonunu verir

Örnek;

```
Private Sub Form_Load()  
Dim f  
Dim d As Integer  
Open "License.txt" For Input As #1  
Input #1, f  
d = Loc(1)  
Show  
Print f  
Print d  
Close (1)  
End Sub
```

Lof Fonksiyonu

Open deyimi ile açılan dosyada okunan yazılan pozisyonu verir.

Kullanımı;

Sonuc = Lof (Dosya Numarasi)

D.Numarasi : Uzunluğu bulunacak dosya

Sonuc : Dosyanın uzunluğunun üretildiği integer değerdir

Örnek;

```
Private Sub Form_Load()  
Dim d As Integer  
Open "License.txt" For Input As #1  
d = Lof(1)  
Show  
Print d  
Close (1)  
End Sub
```

Seek Fonksiyonu

Open deyimi ile açılan dosyada okunan yazılan degiskenin uzunlugunu verir.

Kullanimi;

Sonuc = Seek (Dosya Numarasi)

D.Numarasi : Uzunlugu bulunacak dosya

Sonuc : Dosyada okunan degiskenin uzunlugunun üretildiği integer degerdir

Örnek;

```
Private Sub Form_Load()  
Dim MyChar  
Open "c:\mesaj11.txt" For Input As #1  
Input #1, MyChar  
MsgBox Seek(1)  
Loop  
Close #1  
End Sub
```

EOF Fonksiyonu

Open deyimi ile açılan dosyanın sonuna gelinip gelinmediğini belirtir.

Kullanimi;

Sonuc = EOF (Dosya Numarasi)

D.Numarasi : Kontrolü yapılacak dosya numarasi

Sonuc : Eger dosyanın sonuna gelinmisse True gelinmemisse False degeri üretilir

Örnek;

```
Private Sub Form_Load()  
Dim InputData  
Open "c:\mesaj11.txt" For Input As #1  
Do While Not EOF(1)  
Input #1, InputData  
MsgBox InputData  
Loop  
Close #1  
End Sub
```

F. Format Dönüşüm Fonksiyonlari

Değişik tipteki değişkenleri, bir birlerine çevirmek için kullanılan komutlar aşağıda açıklamaları ile verilmiştir.

Array () Fonksiyonu

Variant türündeki değişkeni dizi haline dönüştürür. Bu diziler içine atama bu dönüşüm sırasında olabilmektedir.

Kullanımı;

Variant Değişken = Array (değer1,değer2,...,değerN)

V.Değişken : Dizi haline dönüştürülecek variant değişken

Değer1 : Bu dizinin ilk değeri

Örnek;

```
Private Sub Form_Load()
    Dim MyWeek, MyDay
    MyWeek = Array("Mon", "Tue", "Wed", "Thu", "Fri", "Sat")
    MyDay = MyWeek(2)
    MyDay = MyWeek(4)
End Sub
```

Ccur () Fonksiyonu

Verilen değişkeni Para türüne çevirir.

Kullanımı;

Sonuc = CCur (Sayı)

Sayı : Para türüne çevrilecek değişken

Sonuc : Fonksiyonun ürettiği yeni değer

Örnek;

```
Private Sub Form_Load()
    Dim MyDouble, MyCurr
    MyDouble = 543.214588
    MyCurr = CCur(MyDouble)
    MsgBox MyCurr
End Sub
```

CDbl () Fonksiyonu

Verilen degiskeni Double yani çift duyarlikli degisken tipine çevirir.

Kullanimi;

Sonuc = CDbl(Sayi)

Sayi : Double türüne çevrilecek degisken

Sonuc : Fonksiyonun ürettiği Double deger

Örnek;

```
Private Sub Form_Load()  
Dim MyCurr, MyDouble, d  
MyDouble = CDbl(26.5666)  
d = VarType(MyDouble)  
MsgBox d 'd=5 yani MyDouble Double degisken tipindedir.  
End Sub
```

Cint () Fonksiyonu

Verilen degiskeni Integer türüne çevirir.

Kullanimi;

Sonuc = Cint (Sayi)

Sayi : Integer türüne çevrilecek degisken

Sonuc : Fonksiyonun ürettiği Integer deger

Örnek;

```
Private Sub Form_Load()  
Dim MyDouble, MyInt  
MyDouble = 2345.5678  
MyInt = Cint(MyDouble)  
MsgBox MyInt  
End Sub
```

CLng () Fonksiyonu

Verilen degiskeni uzun tamsayi yani Longint degisken türüne çevirir.

Kullanimi;

Sonuc = CLng (Sayi)

Sayi : Longint türüne çevrilecek degisken

Sonuc :Fonksiyonun ürettiği Longint deger

Örnek;

```
Private Sub Form_Load()
Dim MyVal1, MyVal2, MyLong1, MyLong2
MyVal1 = 25427.45: MyVal2 = 25427.55
MyLong1 = CLng(MyVal1) ' MyLong1 = 25427.
MyLong2 = CLng(MyVal2) ' MyLong2 = 25428.
End Sub
```

CSng () Fonksiyonu

Verilen degiskeni tek duyarlikli tamsayi yani Single degisken türüne çevirir.

Kullanimi;

Sonuc = CSng (Sayi)

Sayi : Single türüne çevrilecek degisken

Sonuc : Fonksiyonun ürettiği Single deger

Örnek;

```
Private Sub Form_Load()
Dim MyDouble1, MyDouble2, MySingle1, MySingle2
MyDouble1 = 75.3421115: MyDouble2 = 75.3421555
MySingle1 = CSng(MyDouble1) ' MySingle1 = 75.34211.
MySingle2 = CSng(MyDouble2) ' MySingle2 = 75.34216.
End Sub
```

CStr () Fonksiyonu

Verilen ifade degerini dizgi yani string türe dönüştürür. Fonksiyona hangi ifade degeri verilirse verilsin o ifadeyi String bilgiye dönüştürür.

Kullanimi;

Sonuc = CStr (ifade)

Sayi : String türüne dönüştürülecek degisken

Sonuc : Fonksiyonun ürettiği String deger

Örnek;

```
Private Sub Form_Load()
Dim MyDouble As Integer
Dim MyString
MyDouble = 437.324
MyString = CStr(MyDouble) ' MyString = "437.324".
```

```
MsgBox VarType(MyString)
End Sub
```

Yukarıdaki örnekte Integer olarak tanımlanan MyDouble degiskeni string degiskene dönüştürülmüştür. Bunuda VarType kontrol ettiğimizde fonksiyon String degiskenin kodu olan 8 degerini bize vermektedir.

Cvar () Fonksiyonu

Verilen ifade degerini variant türde degiskene dönüştürür.

Kullanimi;

Sonuc = CVar (ifade)

Sayi : Variant türüne dönüştürülecek degisken veya deger

Sonuc : Fonksiyonun ürettiği Variant deger

Örnek;

```
Private Sub Form_Load()
Dim MyInt, MyVar
MyInt = 4534
MyVar = CVar(MyInt & "000")
MsgBox MyVar
End Sub
```

DataValue () Fonksiyonu

String formatındaki {Ör; "02,12,1980" gibi} degiskeni tarih yani Date formatına çevirir.

Kullanimi;

Sonuc = DateValue (ifade)

Sayi : Date türüne dönüştürülecek string degisken veya deger

Sonuc : Fonksiyonun ürettiği Date yani tarih formatındaki yeni deger

Örnek;

```
Private Sub Form_Load()
Dim MyDate
MyDate = DateValue("02,12, 1969")
MsgBox MyDate ' MyDate = 02.12.1969
End Sub
```

Format,Format\$ () Fonksiyonu

Verilen bilgiyi istenilen formatta düzenler.

Kullanimi;

Sonuc = Format (degisken , format tipi)

Degisken : Formatli hale getirilecek degisken

Format tipi : Degiskenin hangi format tipinde olacagidir

Sonuc :Fonksiyonun ürettiği herhangi bir degisken

Örnek;

```
Private Sub Form_Load()
Dim MyTime, MyDate, MyStr
MyTime = #5:04:23 PM#
MyDate = #1/27/93#
MyStr = Format(Time, "Long Time")
MyStr = Format(Date, "Long Date")
MyStr = Format(MyTime, "h:m:s") ' Dönüs Degeri = "17:4:23".
MyStr = Format(MyTime, "hh:mm:ss AMPM")'Dönüs=
"05:04:23 PM".
MyStr = Format(MyDate, "dddd, mmm d yyyy")
MyStr = Format(23) ' Returns "23".
MyStr = Format(5459.4, "##,##0.00")'Dönüs Degeri =
"5,459.40".
MyStr = Format(334.9, "###0.00") ' Dönüs Degeri = "334.90".
MyStr = Format(5, "0.00%") ' Dönüs Degeri = "500.00%".
MyStr = Format("HELLO", "<") ' Dönüs Degeri = "hello".
MyStr = Format("This is it", ">")'Dönüs Degeri = "THIS IS IT".
End Sub
```

Str,Str\$ () Fonksiyonu

Integer veya double formatındaki degisken veya verileri string formatına dönüştürür.

Kullanimi;

Sonuc = Str (deger)

Deger : String formatına dönüştürülecek herhangi bir degisken veya veri

Sonuc : Fonksiyonun ürettiği String formatındaki sonuç degeri

Örnek;

```
Private Sub Form_Load()
Dim MyString
```

```
MyString = Str$(459) ' Dönüş degeri = " 459".  
MyString = Str$(-459.65) ' Dönüş degeri = "-459.65"  
MyString = Str$(459.001) ' Dönüş degeri = " 459.001"  
MsgBox VarType(MyString) ' Dönüş degeri = 8 { String }  
End Sub
```

TimeSerial () Fonksiyonu

Integer, string veya double formatındaki {Ör; 16,35,17 gibi} verileri alıp saat formatına dönüştürür.

Kullanimi;

Sonuc = TimeSerial (deger1, deger2, deger3)

Deger1 : Saat formatına dönüştürülecek verinin Saat'i oluşturan kısmı

Deger2 : Saat formatına dönüştürülecek verinin dakika'yi oluşturan kısmı

Deger3 : Saat formatına dönüştürülecek verinin saniye'yi oluşturan kısmı

Sonuc : Fonksiyonun ürettiği Time { 16:35:17 gibi } formatındaki sonuç degeri

Örnek;

```
Private Sub Form_Load()  
Dim MyTime  
MyTime = TimeSerial(16, 35, 17)  
MsgBox MyTime ' MyTime = 16:35:17  
End Sub
```

G. Yükleme Fonksiyonlari

End Fonksiyonu

Hiç kosulsuz programdan çıkar yani programın akisini direkt sonlandırır

Örnek;

```
Private Sub Form_Load()  
Msgbox "Veli Akçakaya"  
End 'program kosulsuz sonlandirildi  
End Sub
```

Exit Sub Fonksiyonu

Formun veya denetimin kod sayfasındaki kodlari bu komutun kullanildigi satirdan itibaren okumaz yani yok sayar.

Inputbox Fonksiyonu

Msgbox kullaniciya bilgi verme, onu uyarma ve bu uyarilar karsisinda gerektiginde bir cevap alabiliyordu. Inputbox komutu da disaridan bilgi alır. Fakat bu komut Msgbox komutundan farkli olarak belirli degerleri degil kullanicinin kendi girdigi verileri almaktadır. Yani disaridan herhangi bir String, Integer vs. tipinde veriler alabiliriz.

Kullanimi;

Inputbox (" Açıklama " , " Baslik " , Deger)

Açıklama : Mesaj kutusuna ne tür veri girilecegi açıklama olarak yazılır.

Baslik : Inputbox 'un Caption özelligidir.

Sonuc : İlk ekrana geldiginde bulunacak deger.

Load Fonksiyonu

Belirtilen objeyi yükler.

Örnek;

```
Private Sub Form_Load()  
Msgbox "Form 2 yüklenecek"  
Load Form2  
End Sub
```

LoadPicture Fonksiyonu

Istenilen resim formatindaki dosyayi belirtilen objeye yükler

Kullanimi;

Obje.Picture = LoadPicture(Dosya adi)

Dosya Adi : Yüklenecek dosyanin tam yolu ile adi

Örnek;

```
Private Sub Form_Load()  
Form1.Picture = LoadPicture("C:\windows\Bubbles.bmp")  
End Sub
```

Msgbox Fonksiyonu

Tüm görsel programlama dillerinde kullanıcidan bilgi alan, kullanıcıya bilgi veren gerektiğinde uyarı mesajı bazı komutlar, denetimler vardır. Visual Basic'te bu komut MsgBox tür.

Kullanımı;

Msgbox Mesaj , Butonlar , Başlık , Yardım Dosyası ismi , Bağlam

Mesaj : Mesaj kutusu ile ekrana yazılmasını istediğiniz mesaj. Bu bir değişken olabileceği gibi bir değer de olabilir.

Butonlar : Mesaj kutusunda çıkmasını istediğiniz butonlar ve/veya grafikler. Butonların yanına isterseniz Visual Basic'in önceden ayarlanmış resimlerin'den de çıkartabilirsiniz. Hem resim hem de buton çıkartmak için buton ve resim arasına "+" koymanız gerekir.

Başlık : Başlık kısmı ise Mesaj kutusunun Caption özelliğidir. Titlebar da çıkmasını istediğimiz yazıyı burada yazabiliriz.

Msgbox Ekran Tusları

vbOKOnly	Sadece OK tuşu
vbOKCancel	OK ve Cancel tuşları
vbYesNo	Yes ve No tuşları
vbYesNoCancel	Yes, No ve Cancel tuşları
vbRetryCancel	Retry ve Cancel tuşları
vbAbortRetryIgnore	Abort, Retry ve Ignore tuş.

Msgbox Ekran Resimleri

vbInformation	Mesaj kutusuna "i" resmini ek.
vbExclamation	Mesaj kutusuna "!" resmini ek.
vbQuestion	Mesaj kutusuna "?" resmini ek.
vbCritical	Mesaj kutusuna "X" resmini ek.

Çoğu komut gibi MsgBox komutunun da ürettiği bir sonuç değeri vardır. Komutu, bu değeri bir değişkene atayarak aşağıdaki gibi kullanırız;

sonuc = MsgBox ("Bir hata oluştu", vbRetryCancel + vbCritical, "Dikkat !")

Aşağıdaki tuşların ürettikleri sonuç değerleri verilmiştir.

<u>Tus</u>	<u>S.Degeri</u>
OK	1
Cancel	2
Yes	6
No	7
Retry	4
Abort	3
Ignore	5

Show Fonksiyonu

Belirtilen formu veya objeyi gösterir.

Kullanimi;

Form_Adi.Show

Örnek;

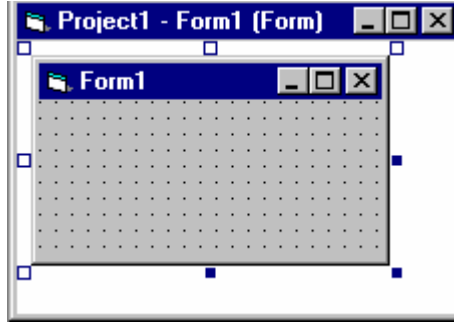
```
Private Sub Form_Load()  
  MsgBox "Form 2 gösterilecek"  
  Form2.Show  
End Sub
```

Unload Fonksiyonu

Çalışır haldeki formu kapatmak için kullanılır. Bu sadece form değil herhangi bir obje de olabilir. O an çalışan formu kapatmak için ; Unload Me komutu kullanılabilir. Bu komut yerine; Unload Form Adı komutu kullanılabilir.

6-FORM KULLANIMI

Form windowsdaki pencerenin Visual Basicteki karşılığıdır. Bütün pencerelerde olduğu gibi bu formu çevreleyen pencerenin başlık çubuğu vardır. Başlık çubugundan başka form nesnesini sınırlayan pencerenin sağ üst köşesinde **Ekrani Kapla Simge Durumunu Küçült** ve **Kapat** düğmeleri bulunmaktadır. Aşağıda formun bir şekli görülmektedir.



Form nesnesini çevreleyen pencerenin sol üst köşesinde **Denetim Düğmesi** bulunmaktadır. Bu düğmeye tiklanırsa **Denetim Menüsü** açılır. Form nesnesini çevreleyen pencerenin denetim menüsünde pencereyi simge durumuna küçültmede kullanılan simge durumuna küçültmede kullanılan **Simge Durumunu Küçült** ve visual basic penceresini kaplamasını sağlayan **Ekrani Kapla** gibi komutlar bulunmaktadır. Formu içeren pencerenin denetim menüsünde ki **Close** komutu ile form kapatılır. Bu pencerenin denetim menüsünde bulunan komutlar, herhangi bir Windows uyumlu pencerenin denetim menüsünde bulunan komutlardan farklı değildir.

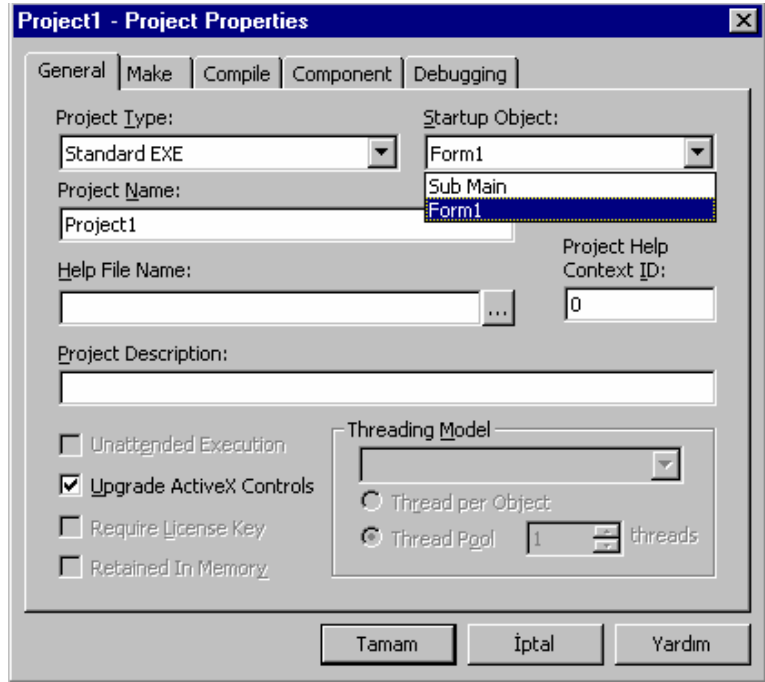
Formu içeren pencerenin boyutlarını Windows uyumlu diğer pencereler gibi değiştirebilirsiniz (herhangi bir uçundan tutup çekerek). Denetim menüsünden **Maximize** komutunu verecek olursanız formu içeren pencere visual Basic penceresinin tümünü kaplar. **Minimize** komutu ise pencereyi simge durumuna küçültür.

Visual Basic penceresinin sol alt köşesindeki çubuğun üzerine konulan düğmelerden yararlanarak formu içeren pencereyi tekrar açabilir veya kapatabilirsiniz. Eğer o sırada projenin formu üzerinde işlem yapmayacaksanız, formu içeren pencereyi kapatabilirsiniz. Formu içeren pencereyi kapattıktan sonra, form üzerinde işlem yapmaya gerek duymanız halinde **Project Explorer** penceresinden veya **View** menüsündeki **object**

komutundan yararlanabilirsiniz. Forma ait baslik çubuguna çift tiklanirsa form, ekrani kaplama öncesi boyutuna ulasir. Formu içeren pencerenin içinde yer alan formlar seçili duruma getirilmis olan nesneler gibi etrafi sekiz adet küçük kare ile çevrilmektedir. Bu sirada Mouse isaretçisini formun sag üst kösesine getirecek olursak isaretçi seklinde degismeler olur. Formun büyüklüğünü formun alt kenarından Mouse isaretçisi ile tutarak degistirebilirsiniz.

Windows arabiriminin en temel kontrolü formlardir. Windowsta hemen hemen her program formlar üzerinde çalışir. Zaten windows kelimesinin Türkçe anlamı olan pencereler de bu formlardir. Boyutlandırılabilir. Özelliği sayesinde aynı anda ekranda tek bir program olmak zorunda değildir.

- Formun propertislerini formun alt programlarında yazarken formun ismini kullanmak zorunda değilsiniz, direkt propertis ismini vermeniz yeterlidir. Yani **form1.caption** ile sadece **caption.form1**'in alt programlarında aynı etkiye sahiptir.
- Programınızda birden fazla form bulunacaksa ilk olarak ana form çalışacaktır. Diğer formları programınızda kullanacağınız **form2.show** gibi yöntemle aktif hale getirmelisiniz.
- Bir formda başka bir formun bir özelliğine ulaşabilmek için kontrol adından önce formun adı da verilmelidir. Örneğin **form2** üzerindeki **label1**'e ulaşmak için **form2.label2.özellik** şeklinde kullanılır.
- Programınızda birden fazla form bulunacaksa ilk oluşturduğunuz form ana formdur ve program çalışmaya o formla başlar. Eğer başlangıçta çalışacak formu çalıştırmak istiyorsanız; **project_properties** menüleri ile açılan aşağıdaki pencerenin **startup object** listesinden istediğiniz formu seçebilirsiniz.



Yukarıdaki pencerede gördüğünüz gibi programlar çalışmaya formdan başlayabileceği gibi ismi main olan bir alt programdan da başlayabilir. Bunun için bir modül oluşturulur ve sub main ile başlayan bir alt program yazılır. Böylece program buradan başlar.

- Formlar ikiye ayrılır. Biri programlarda kullandığımız tek basına çalışan SDI (Single ocument Interface) formlardır.
- İkinciside MDI (multi document Intrface) formlardır. MDI formlar içinde kendisine bağlı formlar bulundurulur. Bu formlar (child form) MDI forma bağlıdır. MDI formla birlikte hareket ederler. Asagıda bir **MDI** form ve onun içinde bulunan **Child** formlar görülmektedir.

SDI FORMLAR

Tek basına Çalışan formlara **SDI** form denir. Normalde yeni bir projeye başlarken projede bulunan form ve daha sonra eklenecek formlara **SDI** formlardır. Bu form herhangi bir forma bağlı kalmaksizin kendi başlarına çalışırlar.

Properties - Form1

Form1 Form

Alphabetic | Categorized

(Name)	Form1
Appearance	1 - 3D
AutoRedraw	False
BackColor	&H80000000
BorderStyle	2 - Sizable
Caption	Form1
ClipControls	True
ControlBox	True
DrawMode	13 - Copy Pen
DrawStyle	0 - Solid
DrawWidth	1
Enabled	True
FillColor	&H00000000
FillStyle	1 - Transparent
Font	MS Sans Serif
FontTransparent	True
ForeColor	&H80000012
HasDC	True
Height	3600
HelpContextID	0
Icon	(Icon)
KeyPreview	False
Left	0
LinkMode	0 - None
LinkTopic	Form1
MaxButton	True
MDIChild	False
MinButton	True
MouseIcon	(None)
MousePointer	0 - Default
Moveable	True
NegotiateMenus	True
OLEDropMode	0 - None
Palette	(None)
PaletteMode	0 - Halftone
Picture	(None)
RightToLeft	False

Formun özellikleri

- Program içerisindeki adını belirler.
- Formdaki görüntüsünü belirler.
- Kendini otomatik olarak yenilemesini sağlar.
- Arka plan rengini belirler.
- Kenarlık biçimleri belirler.
- Başlık bilgisi tanımlar.
- Paint olayı ile ilgilidir.
- Denetim menüsünü iptal eder.
- Görünüm biçimi ile ilgilidir.
- Görünüm biçimi ile ilgilidir.
- Çizgi kalınlığıdır.
- Formun kullanılabilirliğini belirler.
- Kutuların iç boyama rengini ve desenini belirler.
- Nesnelerin içindeki çizgilerin şeklini belirler.
- Yazı tipini belirler.
- Yazı altında resim yazı varsa bunu gösterir.
- Açıklama yazısının rengini belirler.
- Nesnelerin yüksekliğini belirler.
- Yardım dosyasındaki konu numarasını belirler.
- Formun iconu
- Formun keypress olayını çalıştırması sağlar.
- Sola hizasını belirler.
- Max butonunun iptal edilmesi
- MDI form olarak kullanılacağını belirler.
- Min butonunu iptal eder.
- Nesnede hangi resmi alacağını belirler.
- Nesnede hangi şekli alacağını belirler.
- Kullanıcı tarafından taşınmayacağını belirler.

Form üzerinde göstereceği resmi tanımlar.

ScaleHeight	3195	Çizim yapılacak alanin yüksekligini ayarlar.
ScaleLeft	0	Çizim yapılacak alanin sol kenarini ayarlar.
ScaleMode	1 - Twip	Çizim yapılacak alanin ölçü birimini ayarlar.
ScaleTop	0	Çizim yapılacak alanin üst kenar ayarlar.
ScaleWidth	4680	Çizim yapılacak alanin genisligini ayarlar.
ShowInTaskbar	True	Görev çubugunda simgelestirmedir.
StartupPosition	3 - Windows De	Baslangiç konumudur.
Tag		Ek bilgidir.
Top	0	Üstten hizasini belirler.
Visible	True	Formda görünüp görünmeyecegini belirler.
WhatsThisButton	False	
WhatsThisHelp	False	
Width	4800	Formun genisligini belirler.
WindowState	0 - Normal	Formun çalışması üç şekilde olur.vbNormal vbMinimized,vbmaximized dir.

FORMA AIT BAZI ÖNEMLİ OLAYLAR

- 1. Private Sub Form_Activate()**:Olay formun aktif olduğu an çalışır.
- 2. Private Sub Form_Click()**:Olay Mouse un sol tusu ile forma cliklendiginde çalışır.
- 3. Private Sub Form_DblClick()**:Olay nesneye Mouse un sol tusla çift tiklandigi anda çalışır.
- 4. Private Sub Form_Deactivate()**: Olay formun aktif olmadığı anda çalışır. yani baska form aktif olduğu an.
- 5. Private Sub Form_Dragdrop(Source As control,X As Single,Y As Single)**:olay sürükleme olayı formda bittiginde çalışır. Source sürükleme olayinin basladigi nesneyi temsil eder. X ve Y Mouse un koordinat degerlerini verir.
- 6. Private Sub Form_Dragover(Source As control,X As Single,Y As Single,Stade As integer)**: Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır. State ise sürükleme olayinin hangi asamada olduğu degerini verir.Stade özelligi 3 deger tasir.

7. Private Sub Form_Gotfocus(): Olay, aktif nesne olduğunda çalışır. Clicklendiğinde aktif nesne olur.

8. Private Sub Form_Load(): Program çalışmaya başladığı an çalışır. Clicklendiğinde aktif nesne olur. Dolayısıyla program başlarken yapması gereken işler genellikle bu olaya yaptırılır.

9. Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer): Olay klavyenin tusuna basıldığı an çalışır. KeyCode klavyeden basılan tusun değerini tasir. Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir. Bunlar: 1-Shift, 2-Ctrl ve 4-Alt değerleridir. Beraber basıldıklarında gelen değer toplamlarından oluşur. Örneğin Ctrl+alt=6 dir.

10. Private Sub Form_KeyPress(KeyAscii As Integer): Olay keyDown olayından sonra çalışır. KeyAscii klavyeden basılan tusların ASCII karşılığını verir.

11. Private Sub Form _KeyUp (KeyCode As Integer, shift As Integer): Olay KeyPress olayından sonra, klavyeden basılan tus bırakıldıktan sonra çalışır.

12. private Sub Form_Lostfocus(): Olay nesneyi terk edildiği anda çalışır. Terk etme işlemi herhangi bir nesneye clicklendiğine veya TAB ile atlama Yapıldığında olur.

13. Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single): Formun üzerine mouse un herhangi bir tusu ile clicklendiğinde çalışır. Button basılan tusun değerini verir. Shift ise mouse ile birlikte klavyeden basılan tusun değerini verir. X ve Y ise mouse un bulunduğu koordinatlarını verir.

14. Private Sub Form _MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): Formun üzerinde mouse hareket ettirildiğinde çalışır.

15. Private Sub Form _MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single): Mouse un tuları bırakıldığı anda çalışır.

16. Private Sub Form_Paint(): Form a herhangi bir şey yazıldığında veya çizildiğinde çalışır.

17. Private Sub Form_Resize():Form boyutlandırıldığında çalışır.

18. Private Sub Form_Unload(Cancel As Integer): Form kapatılmak istendiğinde (Alt+F4 ile vs.) en son bu olay çalışır. Cancel kapatma istegini düzenler. 1 ise iptal eder.

19. Private Sub Form_QueryUnload(cancel As Integer,UnloadMode As Integer): Unload olayıyla aynı işi yapar ancak formun kimin tarafından kapatılmaya çalışıldığını da öğrenebilirsiniz. Ayrıca bu olay unload olayından önce meydana gelir ve burada **cancel=true** ile kapatma olayı iptal edilirse unload olayı meydana gelmez.

20. Private Sub Form_Terminate():Bir formun bütün elemanlarının bellekten durumunda oluşur.

Yapısı:private sub nesne_terminate()

FORMUN METHODLARI

Form üzerinde belli işlemlerin yapılmasını sağlar. Form nesnesinin methodları şunlardır:

Cls	Formun üstünü temizler.
DrawCircle	Daire çizer.
DrawLine	Çizgi çizer.
Hide	Gizler.
Move	Tasir
Pset	Bir noktanın rengini verir.
Refres	Yineler.
SetScale	Ölçü birimi düzenler.
SetFocus	Odaklanmayı sağlar.
Show	Formu gösterir.
TextHeight	Metin yüksekliği (bastırılacak)
TextWidth	Metin genişliği (bastırılacak)
Zorder	Grafik düzeyini belirler.
Printform	form üzerinde görülen bütün kontrollerin yazdırılmasını sağlar. .

CLS METHODU

Cls methodu ile print veya başka bir method ile yazılanlar silinir. **Cls** methodundan formdaki diğer nesneler etkilenmez. Eğer temizlemek istediğiniz form aktif değilse bunu program kodunda belirtmeniz gerekir. Aşağıdaki program kodu ile projedeki başka bir formun üzeri temizlenmektedir.

Yapısı:nesne.cls

```
Private Sub Form1_Click()  
Form2.Cls  
End Sub
```

DRAWLINE METHODU

Bu method ile aktif olan veya belirtilen formun üzerine çizgi ve dörtgen çizer.

Yapisi: nesne.**drawline** x1,y1,x2,y2[renk]

Genel olarak bu methodun kullanisi şöyledir:

Line(Baslama X, Baslama Y) - (Bitis X, Y), ÇizgiRengi,BF

Eger çizginin baslama yeri belirtilmezse baslama yeri olarak forma ait CurrentX ve CurrentY degiskenlerinin o anki içeriği dikkate alınır. Asagidaki program kodunda baslama yeri belirtilmedi. Program kodunu yazıp programı çalıştıralım ortaya çıkan formu hep birlikte görelim.

```
Private Sub Form_Click()  
DrawWidht=3  
Line- (5000,1500)  
End Sub
```



Görüldüğü gibi çizgi (0,0)koordinatlarından başladı.

DRAWCIRCLE METHODU

Bu methodla forma istenen çap, kalınlık ve renkte çember elipsler ya da bir eğri çizmeyi sağlar. Genel yazılışı şu şekildedir.

Circle(baslama x,baslama y),yarıçap,çizgi rengi

Yapisi: nesne.**drawcircle** x,y,yarıçap[renk,açı]

HIDE METHODU

Bir formu gizler ancak onu unload etmez.

Yapisi: nesne.**Hide**

Nesne bilgisi gizlenecek formu belirtir. Bu form gizlendiğinde ekranda görünmez. **Visible** özelliği **False** olur. Gizli bir formun kontrollerine kullanıcılar tarafından erişilemez.

Form1.Hide

MOVE METHODU(SOL,üst,GENISLIK, YÜKSEKLİK)

Formun veya herhangi bir kontrolün konumunu ve boyutlarını tek seferde değiştirir.

Kontrolün Left, Top, width, height özellikleriyle dört ayrı seferde yapacağınız işi bu methodla tek seferde yapabilirsiniz. Özelliklerle yapmanız durumunda kontrol dört defa şekil değiştirecek ve dolayısıyla daha ağır ve çirkin bir görüntü oluşacaktır.

Aşağıdaki her iki komut grubunda aynı işi yapar.

'özelliklerle

form1.left=100

form1.top=100

form1.width=screen.width-200

form1.left= screen.height-200

'Methodla

form1.move 100,100, screen.width-200, screen.height-200

PSET METHODU

Bu method ile istenen formun üstüne veya forma dahil edilen resim kutusuna bir nokta konulur.

Genel yazılışı

Pset(pozisyon x, pozisyon Y),nokta rengi

Pset() methoduna 3. parametre olarak renk kodu verilmezse o an için geçerli olan renk kodu esas alınır. 1. ve 2. parametreler ise noktanın formun neresine konulacağı belirlenir.

Forma konulacak noktanın yerini belirlemek için **Pset()** methoduna parametre olarak yazılan koordinat değerleri için sürekli olarak formun sol üst köşesi baz alınır. Eğer forma arka arkaya nokta eklerken baz olarak bir önceki noktanın yeri alınmak istenmiyorsa **pset()** methoduna **step** parametresi eklenir. Aşağıdaki program kodu buna bir örnektir.

Private Sub form_click()

Drawwidth=2

Pset(500,500)

For I=500 to 2500 step 10

Pset step(10,10)

NextI

End Sub

REFRESH METHODU

Form üzerinde yapılan degisikliklerin aninda gösterilmesini saglar.Normalde form veya baska bir kontrol(diskle ilgili kontroller hariç) üzerinde yapılan degisiklikler,kontrol windows'a geçtiginde Windows tarafından veya kontrol tarafından yapılır. Ancak yapılan degisikligin aninda ekranda görüntülenmesini istiyorsanız kontrole kendini yenilemesini söyleyebilirsiniz.

ÖRNEK:Bir liste kutusuna1000 tane eleman eklemek istediginizi düşünelim.

For I = 1 to 1000

List1.additem I*0,005/100+I*10.548/9,55

Next

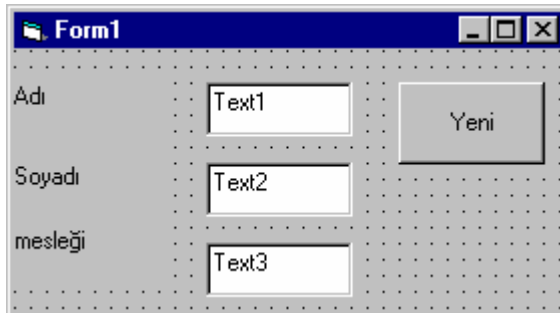
Bu kodla belli bir süre beklemeden sonra listeye elemanların tümün aynı anda eklendiğini görürsünüz. Çünkü For_next döngüsü bitmeden kontrol Windows'a geçeceği için listeye eleman eklendiği anda listede görülmeyecektir. Ancak döngü bittikten sonra kontrol windows'a geçecek ve elemanların tümü aynı anda listeye eklenecektir. Bu yöntemle döngü bitene kadar kullanıcı ekranda hiçbir degisiklik göremez ve hiçbir iş yapamaz.

SETFOCUS METHODU

Klavye kontrolünün **Setfocus** yapılan nesneye geçmesini saglar. Bir nesneye **setfocus** methodunu uygulayabilmek için o nesnenin **enabled** özelliğinin **true** olması gerekir. Aksi taktirde hata oluşacaktır.

Bu özellik daha kolay kullanılabilir arabirimler olusturmak için faydalidir.

ÖRNEK: Asagidaki gibi basit bir formumuz olsun.



Kullanıcı "yeni" düğmesini tıklayınca text kutularının içeriğini bosaltalım. Kullanıcının bu düğmeyi tıklamasıyla kontrol komut düğmesine geçecek ve orada kalacaktır. Halbuki "yeni" düğmesi tiklandıktan sonra tekrara aynı düğmenin tiklanması mantikli olmayacağı için kontrolün komut düğmesinde değil text1 kutusunda olması programın daha kullanilir olmasını sağlayacaktır.

```

'ÖRNEK:setfocus
private sub command1_click()
text1=" "
  text2=" "
  text3=" "
  text1.setfocus'kontrolü text1 kutusuna ver
End Sub

```

Bu örnek küçük ve basit olduğundan kontrolü bıraktığımız text1 kutusunun enabled özelliği kontrol etme ihtiyacı duymadık. Ancak program daha karışık ise klavye kontrolünün devredilmek istendiği nesne programın herhangi bir yerinde pasif yapılmış olabilir. Böyle durumlarda nesnenin pasif olmadığını kontrol ettikten sonra klavye kontrolünün verilmesi gerekir.

If text1.enabled then text1.setfocus

SHOW[SITIL] METHODU

Formun **visible** özelliğinin **true** yapılması gibidir. Ancak parametre ile kullanılırsa formun gösterim şeklini etkiler.

Burada **sitil** kullanılmazsa veya 0 kullanılırsa **visible=true** ile aynı işi yapar.

Sitil değeri 1 verilirse form **modal** olarak gösterilir. Form modal ise, form gizlenmeden veya kapatılmadan program içindeki diğer formlara ulaşım engellenir, yani form ekranda olduğu sürece kontrolü elinde tutar, programınızdaki diğer formlara geçişi engeller. (mesaj pencerelerinde olduğu gibi)

TEXTWIDTH VE TEXTHEIGHT METHODLARI

Bir forma yazılan sabit bilgiler, formun içerisinde o anda bulunan font ve punto değerlerine bağlı olarak belli bir genişliğe ve yüksekliğe sahip olur. Sabit bilgilerin genişliği **TextWidth**, yüksekliği ise **TextHeight** ile öğrenilebilir.

ZORDER MOD METHODU

Bu metod kullanıldığı nesneyi en öne veya en arkaya getirmek için kullanılır. Burada metodun 0 olması kontrolün diğerlerinin önüne, 1 olması arkasına getirilmesini sağlar.

Form üzerinde görülen bütün kontrollerin yazdırılmasını sağlar. Form ve **PictureBox** üzerinde ki metodlarla yapılan yazım ve çizimlerin de yazdırılabilmesi için **AutoRedraw** özelliklerinin **True** olması gerekir. Bu özellikler **True** ise Form üzerinde görülen menüler ve Form çerçevesi hariç her şey aynen yazıcıdan çıkar.

MDI FORMLAR

MDI formlar diger formlardan daha farkli bir yapiya sahiptir. MDI forma word programini örnek verebiliriz. **SDI (Single Document Interface)** uygulamalarinda tek bir form bulunur. **MDI (multiple Documents Interface)** uygulamalar ise çok sayida formdan oluşur.

Explorer_style arabirimlerinde ise iki kisimdan olusan bir pencere yapisi vardır. Genellikle hiyerarsik bir görünümü vardır.

MDI arabirimi tek bir MDI form ve içinde yer alan çok sayida alt formda (child) oluşur. Bu ana form "parent" form denir. Ana form içinde çok sayida alt form açilabilir. Bu çok sayida belge sayisi ya da form anlamina gelir. MDI formlar bir konteynerdir. Üzerinde komut düğmesi vb. kontroller kullanilamaz. Bu nedenle MDI formlar veri girişi vb rutin işlemler için kullanilamaz. Genellikle menü çubugu aracılığıyla yönetilir. MDI form içinde bir alt form simgelestirildiginde simgesi ana form içinde yer alır.

Bir projeye sadece bir MDI form eklenir. Ardindan projeye normal formlar eklenebilir. bu formlardan istenilen formun **MDIChild** özelliği **True** yapılarak bu form ana form içinde alt form yapılır. MDI form sisteminin en önemli özelliği çalışma zamanında alt formların kolayca açılıp kapatılmasıdır. Ayrıca açılan formun menü çubugu MDI formun menü çubugu olur.

Bir MDI Uygulama Yaratmak

1. Bir proje baslatılır.

2. **Project** menüsünden **Add MDIForm** komutu seçilir.

NOT: Bir uygulama sadece bir tane MDI forma sahip olabilir.

3. Projeye yeni bir form eklemek için **project** menüsünden **Add MDIForm** komutu seçilir.

4. Ardindan yeni formun özelliklerinden **MDI Child** özelliği **True** yapılır. Yeni form ekleme ve bunlardan istenildiği kadarının alt form (child) yapmak için bu işlem tekrar edilir.

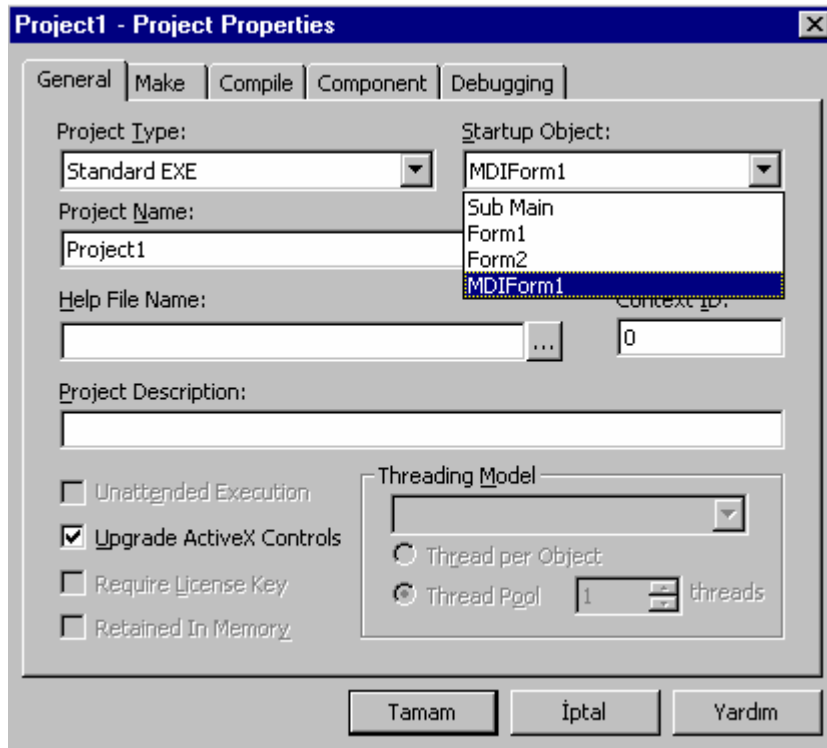
Çalışma Zamanında MDI Formun Özellikleri

- Bütün alt formlar MDI formun çalışma alanını kullanırlar. Bu nedenle onların hareketi MDI formu alanı ile kısıtlıdır.
- Bir alt form simgelestirildiginde simgesi MDI form içerisinde görülür. MDI form simgelestirildiginde ise görev çubugunda görülür. MDI formun simgelestirilmesi ve tekrara açılması durumunda alt formlar eski durumlarında (simge ya da normal büyüklükte) görülür.
- Bir alt form (child form) ekranı kapatıldığında onun **caption** bilgisi (başlığı) MDI formun başlığı olur.

- AutoShowChildren özelliği ile alt formlar yüklendiklerinde otomatik olarak gösterilirler ya da gösterilmezler. AutoshowChildren=True ya da false olarak düzenlenir.
- Aktif alt formun menü çubuğu varsa bu menü çubuğu MDI formun menü çubuğu olur.

ÖRNEK: bir örneği adım adım geliştirelim.

- Öncelikle yeni bir proje başlatın. Su anda programa ait bir formun projede olması gerekiyor.
- Yeni bir MDI formu da **project** menüsünden **Add MDI Form** seçeneği ile oluşturalım.
- Projede bulunan form1'in **MDIChild** özelliğini de **True** yaparak bir MDIChild oluşmasını sağlayalım.
- Ayrıca programın MDI form'dan başlaması içinde **project** menüsünden **project properties** ile açılan aşağıdaki pencerenin **General** kısmındaki **startup object** seçeneğini **MDIForm1** yapalım.



- Programı bu haliyle çalıştırdığınızda yalnız MDI form görülecektir. MDIChild formun da görülebilmesi için MDI formun load olayına aşağıdaki kodu ekleyin.

```
Private sub MDIForm _Load
```

```
    Form1.show
```

```
End Sub
```

- Bu MDIChild formların yenisini oluşturmak için, yeni formla oluşturulup **MDIChild** özellikleri **true** yapılabilir. Ancak eğer bu child formlar aynı kontrollere sahip olarsa bu yeni formlar elimizdeki MDIChild formdan türetilir. Bu iş için Dim degimini şu şekilde kullanacağız.

Dim formadi As New OrjinalForm

Formadi : Oluşturulacak yeni formun adı

OrjinalForm : Türetilcek yeni formun adı

Dim form As New form1

Yukarıdaki komutla form1 komutundan, form isimli yeni bir form oluşturulabilir. Bu komut o formu oluşturur ancak görüntülemeyi. Görüntülemek içinde

Form.show komutunu kullanacağız.

- Şimdi bu komutları kullanarak programımıza yeni formlar oluşturalım. Bunun için form1 üzerine **yeni form** caption'lu bir komut düğmesi ve bu düğmenin **click** koduna da

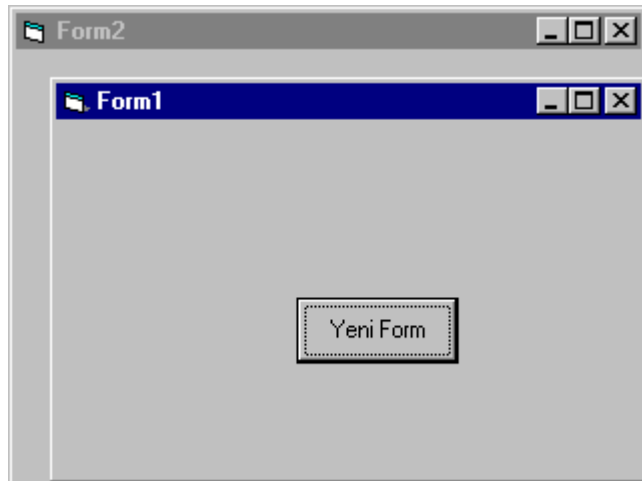
```
Private Sub command1_click ()
```

```
    Dim form As New form1
```

```
    Form1.show
```

```
End Sub
```

kodlarını yazalım. Programı bu haliyle çalıştırıp aşağıdaki gibi bir görüntü elde edebiliriz.



Burada görüldüğü gibi form1'den türetilen bütün formların **caption**'leri form1'dir. Bunu her form için arttırarak degistirelim. Bu is için **Forms.Count** özelligini kullanacagiz. Bu özellik programda bulunan bütün formlarin sayisini verir,her yeni form olusturulduunda bu sayi bir artacagindan olusan formlarin basliklarida farkli numaralarda olacaktir.Komut düğmesinin clik olayini su sekle çevirelim.

```
Private sub command1_click ()
    Dim form As new form1
    Form1.show
    Form.caption = "Form" & (forms.count _1)
End Sub
```

Forms.Count MDIForm dahil formlarin sayisini verdigi için eksigini alarak Child formlarin sayisini bulmus olduk. Böylece olusan her forma Form1,Form2 gibi basliklar verilecektir.

Alt Formlarin Kullanimi

Bir MDI uygulamanin tasariminda child formlarin yönetimi önemlidir. Child formlarin yüklenmesi,kaldirilmesi ve bu formlar arasinda geçis en önemli islemlerendir. Bir child form yüklenecegi zaman onun MDI formu (ana formu) otomatik olarak yüklenir. Buna karsin bir MDI form yüklendiginde child formlari otomatik olarak yüklenip gösterilemez. AutoshowChildren özelligi ile MDI child formlarin gizli olarak yüklenmesi saglanir ancak formlari göstermek için show formu kullanilir.

Formların Büyüklüğü ve Konumu

Bir MDI child form ilk açıldığındaki boyutu MDI ana formun büyüklüğüne göre değişir. Bir MDI Child formun ise boyutları ve konumu değiştirilebilir. Bu işlem için `BorderStyle=2` olmalıdır.

Alt Formdaki Bilgiler

Kullanıcı MDI uygulamadan çıkacağı zaman çalışmalarını kayıt etme fırsatına sahiptir. Bu işlem bir public değişken ile sağlanabilir.

Public kayıt edilmedi As Boolean

Bu değişken her child formda tanımlanır. Ardından Text1 metin kutusunda değişiklik yapıldığında bu değişkenin değeri True yapılır. Böylece text1 içinde değişiklik yapıldığı anlaşılar.

```
Private Sub Text1_change ()  
    KayitEdilmedi=True  
End Sub
```

Bununla birlikte child formun kayıt edilmesi durumunda ise KayitEdilmedi değişkenine False değeri verilir. Böylece bir daha kayıt edilmesine gerek kalmaz.

```
Sub mnuKaydet_click ()  
    ' text1 kaydet.  
    Dosya kaydet  
    ' degiskeni düzenle.  
    KayitEdilmedi = False  
End sub
```

Bu örnekte **kayıt Edilmedi** değişkeni bir flag(bayrak) gibi çalışır ve kullanıcının uygulamadan çıkıp çıkmadığına karar verir. **Unload** olayından önce **QueryUnload** olayı olustugu için çalışmaların kayıt edilmesi bu asamada yapılır.

```
Private Sub mnuÇikis_click()  
    ' MDI formdan çikis  
    ' unload MDI form  
    ' her form için Queryunload olayi olurur.  
    Unload frmMD  
    End  
End sub
```

```
Private Sub Form_Queryunload(cancel As integer, _  
    UnloadMode As integer)  
    If KayitEdilmedi then  
        ' kayıt et.
```



```

        DosyaKaydet ' kayıt yordamı
    End If
End Sub

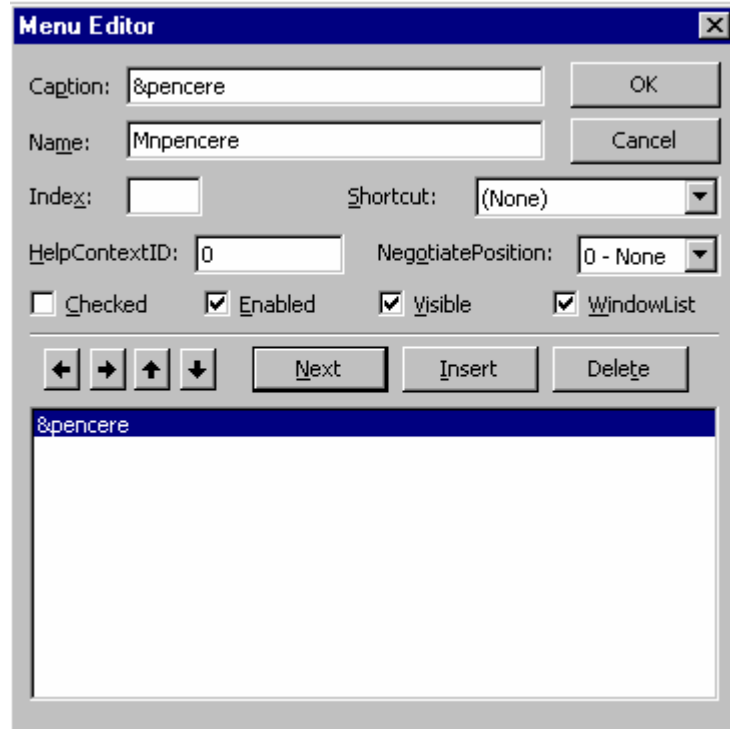
```

MDI FORMDA MENÜLER

MDIChild formların menülerinin **WindowsList** özelliği vardır. Bir menüye bu özellik verildiğinde MDIChild formların isimleri otomatik olarak bu menüye eklenir ve bu menüden seçilen form otomatik olarak bu menüye eklenir ve bu menüden seçilen form otomatik olarak aktif hale gelir. Bu işlemler için herhangi bir kod da gerekmez. Sadece ilgili menünün windowslist özelliğini menü editor penceresinde işaretlemek gerekir.

- MDI Formun bir çok özelliğinin diğer kontrollerin özelliklerinden bir farkı yoktur. Yalnız menülerde bir farklılık vardır. MDI formun menüleri olabilir ancak bu menüler bir MDIChild menü yokken menü çubuğunda görülür. Eğer bir MDIChild form varsa o Child formun menüleri menü çubuğu yerine geçer.

Bu özelliği Tools_menü editör menüsünden açabiliriz.



WINDOW(PENCERE) MENÜSÜ YARATMAK

Uygulamaların bir çoğunda Windows(pencere) menuşu vardır.Windows menuşunun amacı alt formlar arasında geçim yapmaktır.Bunun yanı sıra bir MDI uygulamada bulunan standart Tiler(Döşe) ve cascade(basamakla) işlemleri vardır.Işte visual Basic ile bir Window menüsü yaratmak ve onun içinde pencereleri dösemek için belli methodlar kullanılır.

Arrange metodu

Bu yöntem ile MDI form içindeki Child formlar düzenlenebilir. Su dört değerden birini alır:

Arrange	Anlami
0 vbCascade	Minimize durumunda olmayan bütün child formlar Basamaklanır.
1 vbTileHorizontal	Minimize durumunda olmayan bütün Child Formlar yatay olarak döşenir.
2 vbTileVertical	Minimize durumunda olmayan bütün Child Formlar dikey olarak döşenir.
3 vbarrangeIcons	Minimize durumunda olmayan bütün Child Formlar simgeleri yerleştirir

MDI form içindeki bütün formları yatay olarak dösemek için :

Mdiform1.arrange 1

Kullanılabileceği gibi

Mdiform1.arrange vbTileHorizontal

Seklinde sembolik ismi ile de kullanılabilir.

Properties - MDIForm1	
MDIForm1 MDIForm	
Alphabetic Categorized	
(Name)	MDIForm1
Appearance	1 - 3D
AutoShowChildren	True
BackColor	&H80000000C8
Caption	MDIForm1
Enabled	True
Height	3600
HelpContextID	0
Icon	(Icon)
Left	0
LinkMode	0 - None
LinkTopic	MDIForm1
MouseIcon	(None)
MousePointer	0 - Default
Moveable	True
NegotiateToolbars	True
OLEDropMode	0 - None
Picture	(None)
RightToLeft	False
ScrollBars	True
StartUpPosition	3 - Windows Defa
Tag	
Top	0
Visible	True
WhatsThisHelp	False
Width	4800
WindowState	0 - Normal

MDIFormun Özellikleri

- Program içerisindeki adini belirler.
- Görünümü belirler.
- MDIChild formlarin gizli olarak yüklenmesidir.
- Zemin rengidir.
- Baslik bilgisi tanımlanir.
- MDIFormun kullanabilirliğini belirler.
- MDIFormun yüksekliğini belirler.
- MDIFormun iconudur.
- Sola hizasini belirler.
- Nesne üzerinde hangi resmi alacagini belirler.
- Nesne üzerinde hangi sekli alacagini belirler.
- MDIFormun tasınabilirliğini iptal eder.
- Form üzerinde gösterecegi resim tanımlanir.
- Scall barların nerede çıkacağını belirler.
- Ek bilgidir.
- Üstten hizasini belirler.
- Görünürlük.
- Formun genişliğini belirler.
- Pencere durumu.

MDIFORMUN BAZI OLAYLARI

- 1. Private Sub MDIForm_Activate()**:Olay formun aktif olduğu an çalışır.
- 2. Private Sub MDIForm_Click()**:Olay Mouse'un sol tusu ile forma tıklendiğinde çalışır.
- 3. Private Sub MDIForm_DblClick()**:Olay nesneye Mouse un sol tustla çift tıklandığı anda çalışır.

4. Private Sub MDIForm_Deactivate(): Olay formun aktif olmadığı anda çalışır. yani başka form aktif olduğu an.

5. Private Sub MDIForm_Dragdrop(Source As control,X As Single,Y As Single):Olay sürükleme olayı formda bittiginde çalışır. Source sürükleme olayının başladığı nesneyi temsil eder. X ve Y Mouse un koordinat degerlerini verir.

6. Private Sub MDIForm_Dragover(Source As control,X As Single,Y As Single,Stade As integer): Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır. State ise sürükleme olayının hangi asamada olduğu degerini verir.Stade özelligi 3 deger tasir.

8. Private Sub MDIForm_Load(): Program çalışmaya başladığı an çalışır. Clicklendiğinde aktif nesne olur. Dolayisiyla pogram baslarken yapması gereken isler genellikle bu olaya yaptirilir.

9. Private Sub MDIForm_MauseDown(Button As Integer, Shift As Integer, X As Single, Y As Single): Formun üzerine mouse un herhangi bir tusu ile clicklendiğinde çalışır. Button basılan tusun degerini verir. Shift ise mouse ile birlikte klavyeden basılan tusun degerini verir. X ve Y ise mouse un bulunduğu koordinatlarını verir.

10. Private Sub MDIForm _MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): Formun üzerinde mouse hareket ettirildiğinde çalışır.

11. Private Sub MDIForm _MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single): Mouse'un tustari bırakıldığı anda çalışır.

12. Private Sub MDIForm_Resize():Form boyutlandırıldığında çalışır.

13.Private Sub MDIForm_Unload(Cancel As Integer): Form kapatılmak istendiğinde (Alt+F4 ile vs.) en son bu olay çalışır. Cancel kapatma istegini düzenler. 1 ise iptal eder.

14. Private Sub MDIForm_Terminate():Bir formun bütün elemanlarının bellekten durumunda oluşur.

MODULLER

VB kodlari modüller içinde saklanır. Üç çeşit modül vardır.

Form, Standart ve Class.

Basit uygulamalarda sadece bir form vardır ve tüm kod bu formun modülü içinde durur (uzantisi frm). Ancak programınız büyüdükçe ek formlar eklersiniz. Sonunda, bir çok farklı formda tekrar tekrar yazmak zorunda olduğunuz kodlar olduğunu farkedersiniz. Her forma aynı işi yapan kodları yazmak istemeyeceğiniz için ortak görevleri içinde barındıran bir ayrı modül yazarsınız. Bu ayrı modül bir standart modüldür (uzantisi bas). Zaman geçtikçe ortak kullanılan yordamları içeren bir modüller kütüphanesi kurarsınız.

Her standart, class yada form modülü sunları içerebilir:

Deklerasyonlar: constant, type, variable ve dynamic-link library (DLL) yordam deklasyonları standart, class yada form modüllerinin modül seviyesine yazılır.

Yordamlar: Sub, function yada Property yordamları

Form Modülleri

Form Modülleri (uzantisi FRM) VB'nin temelidir. İçlerinde ait oldukları formun tüm özellikleri vardır. Genel yordamlar, form düzeyi değişken deklasyonları, constantlar (değişmezler), type'lar (kullanıcının belirlediği veri türleri)

Standard Modülleri

Standard Modüller (uzantıları BAS) diğer modüllerini ortak kullandıkları yordamları ve değişken tanımlamalarını tutarlar. Global (tüm uygulamada geçerli) yada modül düzeyinde geçerli değişkenler, değişmezler, türler tutarlar. Standart modüller dikkatli yazılırlarsa diğer projelerde de kullanılabilirler.

Class Modülleri

Class Modülleri (uzantıları CLS) VB'nin nesne tabanlı programlama özelliğinin temelidir. Yeni nesneler yaratmak için class modülleri içine kod yazarsınız. Bu nesnelerin sizin belirlediğiniz özellikleri ve metodları olur.

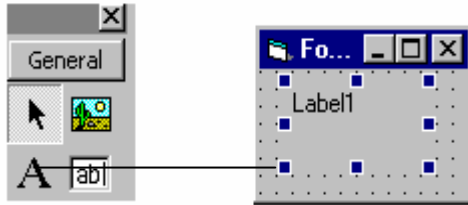
Microsoft Visual Basic 6.0

Aslında formlar üzerlerine kontroller yerleştirilmiş pencereler gösterebilen birer class modülüdür.

7-VISUAL BASIC KONTROLLERİ

A-DEFAULT COMPENENT LIST

1-Label(etiket): Form üzerindeki bilginin gösterilmesini sağlar. Bir textbox'a açıklamaya ya da hesaplanan bir degerin gösterilmesi için kullanılır.



Properties - Label1	
Label1 Label	
Alphabetic	Categorized
(Name)	Label1
Alignment	0 - Left Justify
Appearance	1 - 3D
AutoSize	False
BackColor	&H8000000F&
BackStyle	1 - Opaque
BorderStyle	0 - None
Caption	Label1
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	&H80000012&
Height	495
Index	
Left	240
LinkItem	
LinkMode	0 - None
LinkTimeout	50
LinkTopic	
MouseIcon	(None)
MousePointer	0 - Default
OLEDropMode	0 - None
RightToLeft	False

Label Nesnesinin Özellikleri

- Program içerisinde kullanılacak isim.
- Üzerinde bulunan yazının hizalanmasını sağlar.
- Form üzerinde görünüşü belirler.
- Genişliğini üzerinde bulunan yazıya ayarlar.
- Arka plandaki rengini belirler.
- Nesnenin gözüktüp gözükmeyeceğini belirler.
- Sekli forma gömülü veya normal ayarlar.
- Formda görünecek yazıdır.
- Veritabanına bağlantı için alan seçilmesini sağlar.
- Veritabanından alınacak bilginin formatını belirler.
- Veri tabanının tablosunu belirler.
- Sürükleme başladığında farenin alacağı şekli belirler.
- Sürüklemenin otomatik, manual olmasını belirler.
- Clicklenebilirliğini belirler. True ise click olayı çalışır.
- Yazı tipini belirler.
- Yazının rengini belirler.
- Nesnenin yüksekliğini belirler.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.

TabIndex	0	Kaçinci TAB da kendine ulasacagini belirler.
Tag		Herhangi bir mesaj saklamak için kullanilir.
ToolTipText		Nesnedeyken mesaj verdirmek için kullanilir.
Top	1320	Üstten hizasini belirler.
UseMnemonic	True	Form da görünüp görünmeyecegini belirler.
Visible	True	
WhatsThisHelpID	0	Nesnenin genisligini belirler.
Width	1215	
WordWrap	False	İçerdigi bilgiyi forma kelime bölerek ya da bölmeden getirecegini belirler.

LABEL NESNESİNİN BAZI ÖNEMLİ OLAYLARI

1-Private Sub Label1_Click(): Olay etikete mouse'un sol tusuna basildiginda çalisir. Ayni zamanda bir kontrol nesnesinin value degeri degistirince de click olayi meydana gelir.

2-Private Sub Label1_Change():Olay etiketin Caption'i degistigi anda çalisir.Kod içinde caption özelliginin degistirilmesi ya da bir DDE linkinin güncelleme yapilmasi change olayini olusturacaktır.

3-Private Sub Label1_DblClick():Olay etikete sol tusla çift tiklandigi anda çalisir.

4-Private Sub Label1_DragDrop(Source As Control, X As Single, Y As Single):Olay sürükleme olayi etikette bittiginde çalisir. Source sürükleme olayinin basladigi nesneyi temsil eder. X ve Y mouse'un koordinat degerlerini verir.

5-Private Sub Label1_DragOver(Source As Control,X As Single, Y As State As Integer):Olay sürükleme esnasinda mouse nesne üzerinden geçerken çalisir. State ise sürükleme olayinin hangi asamada oldugu degerini verir. State özelligi 3 deger tasir.

6- Private Sub Label1_Mousedown(Button As Integer,Shift As Integer, X As Single, Y As Single):Etiket üzerine mouse'nin herhangi bir tusu ile click'lendiginde çalisir.Button basilan tusun degerini verir.Bunlar:

<u>Anlami</u>	<u>Deger</u>	<u>Açıklama</u>
vbLeftbutton	1	Sol Tus
vbRightButton	2	Sag Tus
vbMiddleButton	4	Orta Tus

Shift ise mouse'la birlikte klavyeden basılan tusun degerini verir. Bunlar:

<u>Anlami</u>	<u>Deger</u>	<u>Açıklama</u>
vbShiftMask	1	Shift Tusu
vbCtrlMask	2	Ctrl Tusu
vbAltMask	4	Alt Tusu

X ve Y ise mouse'un bulunduğu koordinatlarını verir.

7-Private Sub Label1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): Etiket üzerindeyken çalışır.

8-Private Sub Label1_MouseUp(Button As Integer, X As Single, Y As Single): Mouse'un tusları bırakıldığı anda çalışır.

LABEL NESNESİNİN METHODLARI

Aligment: Yazının sola sağa veya ortaya yazılmasını sağlar.

Appearance: Bu özellik kontrolün üç boyutlu görünümünü ayarlar.

Autosize: Nesnenin boyutları içeriğinin boyutlarına göre yeniden ayarlanacaktır.

Backcolor: Nesnenin zemin rengini değiştirir. Bu özelliği oluşturan değer 7 digitlik bir hexadesimal sayıdır. (0 ile &HFFFFFF arasında) en düşük iki digit kırmızı, sonraki iki digit yeşil, ve diğer iki digit mavi rengin yoğunluğunu gösterir. Bu ikili digitlerin düşük değerlerde olmaları rengin koyuluğunu sağlar. En yüksek seviyeli digitin rengi 0 değilse windows'un system renklerinin kullanılacağını gösterir. properties penceresinde olmayan renkler, bu digitler ayarlanarak elde edilebilir. Kullanılacak renk sayısı 16.777.216 renktir.

Örneğin açık tonlu yeşili elde etmek için &H100FF00 değeri kullanılır.

Renkleri ayarlamak için **QBColor()** fonksiyonu da kullanılabilir. **QBColor()** fonksiyonuyla 16 temel renkten biri alınabilir.örneğin 2 nolu renk için asagidaki satir kullanılabilir.

Text1.BackColor=QBColor(2)

Backstyle: Nesnenin üzerinde bulunduğu konuma uymasini saglar.

BorderStyle:Nesnenin ekran üzerindeki sinirlarinin çerçeve seklini belirler.

Caption:Görünecek yazidir.

Datafield: Veritabaninin herhangi bir alana baglanmasini saglar.

Dataformat:Veritabanindaki bilgini formatini ayarlar.

Datasource:Veri tabanindaki tablotu belirler.

Dragicon:Sürükleme olayinda farenin alacagi sekli belirler.

Dragmode:Sürükleme olayinin otomatik veya manual olacagini belirler.

Enabled: Clicklenebilirliğini belirtir true ise click olayinda çalışır.

Font:VB'nin 5.0 versiyonu ile bütün font özelliklerini bir arada bulunduran ve tek seferde hepsini birden belirtmeye yarayan font nesnesi tanımlanmıştır.Bu nesnenin alt özellikleriyle tek tek ayarlama yapılabilecegi gibi tek seferde de baska bir font nesnesinin bütün özellikleri atanabilir.

Fontbold: True ise nesne için kullanılan yaziyi koyu yapar.

FontItalic:True ise nesne için kullanılan yazı egik yapar.

Fontname: Nesne için kullanılan yazının fontunu belirler.Bu font çalışma esnasında sistemde bulunan bir fontun numarası veya ismi verilerek değiştirilebilir.

FontSize:Nesne için kullanılan yazının puntosunu belirler.Maksimum 2048 olabilir.

Fontstrikethru: True ise nesne için kullanılan yazının ortasını çizer.

Fontunderline: True ise nesne için kullanılan yazının altını çizer.

Fontcolor: Yazının rengini belirler.

Height,width: Nesnenin boyunu ve enini belirler. Bu özellik değiştirilecek kontrolün boyutları ayarlanabilir.

Index: Aynı isimli birden çok nesne oluşturulmuşsa VB bu nesneleri bir dizi olarak görür. Index parametresi bu nesnenin dizideki kaçınıcı eleman olduğunu belirler, bu nesneye bu dizi indexiyle ulaşılır.

Index özelliğine bir sayı verilmişse o kontrole isminden sonra bu sayı verilerek ulaşılabilir. Örneğin index özelliği 2 olan bir text kutusunun text özelliğine

Text1(2).text

Satırı ile ulaşılabilir.

Bir kontrolü dizi olmaktan çıkarmak için ise index özelliğini silmeniz gerekir.

Left,Top: Nesnenin sol ve üst noktalarının koordinatlarını belirler. Bu koordinatlar ekran koordinatları olmayıp içinde bulunduğu nesneye bağlı koordinatlardır. Bir nesne **form**, **Picture box** veya **frame** içinde bulunabilir. İçinde bulunduğu nesnenin sol üst koordinatları 0,0 olmak üzere **left** ve **top** bu noktaya olan uzaklıktır. VB'de ölçü birimi olarak twip kullanılır. Ancak istenirse formun **scalemode** özelliği değiştirilerek ölçü birimi olarak piksel, veya metrik birimler verilebilir.

Mouseicon, pointer: Mouse göstergesinin nesne üzerine geldiğinde alacağı şekli belirler. 0 ile 15 arası bir sayıdır. 0 normal hali 11 bekleme hali vb.

Bu özelliğe 99 değerini vererek kendi tasarladığınız (ICO veya CUR dosyasından)

Bir kursörü seçebilirsiniz. Bu işlem için kendi tasarladığınız kursörün bulunduğu dosyanın adını **mouseicon** özelliğine girmeniz gerekir.

ÖRNEK: Aşağıdaki programı çalıştırarak Mouse şeklini görebilirsiniz. Her text kutusuna tıkladığınızda Mouse göstericinin şekli değişecek ve numarası text kutusunda görülecektir.

ÖRNEK : MousePointer

```
private Sub Text1_ click ()
```

```
    static i
```

```
    text1.text=str(i) + "numaralı Mouse pointer"
```

```
    text1.Mousepointer=i
```

```
    i=i+1
```

```
    if i = 16 then End
```

```
End Sub
```

Move: Formun veya herhangi bir kontrolün konumunu ve boyutlarını tek seferde değiştirir.

Kontrolün Left,Top,width,height özellikleriyle dört ayrı seferde yapacağınız işi bu methodla tek seferde yapabilirsiniz. Özelliklerle yapmanız durumunda kontrol dört defa şekil değiştirecek ve dolayısıyla daha ağır ve çirkin bir görüntü oluşacaktır.

Aşağıdaki her iki komut grubu da aynı işi yapar.

'özelliklerle

form1.left= 100

form1.top= 100

form1.width=screen.width-200

form1.left= screen.height-200

'Methodlar

form1.move 100,100, screen.width-200, screen.height-200

Parent: Nesnenin üzerinde bulunduğu form'a ulaşmayı sağlar. Nesnenin isminden sonra verilen **parent** komutu o nesneyi değil nesnenin üzerinde bulunduğu formu temsil eder.

Refresh: Form üzerinde yapılan değişikliklerin anında gösterilmesini sağlar.

TabIndex:Form üzerinde kullanıcının ulaşabileceği bir nesnenin bir tabindexi vardır. Tabindex kullanıcının **tab** tuşuyla kontroller arasında dolasırken bu kontrollerin sıralamasını belirler.

Tag:Bu özelliğin Visual Basic için hiçbir anlamı yoktur. Kullanıcı bu parametreyi bir değişken gibi kullanabilir. Bu özellik bir string değişken olarak kullanılmalıdır.

ToolTipText:Windows altında çalışan bir çok modern programda program öğelerinin birinin üzerinde fare ile kısa bir süre durduğunuzda açılan bir balonla onun ne işe yaradığını yazan mesajlar görmüşsünüzdür. VB'de de bu iş o kontrolün **tooltipText** özelliğine atanacak metinle yapılır. Bu özelliğe verdiğiniz metin, kullanıcının o kontrol üzerinde kısa bir süre durmasıyla, küçük bir kutu içerisinde gösterilecektir.

Örnek olarak aşağıdaki formu hazırlayın . text1 ve text2 ye girilen notların ortalamasını label4 içerisinde gösterelim kullanıcı text1 üzerinde durunca 1.sınav notunuzu giriniz, text2 üzerinde durunca 2.sınav notunuzu giriniz ve label4 üzerinde durunca da 2. sınav ortalaması diye yazsın.

```

Project1 - Form1 (Code)
Text2 Change

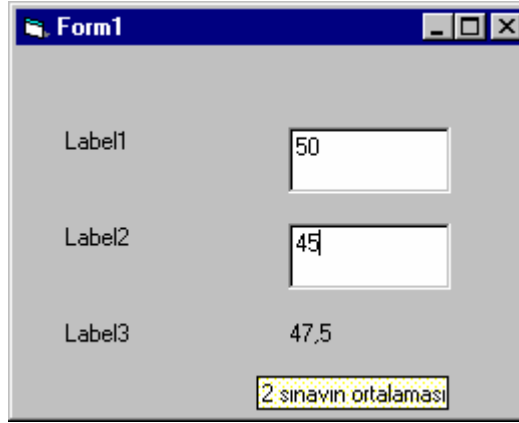
Private Sub Form_Load()
    Label4.ToolTipText = "2 sınavın ortalaması"
    Text1.ToolTipText = "1. sınavın notunu giriniz"
    Text2.ToolTipText = "1. sınavın notunu giriniz"
End Sub

Private Sub Text1_Change()
    'text kutusunun içeriği değiştiğinde ortalamayı bul
    Label4 = (Val(Text1) + Val(Text2)) / 2
End Sub

Private Sub Text2_Change()
    'text kutusunun içeriği değiştiğinde ortalamayı bul
    Label4 = (Val(Text1) + Val(Text2)) / 2
End Sub

```

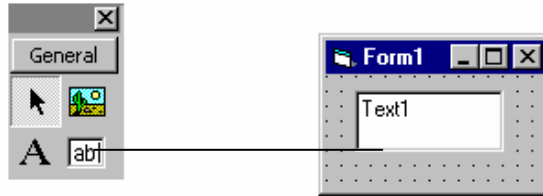
Program çalıştıktan sonra kontrollerin üzerinde fare ile kısa bir süre durursanız, aşağıdaki gibi o kontrole ilgili açıklamayı gösteren bir kutu açılacaktır.



Visible: True ise nesne görülür, false yapılırsa görülmez belli şartlar gerçekleştiğinde gözükmesini istediğiniz kontrollerin visible özelliğini kullanabilirsiniz.

Zorder: Bu method kullanıldığı nesneyi en öne veya en arkaya getirmek için kullanılır. Burada metodun 0 olması kontrolün diğerlerinin önüne, 1 olması arkasına getirilmesini sağlar.

2-Textbox(metin kutusu): Bilgi girişi için kullanılan bir kontroldür. Programın çalışması veya tasarımı esnasında metin, kullanıcı tarafından değiştirilebilmesinin yanında kesme, kopyalama, yapıştırma, aşağı_yukarı, sağa_sola ilerleyebilme ve birden fazla satır girebilme gibi özelliklere de sahiptir.



Properties - Text1

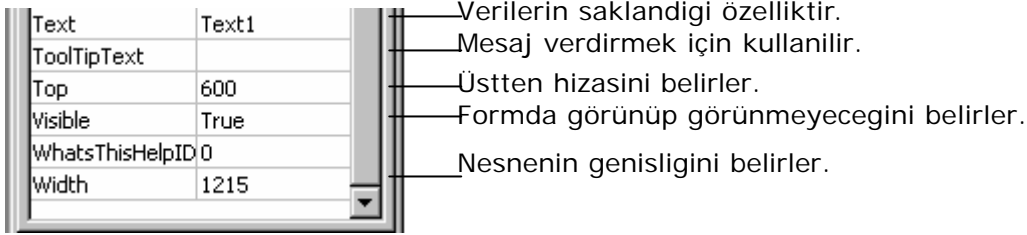
Text1 TextBox

Alphabetic | Categorized

(Name)	Text1
Alignment	0 - Left Justify
Appearance	1 - 3D
BackColor	<input type="checkbox"/> &H80000000
BorderStyle	1 - Fixed Single
CausesValidation	True
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	<input checked="" type="checkbox"/> &H80000000
Height	495
HelpContextID	0
HideSelection	True
Index	
Left	2040
LinkItem	
LinkMode	0 - None
LinkTimeout	50
LinkTopic	
Locked	False
MaxLength	0
MouseIcon	(None)
MousePointer	0 - Default
MultiLine	False
OLEDragMode	0 - Manual
OLEDropMode	0 - None
PasswordChar	
RightToLeft	False
ScrollBars	0 - None
TabIndex	4
TabStop	True
Tag	

Textbox nesnesinin özellikleri

- Program içerisindeki adını belirtir.
- Girilen metni hizalar.
- Formdaki görüntüsünü belirler.
- Arka plan rengini belirler.
- Bulunduğu konuma uymasını sağlar.
- Baska kontrolde validatenin çalışmasını kontrol eder.
- Herhangi bir alana bağlanmasını sağlar.
- Veri biçimini belirler.
- Veritabanındaki bağlanılacak tabloyu belirler.
- Sürükleme olayında farenin alacağı şekli belirler
- Sürüklemenin otomatik ,manual olmasını sağlar
- Clicklenebilirliğini belirler.true click olayı çalışır.
- Yazı tipini belirtir.
- Yazı rengini belirler.
- Nesnenin yüksekliğini belirler.
- Yardım dosyasındaki konu numarası belirlenir.
- Seçilen kısmın gizlenmesini sağlar.
- Indexli kullanımda indexini belirtir.
- Sola hizasını belirler.
- Veri girişine imkan verilip verilmemesini belirtir.
- Maximum karakter sayısını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.
- Birden fazla satır girişine izin verilmesini belirtir
- Verileri belirlenen karakter olarak gösterir.
- Scroll barların nerede çıkacağını belirler.
- Kaçıncı TABda kendine ulaşacağını belirler.
- True;tab tusu ile ulaşır.False tab ile ulaşamaz.
- Herhangi bir mesaj saklamak için kullanılır.



TEXTBOX NESNESİNİN BAZI ÖNEMLİ OLAYLARI

1. Private Sub Text1_Change():Olay Text özelliğinin içeriği değiştiğinde çalışır.

2. Private Sub Text1_Click():Olay metin kutusuna mouse'un sol tusuna basıldığında çalışır.

3. Private Sub Text1_DbClick():Olay metin kutusuna sol tusla çift tıklandığında çalışır.

4. Private Sub Text1_DragDrop(Source As Control, X As Single, Y As Single):Olay sürükleme olayı metin kutusunda bittiğinde çalışır.Source sürükleme olayının başladığı nesneyi temsil eder. X ve Y mouse'un koordinat değerlerini verir.

5. Private Sub Text1_DragOver(Source As Control, X As Single, Y As Single, State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürükleme olayının hangi aşamada olduğu değerini verir.State özelliği üç değer tasir.

6. Private Sub Text1_setfocus():Klavye kontrolün **setfocus** yapılan nesneye geçmesini sağlar. Bir nesneye **setfocus** methodunu uygulayabilmek için o nesnenin **enabled** özelliğinin **True** olması gerekir. Aksi takdirde hata oluşacaktır.

7. Private Sub Text1_GotFocus():Olay nesne,aktif nesne olduğunda çalışır.Clicklendiğinde, TAB tusuyla seçildiğinde aktif nesne olur.

8. Private Sub Text1_KeyDown(Key Code As Integer, Shift As Integer):Olay veri girişi esnasında klavyenin tusuna basıldığı an çalışır.KeyCode klavyeden basılan tusun değerini tasir.Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir.Bunlar: 1-Shift, 2-Ctrl ve 4-Alt

değerleridir. Beraber basıldıklarında gelen değer toplamlarından oluşur. Örneğin Ctrl+Alt=6'dır.

9. Private Sub Text1_KeyPress(KeyAscii As Integer):Olay veri girişi esnasında KeyDown olayından sonra çalışır. KeyAscii klavyeden basılan tuşların ASCII karşılığını verir.

10. Private Sub Text1_KeyUp(KeyCode As Integer, Shift As Integer):Olay KeyPress olayından sonra, klavyede basılan tuş bırakıldıktan sonra çalışır.

11. Private Sub Text1_LostFocus():Olay metin kutusu terk edildiği anda çalışır.

12. Private Sub Text1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single):Metin kutusu üzerine Mouse'un herhangi bir tuşu ile clicklendiğinde çalışır. Button basılan tuşun değerini verir. Shift ise Mouse ile birlikte klavyeden basılan tuşun değerini verir. X ve Y ise mouse'un bulunduğu koordinatlarını verir.

13. Private sub Text1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single):Metin kutusu üzerindeyken çalışır.

14. Private Sub Text1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single):Mouse tuşları bırakıldığı anda çalışır.

TEXTBOX NESNESİNİN METHODLARI

Alignment: Yazının sola sağa veya ortaya yazılmasını sağlar.

Appearance: Bu özellik kontrolün üç boyutlu görünümünü ayarlar.

Backstyle: Nesnenin üzerinde bulunduğu konuma uymasını sağlar.

BorderStyle: Nesnenin ekran üzerindeki sınırlarının çerçeve şeklini belirler.

Caption: Görünecek yazıdır.

Index: Aynı isimli birden çok nesne oluşturulmussa VB bu nesneleri bir dizi olarak görür. Index parametresi bu nesnenin dizideki kaçinci eleman oldugunu belirler. Bu nesneye bu dizi indexiyle ulasilir.

Index özelligine bir sayi verilmissse o kontrole isminden sonra bu sayi verilerek ulasilabilir. Örneğin index özelligi 2 olan bir text kutusunun text özelligine

Text1(2).text

Satiri ile ulasilabilir.

Bir kontrolü dizi olmaktan çıkarmak için ise index özelligini silmeniz gerekir.

CausesValidation: Text kutusu kontrolünü kaybettiğinde yani kullanıcı text kutusundan baska bir kontrole fare veya klavye yardimi ile geçtiğinde validate olayinin çalışıp çalışmayacağını belirler. Normalde bu özellik true'dir ve **validate** olayi çalışir. Bu olaya yazacağınız kodla kullanıcının girdigi bilgiyi anında kontrol edebilir ve gerekiyorsa uyarabilirsiniz.

ForeColor: Yazının rengini belirler. yukarıda anlatılan renk bileşimleri burada da geçerlidir.

HelpContextID: Kontrole ilgili yardım dosyasındaki konu numarası bu özellik ile belirlenir. Programınıza ait yardım dosyası varsa **project_properties** menüleri ile açılan aşağıdaki pencerenin **Help File name** kutusuna bu yardım dosyasının ismi verilir.

Help dosyası belirlendikten sonra help dosyasındaki hangi konunun hangi kontrole karşılık geldiği **HelpContextID** özelligi ile belirlenir. Kullanıcı bu kontroldeyken F1 tusuna basıldığında belirledığınız yardım dosyası açılır ve bu özellige numarasını verdığınız konu gösterilir.

HideSelection: True ise text kutusunda seçilen metnin, kontrol baska bir nesneye geçtiğinde seçilen kısmın gizlenmesini sağlar.

False ise kontrol baska bir nesneye geçse de seçilen kısmın görülmesini sağlar.

hWnd: Windows altında çalışan kontrollerin **handle** diye adlandırılan tanıtıcı bir numarası vardır. Windows bu numarayı kullanarak kontrolleri tanır. Visual basic'te bu numara nesnenin **hWnd** özelligi ile programın çalışması esnasında öğrenilebilir. **hWnd** özelligi genellikle Windows API çağrılarında gereklidir.

MaxLength: Text kutusuna girebilecek maximum karakter sayısını belirler. 0 verilirse bu sınır kaldırılır. Verilen sınır sayısı kadar karakterden fazlasını kabul etmez.

Kullanıcının belli sayıdan daha fazla karakter girmesini önlemek için bu özellik kullanılır.Örneğin kullanıcının 3 harften daha fazla giriş yapmasını önlemek için bu özelliğe 3 değeri verilebilir.

Maxlength özelliği sadece kullanıcının girebileceği karakter sayısını sınırlar.Program kodu ile yapılan atamalarda bu sınır asılabilir.

SelLength,selstart:Windows altında bir çok işlem için bloklama (seçme)yapmak gerektiğini biliyorsunuz. Kullanıcı text kutularında seçme işlemi yapabilir.Kullanıcının seçtiği kısmı öğrenmek için bu ikili özellik kullanılabilir.Ayrıca bu iki özelliği kullanarak siz de program kodu ile seçme işlemini yaptırabilirsiniz.

Bu özellikler seçmenin text içinde başlayacağı konumu ve uzunluğunu belirler.

örneğin ilk 3 harfi seçtirmek için aşağıdaki satırlar kullanılabilir.

Text1.selstart=0

Text1.sellength=3

Tümünü seçtirmek için ise aşağıdaki satırlar kullanılabilir.

Text1.selstart=0

Text1.sellength=len(text1)_1

SelText:Seçilmiş metin bu özellik ile öğrenilip değiştirilebilir.**Seltext** özelliğine yapılan atamalarda,eger seçilmiş bir kısım varsa o kısım silinir,yoksa kursörün bulunduğu noktaya sikistirir.

Text1.seltext="merhaba"

Satiri ile text 1 içinde seçilmiş olan kısmın yerine merhaba yazılacaktır.

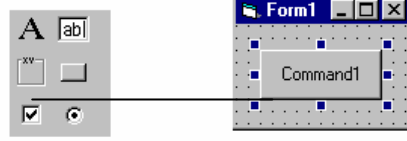
Setfocus:klavye kontrolün **setfocus** yapılan nesneye geçmesini sağlar.Bir nesneye **setfocus** metodunu uygulayabilmek için o nesnenin **enabled** özelliğinin **True** olması gerekir. Aksi takdirde hata oluşacaktır. Bu özellik daha kolay kullanılabilir arabirimler oluşturmak için faydalıdır.

Tabstop:True ise kullanıcı bu nesneye tab tuşu ile oluşabilir. **False** ise tab tuşuyla bu kontrol üzerine gelinmez ,Mouse ile yazılım yoluyla veya varsa kısayol tuşu ile gelebilir.

Yogun bilgi girişi gerektiren formlarda sık kullanılmayacak kontrollerin tabstop özelliklerini false yaparsanız kullanıcının bilgi girişi daha kolay olacaktır.Çünkü genelde yoğun bilgi girişi olan yerlerde tab tuşu sık kullanılır.Gereksiz yere ekrandaki bütün kontrolleri dolasmamak için bazılarının bu özelliğini false yapmak gerekir.

3-Command button(komut düğmesi):

Bir islemin baslatilmasinda kullanimi bir kontroldür.



Properties - Command1	
Command1 CommandButton	
Alphabetic Categorized	
(Name)	Command1
Appearance	1 - 3D
BackColor	&H8000000F&
Cancel	False
Caption	Command1
CausesValidation	True
Default	False
DisabledPicture	(None)
DownPicture	(None)
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
Height	495
HelpContextID	0
Index	
Left	240
MaskColor	&H00C0C0C0&
MouseIcon	(None)
MousePointer	0 - Default
OLEDropMode	0 - None
Picture	(None)
RightToLeft	False
Style	0 - Standard
TabIndex	0
TabStop	True
Tag	
ToolTipText	
Top	240
UseMaskColor	False
Visible	True
WhatsThisHelpID	0
Width	1215

Command button nesnesinin özellikleri

- Program içerisindeki adini belirtir.
- Formdaki görüntüsünü belirler.
- Style graphicalken buton rengini belirler.
- ESC tusuyla click olayinin çalışmasını sağlar.
- Üzerinde görünecek yaziyi belirler.
- Baska kontrolde validate çalışmasını kontrol eder.
- Enterde click olayinin çalışmasını sağlar.
- Kullanilamaz iken resmin ne olacağını belirler.
- Butona basılması resmin ne olacağını belirtir.
- Sürüklemeye farenin alacağı şekli belirtir.
- Sürüklemenin otomatik,manual olmasını sağlar.
- Clicklenebilirliğini belirtir.
- Yazi tipini belirler.
- Nesnenin yüksekliğini belirler.
- Yardım dosyasındaki konu numarası belirlenir.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.
- Butonun göstereceği resmi belirler.
- Butonun resim alıp almayacağını belirler.
- Kaçıncı tab da kendine ulaşacağını belirler.
- TAB ile seçilmesi sonlandırılır.
- Herhangi bir mesaj saklamak için kullanılır.
- Nesne üzerinde mesaj verdirmek için kullanılır.
- Üstten hizasını belirler.
- Formda görünüp görünmeyeceğini belirler.
- Nesnenin genişliğini belirtir.

COMMAND BUTTON NESNESİNİN BAZI ÖNEMLİ OLAYLARI

1. Private Sub Command1_Click():Olay butona mouse'un sol tusuna basıldığında çalışır.

2. Private Sub Command1_DragDrop (Source As Control, X As Single, Y As Single):Olay sürükleme olayı butonda bittiginde çalışır.Source sürükleme olayının başladığı nesneyi temsil eder.X ve Y Mouse'un koordinat değerlerini verir.

3. Private Sub Command1_DragOver(Source As Control,X As Single, Y As Single, State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürükleme olayının hangi asamada olduğu değerini verir.State özelliği 3 değer tasir.

4. Private Sub Command1_GotFocus():Olay, nesne aktif nesne olduğunda çalışır. Clicklendiğinde ve TAB tusuyla seçildiğinde aktif nesne olur.

5. Private Sub Command1_KeyDown(KeyCode As Integer, Shift As Integer):Olay klavyenin tusuna basıldığı an çalışır.KeyCode Klavyeden basılan tusun değerini tasir.Shift Ctrl, Alt ve Shift tuslarına karsilik deger üretir.Bunlar: 1-Shift, 2-Ctrl ve 4-Alt degerleridir.Beraber basildiklarında gelen deger toplamlarından olusur. Örneğin Ctrl +Alt=6 dir.

6. Private Sub Command1_KeyPress(KeyAscii As Integer):Olay KeyDown olayından sonra çalışır.KeyAscii Klavyeden basılan tusların ASCII karsiligini verir.

7. Private Sub Command1_KeyUp(KeyCode As Integer Shift As Integer):Olay KeyPress olayından sonra, Klavyeden basılan tus bırakıldıktan sonra çalışır.

8. Private Sub Command1_LostFocus():Olay buton terk edildiği anda çalışır. Terk etme islemi herhangi bir nesneye cliklendiğine veya TAB ile atlama yapıldığında olur.

9. Private Sub Command1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single):Buton üzerine mouse'un herhangi bir tusu ile clicklendiğinde çalışır. Button basılan tusun degerini verir. Shift ise mouse'la birlikte klavyeden basılan tusun degerini verir. X ve Y ise mouse'un bulunduğu koordinatlarını verir.

10. Private Sub Command1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): Command Button üzerinde Mouse hareket ettirildiğinde çalışır.

11. Private Sub Command1_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As Single): Mouse'un tustari bırakıldığı anda çalışır.

COMMANT BUTTON NESNESİNİN METODLARI

Cancel:Default özelliğine benzer. **ESC** tusu ile aktif hale gelir. **cancel** özelliği **True** olan bir komut düğmesinin bulunduğu formda **ESC** tusuna basılmasıyla bu komut düğmesi aktif yapılmış olur.

Caption: Komut düğmesi içerisinde yazılacak olan mesajı içerir. Diğer nesnelerin **caption** özelliklerinde olduğu gibi altı çizili harf o nesnenin kısa Yol tusudur ve alt tusuyla birlikte altı çizili harfe basılması durumunda nesne aktif hale gelir.

Altı çizili harfi (kısayol tusu) belirlemek için **caption**'da o harfin önüne **&** konur.

Default: Bir komut düğmesinin **Default** özelliği **true** ise o düğmenin bulunduğu form üzerinde **enter**'e basılması durumunda o düğme tiklanmış gibi olur.

Picture, DisabledPicture, DownPicture: Eğer komut düğmesinin **style** özelliğine 1 verilerek resimli bir komut düğmesi olacağı belirtilmişse bu üç özellik ile komut düğmesinin üzerinde bulununca resim belirlenebilir.

Picture: Bu özellik ile belirlenen resim komut düğmesinin aktif(enabled) özelliği false)

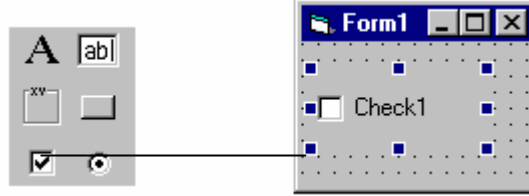
DisabledPicture özelliği ile belirlenen resim komut düğmesinin pasif(enabled özelliği false) iken

Downpicture özelliği ile de ut düğmesinin basılı iken gösterilecek resim belirlenir.

Style: Bu özellik komut düğmesinin yazılımı resimlili olacağını belirler. 0 ise komut düğmesini üzerinde **caption** özelliği ile belirlenen yazı bulunur. 1 ise komut düğmesinin üzerinde **Picture** özelliği ile belirlenen resim bulunur.

4-Check Box(isaret kutusu):

Belli seçeneklerin seçili olup olmadıklarını belirtmeye yarar. Genellikle bir grup seçenekten bir ya da çoğunun seçilmesiyle işler.



Properties - Check1	
Check1 CheckBox	
Alphabetic Categorized	
(Name)	Check1
Alignment	0 - Left Justify
Appearance	1 - 3D
BackColor	&H80000000
Caption	Check1
CausesValidation	True
DataField	
DataFormat	
DataMember	
DataSource	
DisabledPicture	(None)
DownPicture	(None)
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	&H80000012
Height	495
HelpContextID	0
Index	
Left	120
MaskColor	&H00C0C0
MouseIcon	(None)
MousePointer	0 - Default
OLEDropMode	0 - None
Picture	(None)
RightToLeft	False
Style	0 - Standard
TabIndex	0
TabStop	True

Check box nesnesinin özellikleri

- Program içerisindeki adını belirler.
- Üzerinde görünen açıklamayı hizalar.
- Formdaki görüntüsünü belirler.
- Arka plan rengini belirler.
- Üzerindeki açıklamanın girildiği özelliktir.
- Kontrolde validate çalışmasını kontrol eder.
- Herhangi bir alana bağlanılmasını sağlar.
- Veri biçimini Belirler.
- Veritabanındaki bağlanılacak tabloyu belirler.
- Kullanılmaz iken resmin ne olacağını belirler.
- Clicklendiğinde resmin ne olacağını belirler.
- Sürüklemeye de fare ile alınacağı şekli belirler.
- Sürüklemenin otomatik, manual olmasını sağlar.
- Clicklenebilirliğini belirler.
- Yazı tipini belirler.
- Açıklama yazısının rengini belirler.
- Nesnenin yüksekliğini belirler.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.
- Üzerinde görünecek resmi belirler.
- Kontroldeyken resim alınıp alınmamasını sağlar.
- Kaçınca atamada kendine ulaşacağını belirler.
- TAB ile seçilmesi sonlandırılır.

Tag		Herhangi bir mesaj saklamak için kullanılır.
ToolTipText		Mesaj verdirmek için kullanılır.
Top	240	Üstten hizasını belirler.
UseMaskColor	False	
Value	0 - Unchecked	Seçili olup olmasını belirler
Visible	True	Görünüp görünmeyeceğini belirler.
WhatsThisHelpID	0	
Width	1215	Nesnenin genişliğini belirler.

CHECKBOX NESNESİNİN BAZI ÖNEMLİ OLAYLARI

1. Private Sub Check1_Click():Olay isaret kutusuna Mouse'un sol tusuna basıldığında çalışır.

2. Private Sub Check1-DragDrop(Source As Control, X As Single, Y As Single): Olay sürükleme olayı checkbox da bittiginde çalışır. Source sürükleme olayının başladığı nesneyi temsil eder. X ve Y mouse'un koordinat değerlerini verir.

3. Private Sub Check1_DragOver(Source As Control, X As Single, Y As Single, State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır. State ise sürükleme olayının hangi aşamada olduğu değerini verir.State özelliği 3 değer tasir.

4. Private Sub Check1_Gotfocus():Olay, nesne aktif nesne olduğunda çalışır.Clicklendiğinde ve TAB tusuyla seçildiğinde aktif nesne olur.

5. Private Sub Check1_ Keydown (keycode Integer,Shift As Integer): Olay klavyenin tusuna basıldığı an çalışır. Keycode klavyeden basılan tusun değerini tasir. Shift Ctrl,Alt ve shift tuslarına karsilik deger üretir. Bunlar: 1-Shift,2-Ctrl ve 4-Alt degerleridir.Beraber basildiklarinda gelen deger toplamlarindan olusur.Örneğin Ctrl+Alt=6 dir.

6. Private Sub Chenck1_Keypress(KeyAscii As Integer):Olay keydown olayından sonra çalışır.KeyAscii basılan tusların ASCII karşılığını verir.

7. Private Sub Chenk1_KeyUp(KeyCode As Integer,Shift As Integer):Olay KeyPres olayından sonra,klavyeden basılan tus bırakıldıktan sonra çalışır.

8. Private Sub Chenck1_LostFocus():Olay Chenck Box nesnesi terk edildiği anda çalışır. Terk etme işlemi herhangi bir nesneye tıklandığına veya TAB ile atlama yapıldığında olur.

9. Private Sub Chenck1_MouseDown(Button As Integer,Shift As Integer,X As Single Y As Single):Check Box nesnesi üzerinde mouse'un herhangi bir tusu ile tıklandığında çalışır. Button basılan tusun değerini verir. Shift ise mouse ile birlikte klavyeden basılan tusun değerini verir. X ve Y ise mouse'un bulunduğu koordinatlarını verir.

10. Private Sub Chenck1_MouseMove (Button As Integer,Shift As Integer,X As Single Y As Single):Chenck Box nesnesi üzerinde mouse hareket ettirildiğinde çalışır.

11. Private Sub Chenk1_MouseUp(Button As Integer,Shift As Integer, X As Single Y As Single):Mousenin tısları bırakıldığı anda çalışır.

CHECKBOX NESNESİNİN METHOTLARI

Aligment:Yazının sola sağa veya ortaya yazılmasını sağlar.

Appearance:Bu özellik kontrolün üç boyutlu görünümünü ayarlar.

Backstyle: Nesnenin üzerinde bulunduğu konuma uymasını sağlar.

BorderStyle:Nesnenin ekran üzerindeki sınırlarının çerçeve şeklini belirler.

Caption:Görünecek yazıdır.

index: Aynı isimli birden çok nesne oluşturulmuşsa VB bu nesneleri bir dizi olarak görür. Index parametresi bu nesnenin dizideki kaçınıcı eleman olduğunu belirler,bu nesneye bu dizi indexiyle ulaşılır.

Index özelliğine bir sayı verilmişse o kontrole isminden sonra bu sayı verilerek ulaşılabilir. Örneğin index özelliği 2 olan bir text kutusunun text özelliğine

Text1(2).text

Satırı ile ulaşılabilir.

Bir kontrolü dizi olmaktan çıkarmak için ise index özelliğini silmeniz gerekir.

parent: Nesnenin üzerinde bulunduğu form'a ulaşmayı sağlar. Nesnenin isminden sonra verilen **parent** komutu o nesneyi değil nesnenin üzerinde bulunduğu formu temsil eder.

Refresh: Form üzerinde yapılan değişikliklerin anında gösterilmesini sağlar.

TabIndex: Form üzerinde kullanıcının ulaşabileceği bir nesnenin bir tabindexi vardır. Tabindex kullanıcının **tab** tusuyla kontroller arasında dolasırken bu kontrollerin sıralamasını belirler.

CausesValidation: Text kutusu kontrolünü kaybettiğinde yani kullanıcı text kutusundan başka bir kontrole fare veya klavye yardımı ile geçtiğinde validate olayının çalışıp çalışmayacağını belirler. Normalde bu özellik true'dir ve **validate** olayı çalışır. Bu olaya yazacağınız kodla kullanıcının girdiği bilgiyi anında kontrol edebilir ve gerekiyorsa uyarabilirsiniz.

Value: İkisi kullanıcı tarafından değiştirilebilen üç değer alabilir.

0 : isaretsiz

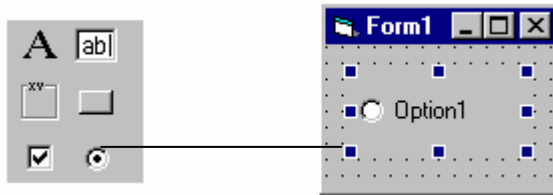
1 : isaretili

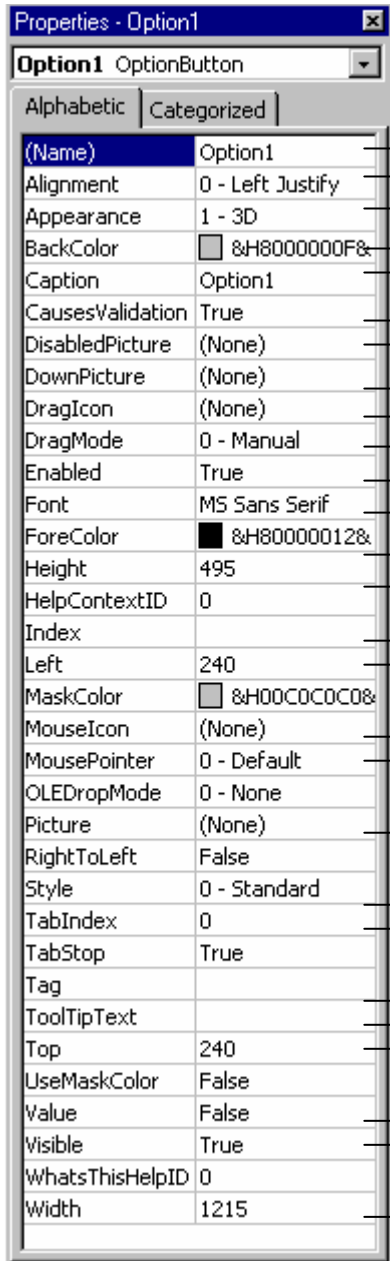
2 : belirsiz

İlk iki değer kullanıcı tarafından kontrol tiklanarak değiştirilebilir. Üçüncü değer ise belirsizlik değerini gösterir ve program tarafından bu durum aktif hale getirilebilir. Belirsizlik, value'nin değerinin 2 olmasıyla ilgili değildir. Sadece o kontrolün ifade ettiği değer belirsiz olduğunu gösterir.

5-Option Button(seçenek düğmesi):

Bir çok seçenekten sadece birisinin seçildiği bir seçenek kontrolüdür. Genellikle bir grup seçenekten birisinin seçilmesi ile işler.





OptionButton nesnesinin özellikleri

- Programın içerisindeki adını belirler.
- Üzerinde görünen açıklamayı hizalar.
- Formdaki görüntüsünü belirler
- Arka plan rengini belirler.
- Üzerindeki açıklamanın girildiği özelliktir.
- kontrolde validate çalışmasını kontrol eder.
- Kullanılamaz resmin ne olacağını belirler.
- Çıklendiğinde resmin ne olacağını belirler.
- Sürükleme de farenin alacağı şekli belirler.
- Sürüklemenin otomatik,manual olmasını belirler.
- Clicklenebilirliğini belirler.
- Yazı tipini belirler.
- Açıklama yazısının rengini belirler.
- Nesnenin yüksekliğini belirler.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesnede hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.
- Üzerinde görünecek yazıyı belirler.
- Kontrolde resim alıp almayacağını belirler.
- Kaçıncı TAB da kendine ulaşacağını belirler.
- TAB ile seçilmesi sonlandırılır.
- Herhangi bir mesaj saklamak için kullanılır.
- Nesnede mesaj verdirmek için kullanılır.
- Üstten hizasını belirler.
- Isaret kutusunun seçili olup olmadığını belirler.
- Formda görünüp görünmeyeceğini belirler.
- Nesnenin genişliğini belirler.

OPTION BUTTON NESNESİNİN BAZI ÖNEMLİ OLAYLAR

- 1. Private Sub Option1_Click()** : Olay seçenek düğmesine mouse un sol tusuna basıldığında çalışır.
- 2. Private Sub Option1_DblClick()** : Olay Option Button nesnesine mouse un sol tusla çift tıkladığı anda çalışır.
- 3. Private Sub Option1_DragDrop(source As Control ,X As Single, Y As Single):** Olay sürükleme olayı Option Button da bittiginde çalışır. Source sürükleme olayının başladığı nesneyi temsil eder. X ve Y mouse un koordinat değerlerini verir.
- 4. Private Sub Option1_DragOver(Source As Control, X As Single, Y As Single, State As Integer)** : Olay sürükleme esnasında mouse nesne üzerinden geçerken çalışır. State ise sürükleme olayının hangi asamada olduğu değerini verir. State özelliği 3 değer tasir.
- 5. Private Sub Option1_GotFocus()**: Olay, nesne aktif nesne olduğunda çalışır. Clicklendiğinde ve TAB tusuyla seçildiğinde aktif nesne olur.
- 6. Private Sub Option1_KeyDown(KeyCode As Integer, Shift As Integer):** Olay klavyenin tusuna basıldığı an çalışır. KeyCode klavyeden basılan tusun değerini tasir. Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir. Bunlar: 1-Shift, 2-Ctrl ve 4- Alt değerleridir. Beraber basıldıklarında gelen toplamlarından oluşur. Örneğin Ctrl+Alt=6 dir.
- 7. Private Sub Option1_KeyPress(KeyAscii As Integer):** Olay KeyDown olayından sonra çalışır. KeyAscii klavyeden basılan tusların ASCII karşılığını verir.
- 8. Private Sub Option1_KeyUp(KeyCode As Integer, Shift As Integer):** Olay KeyPress olayından sonra ,Klavyeden basılan tus bırakıldıktan sonra çalışır.

9. Private Sub Option1_LostFocus(): Olay option button nesnesi terk edildiği anda çalışır. Terk etme işlemi herhangi bir nesneye Clicklendiğine veya TAB ile atlama yapıldığında olur.

10. Private Sub Option1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single): Option Button nesnesi üzerine mouse un herhangi bir tusu ile clicklendiğinde çalışır. Button basılan tusun degerini verir. Shift ise mouse ile birlikte klavyeden basılan tusun degerini verir. X ve Y ise mouse un bulunduğu koordinatlarını verir.

11. Private Sub Option1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): Option Button nesnesi üzerinde mouse hareket ettirildiğinde çalışır.

12. Private Sub Option1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single): Mouse un tuları bırakıldığı anda çalışır.

OPTION BUTTON NESNESİNİN METHOTLARI

Visible: True ise nesne görülür, false yapılırsa görülmez belli şartlar gerçekleştiğinde gözükmesini istediğiniz kontrollerin visible özelliğini kullanabilirsiniz.

Zorder: Bu metod kullanıldığı nesneyi en öne veya en arkaya getirmek için kullanılır. Burada metodun 0 olması kontrolün diğerlerinin önüne, 1 olması arkasına getirilmesini sağlar.

Enabled: Clicklenebilirliğini belirtir true ise click olayında çalışır.

parent: Nesnenin üzerinde bulunduğu form'a ulaşmayı sağlar. nesnenin isminden sonra verilen **parent** komutu o nesneyi değil nesnenin üzerinde bulunduğu formu temsil eder.

Refresh: Form üzerinde yapılan değişikliklerin anında gösterilmesini sağlar.

TabIndex: Form üzerinde kullanıcının ulaşabileceği bir nesnenin bir tabindexi vardır. Tabindex kullanıcının **tab** tusuyla kontroller arasında dolasırken bu kontrollerin sıralamasını belirler.

Tag:Bu özelliğin visual basic için hiç bir anlamı yoktur.Kullanici bu parametreyi bir degisken gibi kullanabilir.Bu özellik bir string degisken olarak kullanılmalıdır.

HelpcontextID:Kontrole ilgili yardım dosyasındaki konu numarası bu özellikle belirlenir. Programınıza ait yardım dosyası varsa **project_properties** menüleri ile açılan asagidaki pencerenin **Help File name** kutusuna bu yardım dosyasının ismi verilir.

Help dosyası belirlendikten sonra help dosyasındaki hangi konunun hangi kontrole karşılık geldiği **HelpContextID** özelliği ile belirlenir.kullanici bu kontroldeyken F1 tusuna basildiginda belirlediginiz yardım dosyası açilir ve bu özelliğe numarasini verdiginiz konu gösterilir.

hWnd: Windows altında çalışan kontrollerin **handle** diye adlandırılan tanitici bir numarası vardır. Windows bu numarayı kullanarak kontrolleri tanir. Visual basic'te bu numara nesnenin **hWnd** özelliği ile programın çalışması esnasında öğrenilebilir. **hWnd** özelliği genellikle Windows API çağrılarında gereklidir.

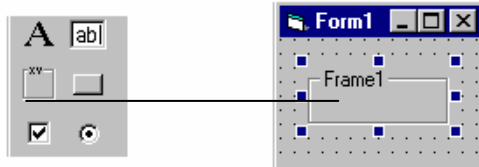
MaxLength:Text kutusuna girebilecek maximum karakter sayısını belirler.0 verilirse bu sınır kaldırılır.Verilen sınır sayısı kadar karakterden fazlasını kabul etmez.

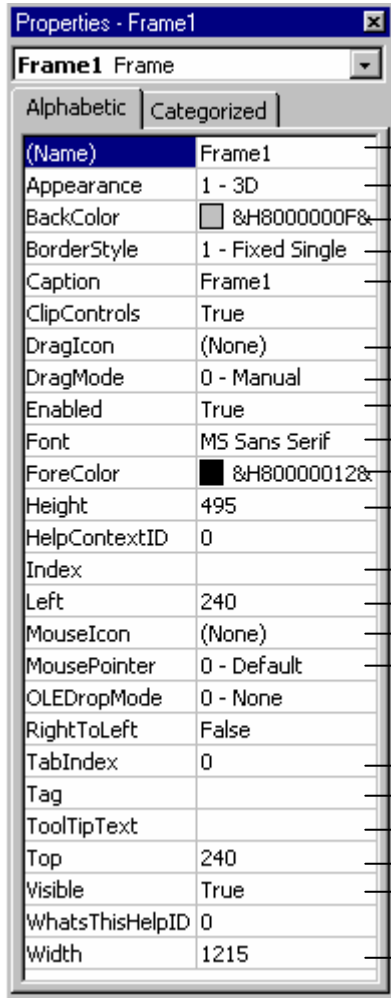
Kullanıcının belli sayıdan daha fazla karakter girmesini önlemek için bu özellik kullanılır.Örneğin kullanıcının 3 harften daha fazla giriş yapmasını önlemek için bu özelliğe 3 değeri verilebilir.

Maxlength özelliği sadece kullanıcının girebileceği karakter sayısını sinirlar.Program kodu ile yapılan atamalarda bu sınır asılabilir.

6-Frame control(çerçeve):

Bu kontrol tek basına değil ,diğer kontrolleri gruplamak için kullanılır. Bir çerçeve içine konan kontroller çerçeveye bağlıdır ve konumları bu çerçeve dışına tasamaz.Özellikle birkaç kontrolü birden görünür veya görünmez yapmak için hepsinin **visiple** özelliğini tek tek değiştirmek yerine çerçevenin **visible** özelliği değiştirilerek çerçeve içindeki tüm kontroller aynı anda görünmez yapılabilir.





Frame nesnesini özellikleri

- Program içerisindeki adını belirler.
- Formdaki görüntüsünü belirler.
- Arka plan rengini belirler.
- Kenar çizgilerini belirler.
- Üzerinde görünecek yazıyı belirler.
- Sürüklemeye fare ile alacağı şekli belirler.
- Sürüklemenin otomatik, manual olmasını belirler.
- Clicklenebilirliğini belirler.
- Yazı tipini belirler.
- Açıklama yazısının rengini belirler.
- Nesnenin yüksekliğini belirler.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesnede hangi şekli alacağını belirler.
- Kaçınca TAB da kendine ulaşacağını belirler.
- Herhangi bir mesaj saklamak içindir.
- Nesne de mesaj verdirmek için kullanılır.
- Üstten hizasını belirler.
- Formda görünüp görünmeyeceğini belirler.
- Nesnenin genişliğini belirler.

FRAME NESNESİNİN ÖNEMLİ OLAYLARI

1. Private Sub Frame_Click(): Olay metin kutusuna mouse'un sol tusuna basıldığında çalışır.

2. Private Sub Frame_DbClick(): Olay metin kutusuna sol tusla çift tıkladığında çalışır.

3. Private Sub Frame _DragDrop(Source As Control, X As Single, Y As Single):Olay sürüklemeye olayı metin kutusunda bittiginde çalışır.Source sürüklemeye olayının başladığı nesneyi temsil eder.X ve Y mouse'un koordinat değerlerini verir.

4. Private Sub Frame _DragOver(Source As Control, X As Single, Y As Single,State As Integer):Olay sürüklemeye esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürüklemeye olayının hangi asamada olduğu değerini verir.State özelliği üç değer tasir.

5. Private Sub Frame _MouseDown(Button As Integer,Shift As Integer X As Single, Y As Single):Metin kutusu üzerine Mouse'un herhangi bir tusu ile clicklendiginde çalışır.Button basılan tusun değerini verir.Shift ise Mouse ile birlikte klavyeden basılan tusun değerini verir. X ve Y ise mouse'un bulunduğu koordinatlarını verir.

6. Private sub Frame _MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single):Metin kutusu üzerindeyken çalışır.

7. Private Sub Frame _MouseUp(Button As Integer, Shift As Integer,X As Single, Y As Single):Mouse tuları bırakıldığı anda çalışır.

FRAME NESNESİNİN METODLARI

Aligment:Yazının sola saga veya ortaya yazilmasini saglar.

Appearance:Bu özellik kontrolün üç boyutlu görünümünü ayarlar.

Autosize: Nesnenin boyutları içeriğinin boyutlarına göre yeniden ayarlanacaktır.

Backcolor:Nesnenin zemin rengini degistirir. Bu özelliği oluşturan değer 7 digitlik bir hexadesimal sayıdır.(0 ile&HFFFFFF arasında) en düşük iki digit kırmızı,sonraki iki digit yeşil,ve diğer iki digit mavi rengin yoğunluğunu gösterir.Bu ikili digitlerin düşük degerde olmaları rengin koyuluğunu sağlar.en yüksek seviyeli digitin rengi 0 değilse windows'un system renklerinin kullanılacağını gösterir.Properties penceresinde olmayan renkler,bu digitler ayarlanarak elde edilebilir. Kullanilabilecek renk sayısı 16.777.216 renktir. Örneğin açık tonlu yeşili elde etmek için &H100FF00 değeri kullanılır.

Renkleri ayarlamak için **QBColor()** fonksiyonu da kullanılabilir. **QBColor()** fonksiyonuyla 16 temel renkten biri alınabilir.Örneğin 2 nolu renk için aşağıdaki satır kullanılabilir.

Text1.BackColor=QBColor(2)

Enabled: Clicklenebilirliğini belirtir true ise click olayında çalışır.

Font:VB'nin 5.0versiyonu ile bütün font özelliklerini bir arada bulunduran ve tek seferde hepsini birden belirtmeye yarayan font nesnesi tanımlanmıştır.Bu nesnenin alt özellikleriyle tek tek ayarlama yapılabileceği gibi tek seferde de başka bir font nesnesinin bütün özellikleri atanabilir.

index: Aynı isimli birden çok nesne oluşturulmuşsa VB bu nesneleri bir dizi olarak görür. Index parametresi bu nesnenin dizideki kaçınıcı eleman olduğunu belirler,bu nesneye bu dizi indexiyle ulaşılır.

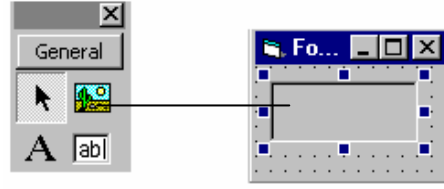
hWnd: Windows altında çalışan kontrollerin **handle** diye adlandırılan tanıtıcı bir numarası vardır. Windows bu numarayı kullanarak kontrolleri tanır. Visual basic'te bu numara nesnenin **hWnd** özelliği ile programın çalışması esnasında öğrenilebilir. **hWnd** özelliği genellikle Windows API çağrılarında gereklidir.

HelpcontextID:Kontrolle ilgili yardım dosyasındaki konu numarası bu özellik ile belirlenir. Programınıza ait yardım dosyası varsa **projest_properties** menüleri ile açılan aşağıdaki pencerenin **Help File name** kutusuna bu yardım dosyasının ismi verilir.

Help dosyası belirlendikten sonra help dosyasındaki hangi konunun hangi kontrole karşılık geldiği **HelpContextID** özelliği ile belirlenir.Kullanıcı bu kontroldeyken F1 tusuna basıldığında belirlediğiniz yardım dosyası açılır ve bu özelliğe numarasını verdığınız konu gösterilir.

HideSelection: True ise text kutusunda seçilen metnin,kontrol başka bir nesneye geçtiğinde seçilen kısmın gizlenmesini sağlar.

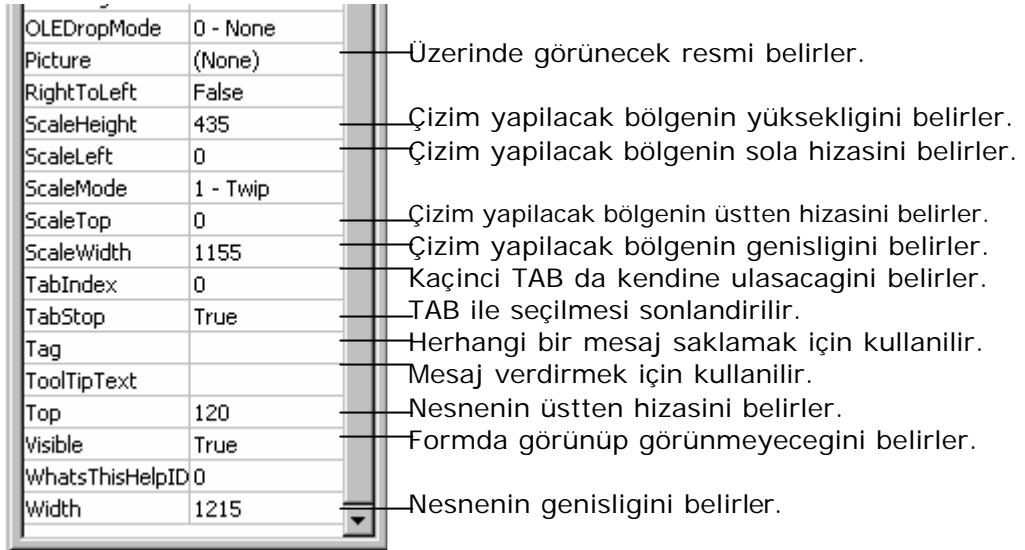
7-PictureBox(resim kutusu): Bu kontrol elemanı Bitmap,ikon,Metafile, Jpeg gibi resimleri görüntülemek için kullanılır.



Properties - Picture1	
Picture1 PictureBox	
Alphabetic	Categorized
(Name)	Picture1
Align	0 - None
Appearance	1 - 3D
AutoRedraw	False
AutoSize	False
BackColor	■ &H8000000F
BorderStyle	1 - Fixed Single
CausesValidation	True
ClipControls	True
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
DrawMode	13 - Copy Pen
DrawStyle	0 - Solid
DrawWidth	1
Enabled	True
FillColor	■ &H00000000
FillStyle	1 - Transparent
Font	MS Sans Serif
FontTransparent	True
ForeColor	■ &H80000012
HasDC	True
Height	495
HelpContextID	0
Index	
Left	120
LinkItem	
LinkMode	0 - None
LinkTimeout	50
LinkTopic	
MouseIcon	(None)
MousePointer	0 - Default
Negotiate	False
OLEDragMode	0 - Manual

PictureBox nesnesinin özellikleri

- Program içerisindeki adını belirler.
- Sürekli formun bir kenarında olmasını sağlar.
- Formdaki görüntüsünü belirler.
- Kendini otomatik olarak yenilemesini sağlar.
- İçerisinin boyutlarına göre yeniden ayarlanır.
- Arka plan rengini belirler.
- Üzerinde bulunduğu konuma uymasını sağlar.
- Kontrolde validate çalışmasını kontrol eder.
- Herhangi bir alana bağlanılmasını sağlar.
- Veri biçimini belirler.
- Veritabanında bağlanılacak tabloyu belirler.
- Sürüklemede farelin alacağı şekli belirler.
- Sürüklemenin otomatik,manual olmasını sağlar.
- Clicklenebilirliğini belirler.
- Kutuların iç boyama rengini ve desenini belirler.
- Nesnenin içindeki çizgilerin şeklini belirler.
- Yazı tipini belirler.
- Yazı altında resim yazı varsa bunu gösterir.
- Açıklama yazısının rengini belirler.
- Nesnenin yüksekliğini belirler.
- Yardım dosyasındaki konu numarasını belirler.
- Indexli kullanımda indexini belirler.
- Sola hizasını belirler.
- Nesne üzerinde hangi resmi alacağını belirler.
- Nesne üzerinde hangi şekli alacağını belirler.



PICTUREBOX NESNESİNİ BAZI ÖNEMLİ OLAYLARI

- 1. Private Sub Picture1_Click()**: Olay Mouse'un sol tusu ile forma tıklendiğinde çalışır.
- 2. Private Sub Picture1_DblClick()**: Olay nesneye Mouse'un sol tusa çift tıkladığında çalışır.
- 3. Private Sub Picture1_DragDrop(Source As Control, X As Single, Y As Single)**: Olay sürükleme olayı metin kutusunda bittiğinde çalışır. Source sürükleme olayının başladığı nesneyi temsil eder. X ve Y mouse'un koordinat değerlerini verir.
- 4. Private Sub Picture1 _DragOver(Source As Control, X As Single, Y As Single, State As Integer)**: Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır. State ise sürükleme olayının hangi aşamada olduğu değerini verir. State özelliği üç değer tasir.
- 5. Private Sub Picture1 _Change()**: Olay picture özelliğinin içeriği değiştiğinde çalışır.
- 6. Private Sub Picture1_GotFocus()**: Olay, nesne aktif nesne olduğunda çalışır. Clicklendiğinde ve TAB tusuyla seçildiğinde aktif nesne olur

7. Private Sub Picture 1_KeyDown(KeyCode As Integer, Shift As Integer): Olay klavyenin tusuna basildiği an çalışır. KeyCode klavyeden basılan tusun değerini tasir. Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir. Bunlar: 1-Shift, 2-Ctrl ve 4- Alt değerleridir. Beraber basıldıklarında gelen toplamlarından oluşur. Örneğin Ctrl+Alt=6 dir.

8. Private Sub Picture 1_KeyPress(KeyAscii As Integer): Olay KeyDown olayından sonra çalışır. KeyAscii klavyeden basılan tusların ASCII karşılığını verir.

9. Private Sub Picture 1_KeyUp(KeyCode As Integer, Shift As Integer): Olay KeyPress olayından sonra ,Klavyeden basılan tus bırakıldıktan sonra çalışır.

10.Private Sub Picture 1_LostFocus(): Olay PictureBox nesnesi terk edildiği anda çalışır. Terk etme işlemi herhangi bir nesneye Clicklendiğine veya TAB ile atlama yapıldığında olur.

11. Private Sub Picture 1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single): PictureBox nesnesi üzerine mouse'un herhangi bir tusu ile clicklendiğinde çalışır. Button basılan tusun değerini verir. Shift ise mouse ile birlikte klavyeden basılan tusun değerini verir. X ve Y ise mouse un bulunduğu koordinatlarını verir.

12. Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single): PictureBox nesnesi üzerinde mouse hareket ettirildiğinde çalışır.

13. Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single): Mouse un tuları bırakıldığı anda çalışır.

14. Private sub picture1_validate(cancel As integer): Olay resim metin kutusu terk edildiği anda çalışır.

15. Private sub picture1_resize():Formun genişliğinin veya yüksekliğinin degismesi ayrıca formun minimize edilmesi bu olayı meydana getirir.

PICTUREBOX NESNESİNİN METHODLARI

Aling:Picture kontrolünün sürekli olarak formun bir kenarında olmasını sağlar.

0-none yerleştirdiği koordinatlarda sabit kalır.

1-aling top sürekli üstte

2-aling bottom sürekli altta

3-aling left sürekli solda

4-Aling Right sürekli sağda

Autoredraw:Nesnenin kendini otomatik olarak yenilemesi ya da yenilenmesi sağlanır.

CurrentX, CurrentY:Methods'lar kullanarak yapılan çizim ve yazımlarda aktif pixelin koordinatları belirlenir.

FillColor:Circle ve line methodu ile form üzerine çizilen çember ve kutuların iç boyama rengini ve desenini belirler.

Fillstyle:Bu özelliğe verilebilecek değerler ve etkiler aşağıdaki gibidir.

0- vbFSSoild	Tam dolu
1- vbFSTransparent	Üzerinde bulunduğu yerin rengi zemin rengi
2- vbHorizontalLine	Yatay çizgili
3- vbVerticalLine	Dikey çizgili
4- vbUpwardDiagonal	Sola Eğik
5- vbDownwardDiagonal	Sağa eğik
6- vbCross	Kareli
7- vbDiagonalCross	Çapraz

hDC: Bir handle numarasıdır. Ancak özel bir numaradır. Handle numarası Windows tarafında verilen herhangi bir numaradır ve o kontrol yok edilene kadar aynıdır. Bu numara Windows uygulamaları ile Device,Driver arasında bir bağlantı kurar. Bu numara API'lerde de kullanılır.

Image:sadece okunabilir bir özelliktir. Picture kontrolü içine methods'lar(print,line,circle gibi)kullanılarak yapılan yazım ve çizimleri temsil eder.

PaintPicture: Resim isleme için geliştirilmiş bir methoddur. Aslında bu method BitBlt Api'sinin yaptığı işlemleri yapar. **Kaynak Picture** parametresi ile isleme girecek resim belirlenir. **Kaynakx, kaynaky** parametreleri ile islenecek

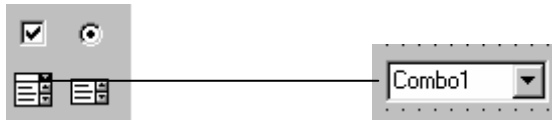
resmin sol üst köşe koordinatları ve **kaynak genişlik, kaynak yükseklik** parametreleriyle de boyutları belirlenir. **Hedefx, hedefy, hedef genişlik, hedefyükseklik** parametreleri ile de islenmiş resmin gösterileceği kontroldeki koordinatları ve boyutları belirlenir.

ScaleHeight, ScaleWidth: Çizimin yapılacağı bölgenin genişliğini belirler. Bu method nesne içinde picture özelliği ile yüklenen resmi etkilemez. Sadece nesnenin methodlarıyla yapılmış yazım ve çizimleri etkiler.

ScaleLeft, ScaleTop: Çizimi yapılacağı bölgenin koordinatlarını belirler. Default olarak 0,0 dir, yani yapılacak çizimler nesnenin sol üst köşesi orjin olmak üzere yapılır. Pozitif değerler orjini sol yukarı tasırken negatif değerler sağ aşağı tasir.

8-COMBOBOX:

Asağı doğru açılabilen bir liste kontrolüdür. Genellikle, değerleri daha önceden belli olan elemanların seçimi için kullanılırlar. Liste kutusuna benzer ancak listedeki elemanlardan sadece seçileni ekranda görüntüler.



compenent ve formda görünüşü

Properties - Combo1	
Combo1 ComboBox	
Alphabetic Categorized	
(Name)	Combo1
Appearance	1 - 3D
BackColor	<input type="checkbox"/> &H80000005&
CausesValidation	True
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	<input type="checkbox"/> &H80000008&
Height	315
HelpContextID	0
Index	
IntegralHeight	True
ItemData	(List)
Left	720
List	(List)
Locked	False
MouseIcon	(None)
MousePointer	0 - Default
OLEDragMode	0 - Manual
OLEDropMode	0 - None
RightToLeft	False
Sorted	False
Style	0 - Dropdown Combo
TabIndex	0
TabStop	True
Tag	
Text	Combo1
ToolTipText	
Top	360
Visible	True
WhatsThisHelpID	0
Width	1095

Combobox Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Formdaki görüntüsünü belirler
- Arka zemin rengini belirler
- Validate olayinin çalışmasını belirler
- Veritabanının bir alana bağlanmasını sağlar
- Veri biçimini belirler
- Veritabanında bağlanacak tabloyu belirler
- Sürüklemeye farenin alacağı şekli belirtir
- Sürükleme olayında mouse'nin şeklini belirler
- Clicklenebilirliğini belirler
- Yazı tipini belirler
- Yazı rengini belirler
- Nesnenin yüksekliğini belirler
- Yardım dosyasındaki konu numarası belirlenir
- Index'i belirler
- Elemanlar tam gösterilebilmek için boyutlanır
- Veri tabanı alanına seçenek eklemek
- Sola hizayı belirler
- Listeye seçenek eklemek
- Text'e girişi yasaklar
- Seçenekleri siraya dizer
- ComboBox'in çalışma şeklini belirler
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- İçine yazılacak metni belirler
- Gizli mesaj vermek için kullanılır
- Üstten hizalama
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

COMBOBOX'IN METOTLARI

COMBOBOX'ın metotları şunlardır:

AddItem:Formda çıkan listede görünmesini istediğimiz elemanları eklemek için kullanılır.

Örnek: Combo1.AddItem "İstanbul"

İstanbul seçeneğini eklemiş olduk.

Appearance:Bu özellik kontrolün üç boyutlu görünümünü ayarlar.

BackColor:Nesnenin zemin rengini değiştirir. Bu özelliği oluşturan değer 7 digitlik bir hexadesimal sayıdır.(0 ile &HFFFFFF arasında) en düşük iki digit kırmızı,sonraki iki digit yeşil,ve diğer iki digit mavi rengin yoğunluğunu gösterir.Bu ikili digitlerin düşük değerlerde olmaları rengin koyuluğunu sağlar.En yüksek seviyeli digitin rengi 0 değilse windows'un system renklerinin kullanılacağını gösterir.Properties penceresinde olmayan renkler,bu digitler ayarlanarak elde edilebilir.Kullanılabilecek renk sayısı 16.777.216 renktir.

Causesvalidation:ComboBox kontrolünü kaybettiğinde yani kullanıcı ComboBox'a başka bir kontrole fare veya klavye yardımı ile geçtiğinde validate olayının çalışıp çalışmayacağını belirler.Normalde bu özellik true'dir ve validate olayı çalışır. Bu olaya yazacağınız kodla kullanıcının girdiği bilgiyi anında kontrol edebilir ve gerekiyorsa uyarabilirsiniz.

Clear:Liste kutusunu temizlemek için kullanılır.

Örnek: List1.Clear

Datafield:Veritabanının herhangi bir alana bağlanmasını sağlar.

Dataformat:Veritabanındaki bilgiyi formatını ayarlar.

Datasource:Veri tabanındaki tabloyu belirler.

Dragicon:Sürükleme olayında fare kullanılacağı şekli belirler.

Dragmode:Sürükleme olayının otomatik ve ya manual olacağını belirler.

Enabled:Clicklenebilirliğini belirtir,true ise click olayında çalışır.

Font:VB'nin 5.0 versiyonu ile bütün font özelliklerini bir arada bulunduran ve tek seferde hepsini birden belirtmeye yarayan font nesnesi tanımlanmıştır.Bu nesnenin alt özellikleriyle tek tek ayarlama yapılabileceği gibi tek seferde de başka bir font nesnesinin bütün özellikleri atanabilir.

Fontbold:True ise nesne için kullanılan yazıyı koyu yapar.

FontItalic:True ise nesne için kullanılan yazı eğik yapar.

Fontname:Nesne için kullanılan yazının fontunu belirler.Bu font çalışma esnasında sistemde bulunan bir fontun numarası veya ismi verilerek değiştirilebilir.

FontSize:Nesne için kullanılan yazının puntosunu belirler.Maksimum 2048 olabilir.

Fontstrikethru: True ise nesne için kullanılan yazının ortasını çizer.

Fontunderline: True ise nesne için kullanılan yazının altını çizer.

Fontcolor:Yazının rengini belirler.

Height,width:Nesnenin boyunu ve enini belirler. Bu özellik değiştirilecek kontrolün boyutları ayarlanabilir.

HelpcontextID:Kontrolle ilgili yardım dosyasındaki konu numarası bu özellik ile belirlenir.Programınıza ait yardım dosyası varsa `projet_properties` menüleri ile açılan aşağıdaki pencerenin Help File name kutusuna bu yardım dosyasının ismi verilir.Help dosyası belirlendikten sonra help dosyasındaki hangi konunun hangi kontrole karşılık geldiği `HelpContextID` özelliği ile belirlenir.Kullanıcı bu kontroldeyken F1 tuşuna basıldığında belirlediğiniz yardım dosyası açılır ve bu özelliğe numarasını verdiğiniz konu gösterilir.

hWnd:Windows altında çalışan kontrollerin handle diye adlandırılan tanıtıcı bir numarası vardır. Windows bu numarayı kullanarak kontrolleri tanır. Visual basic'te bu numara nesnenin `hWnd` özelliği ile programın çalışması esnasında öğrenilebilir. `hWnd` özelliği genellikle Windows API çağrılarında gereklidir.

Index:Aynı isimli birden çok nesne oluşturulmuşsa VB bu nesneleri bir dizi olarak görür. Index parametresi bu nesnenin dizideki kaçıncı eleman olduğunu

belirler, bu nesneye bu dizi indexiyle ulaşılır. Index özelliğine bir sayı verilmişse o kontrole isminden sonra bu sayı verilerek ulaşılabilir.

IntegralHeight: Bu özelliğin değeri True ise ComboBox içindeki elemanlar tam gösterilebilmek için kendisini yeniden boyutlandırır. Normlde bu özellik True'dir ve CombBox'ın boyutları belli aralıklarla değiştirilebilir.

ItemData(Index): Index numaralı elemana comboda görülmeyecek bir numara verilmesini sağlar. Sıralanmış listelerde herhangi bir elemanın listeye konulduğu sıra belli olamayacağı için bu özellik kullanılarak listaya konulan elemana sıra numarası verilebilir.

Left, Top: Nesnenin sol ve üst noktalarının kordinantlarını belirler. Bu kordinantlar ekran kordinantları olmayıp içinde bulunduğu nesneye bağlı kordinantlardır. Bir nesne form, PictureBox veya frame içinde bulunabilir. İçinde bulunduğu nesnenin sol üst kordinantları 0,0 olmak üzere left ve top bu noktaya olan uzaklıktır. VB'de ölçü birimi olarak twip kullanılır. Ancak istenirse formun scalemode özelliği değiştirilerek ölçü birimi olarak piksel, veya metrik birimler verilebilir.

ListIndex: ComboBox'taki aktif elemanı belirtir.

Örnek: Combo1.ListIndex=- 1

-1 olduğunda hiçbirisinin aktif olmadığını belirtir.

ListCount: ComboBox'ta kaç tane seçenek olacağını belirtir.

Örnek: Combo1.ListCount=4

4 tane seçenek olduğunu belirttik.

List(Index): Liste içindeki index numaralı elemanın değerini öğrenmek veya değiştirmek için kullanılır.

Locked: Bu özellik true yapıldıktan sonra kullanıcı ComboBox üzerinde hiçbir değişiklik yapamaz.

Mouseicon, pointer: Mouse göstergesinin nesne üzerine geldiğinde alacağı şekli belirler. 0 ile 15 arası bir sayıdır. 0 normal hali 11 bekleme hali vb.

Bu özelliğe 99 değerini vererek kendi tasarladığınız (ICO veya CUR dosyasından) bir kursörü seçebilirsiniz. Bu işlem için kendi tasarladığınız kursörün bulunduğu dosyanın adını MouseIcon özelliğine girmeniz gerekir.

Move: Formun veya herhangi bir kontrolün konumunu ve boyutlarını tek seferde değiştirir.

Kontrolün Left,width,height propertisleriyle ayrı seferde yapacağınız işi bu methodla tek seferde yapabilirsiniz.Propertislerle yapmanız durumunda kontrol üç defa şekil değiştirecek ve dolayısıyla daha ağır ve çirkin bir görüntü oluşacaktır.

NewIndex:Listeye en son eklenen elemanın listindexini verir.Sorted özelliği False ise bu değer ListCount -1'dir. True ise elemanın sıralanmış listedeki yerinin indexidir.

Parent:Nesnenin üzerinde bulunduğu form'a ulaşmayı sağlar.Nesnenin isminden sonra verilen parent komutu o nesneyi değil nesnenin üzerinde bulunduğu formu temsil eder.

Refresh:Form üzerinde yapılan değişikliklerin anında gösterilmesini sağlar.

RemoveItem:Listeden istenilen elemanı çıkarmak için kullanılır. Listeden hangisi çıkarılmak isteniyorsa onun index numarası yazılır.

Örnek:Combo1.RemoveItem 3

Listeden 4. elemanı çıkarmış olduk. Çünkü index numaraları 0'dan başlar.

SelLength,selstart:Windows altında bir çok işlem için bloklama (seçme)yapmak gerektiğini biliyorsunuz.Kullanıcı ComboBox'larda seçme işlemi yapabilir.Kullanıcının seçtiği kısmı öğrenmek için bu ikili özellik kullanılabilir.Ayrıca bu iki özelliği kullanarak siz de program kodu ile seçme işlemini yaptırabilirsiniz.

SelText:Seçilmiş metin bu özellik ile öğrenilip değiştirilebilir.Seltext özelliğine yapılan atamalarda,eger seçilmiş bir kısım varsa o kısım silinir,yoksa kursörün bulunduğu noktaya sikistirir.

SetFocus:Nesnenin etkin konum önceliğine sahip olmasını sağlar. Örnek:Combo1.SetFocus

Sorted:Listenin alfabetik sıralı olmasını sağlar. Comboya eklenen elemanlara sona değil alfabetik saraya yerleştirir. Ancak bu sıralama işlemi syılar üzerinde doğru etki göstermez.

Style:Bu özelliğe 1 değeri verildiğinde kullanıcı elemanları işaretleyebilir.

TabIndex:Form üzerinde kullanıcının ulaşabileceği bir nesnenin bir tabindexi vardır. Tabindex kullanıcının tab tusuyla kontroller arasında dolasırken bu kontrollerin sıralamasını belirler.

TabStop:True ise kullanıcı bu nesneye Tab ile ulaşabilir.False ise Tab tusuyla bu kontrol üzerine gelinmez,mouse ile,yazılım yoluyla veya varsa kısayol tusuyla gelinebilir.

Tag:Bu özelliğn visual basic için hiçbir anlamı yoktur.Kullanıcı bu parametreyi bir değişken gibi kullanabilir.Bu özellik bir string değişken olarak kullanılmalıdır.

Text:Bu özellik kullanılarak kullanıcıyı girdiği metin üzerinde işlem yapılır.

Örnek: Combo1.Text=" "

ComboBox'ın boş olduğu belirtilir.

ToolTiptext:Windows altında çalışan bir çok modern programda program öğelerinin birinin üzerinde fare ile kısa bir süre durduğunuzda açılan bir balonla onun ne işe yaradığını yazan mesajlar görmüşsünüzdür. VB'de de bu iş o kontrolün tooltiptext özelliğine atanacak metinle yapılır.Bu özelliğe verdiğiniz metin, kullanıcının o kontrol üzerinde kısa bir süre durmasıyla, küçük bir kutu içerisinde gösterilecektir.

TabIndex:Form üzerinde kullanıcının ulaşabileceği bir nesnenin bir tabindexi vardır. Tabindex kullanıcının tab tusuyla kontroller arasında dolasırken bu kontrollerin sıralamasını belirler.

Visible:True ise nesne görülür,false yapılırsa görülmez belli şartlar gerçekleştiğinde gözükmesini istediğiniz kontrollerin visible özelliğini kullanabilirsiniz.

Zorder:Bu metod kullanıldığı nesneyi en öne veya en arkaya getirmek için kullanılır.Burada metodun 0 olması kontrolün diğerlerinin önüne,1 olması arkasına getirilmesini sağlar.

COMBOBOX'IN OLAYLARI

COMBOBOX'ın olayları şunlardır:

Combo1_Change():Nesne içeriği üzerinde değişiklik yapıldığı zaman çalışır.

Combo1_Click():ComboBox listesinden herhangi bir eleman seçildiği zaman çalışır.

Combo1_DblClick():ComboBox'a mouse'un sol tusuyla çift tiklandığında çalışır.

Combo1_DragDrop(Source As Control, X As Single, Y As Single):Olay sürükleme olayı metin kutusunda bittiginde çalışır.Soruce sürükleme olayının başladığı nesneyi temsil eder.X ve Y mouse'un koordinat değerlerini verir.

Combo1_DragOver(Source As Control, X As Single, Y As Single,State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürükleme olayının hangi asamada olduğu değerini verir.State özelliği üç değer tasir.

Combo1_GotFocus():Form üzerinde ComboBox aktif nesneyken çalışır.

Combo1_KeyDown(Key Code As Integer, Shift As Integer):Olay veri girişi esnasında klavyenin tusuna basıldığı an çalışır.KeyCode klavyeden basılan tusun değerini tasir.Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir.Bunlar: 1-Shift, 2-Ctrl ve 4-Alt değerleridir.Beraber basıldıklarında gelen değer toplamlarından oluşur.Örneğin Ctrl+Alt=6'dır.

Combo1_KeyPress(KeyAsciiAs Integer):Olay veri girişi esnasında KeyDown olayından sonra çalışır.KeyAscii klavyeden basılan tusların ASCII karşılığını verir.

Combo1_KeyUp(KeyCode As Integer,Shift As Integer):Olay KeyPress olayından sonra, klavyede basılan tus bırakıldıktan sonra çalışır.

Combo1_LostFocus():ComboBox aktif olmadığında çalışır.

Combo1_Scroll():Farklı liste elemanı seçildiğinde yani liste elemanları değiştiğinde çalışır.

Combo1_Validate(Cancel As Integer):LostFocus olayındaki kontrol olayını yapmaya yarar. Eger nesnenin CausesValidadion özelliği True ise bu olaya yazdığınız kod çalışacak ve nesnenin içeriğini kontrol edebilmenizi sağlayacaktır.

ÖRNEK:İçeriği kullanıcı tarafından değiştirilebilen ComboBox'larda kullanıcının kutuya yazdığı değer listeye eklenmez. Listeye eklenmek isteniyorsa kod yazmak gereklidir.

Örnek olarak bir personelin adini,meslegini ve adresini öğrenmek isteyelim. Mümkün olabilecek bazı meslekleri ise bir ComboBox içinde listeleyelim,eger personelin meslegi bunlardan biri degilse meslegini ComboBox'a eklemesine izin verelim. Personelin adi ve adresi için birer Text kutusu,meslegi için ise Style'i 0 olan bir ComboBox olusturalim.

```
Private Sub Form_Load ()
    Combo1.AddItem "Vasifsiz Isçi"
    Combo1.AddItem "Mühendis"
    Combo1.AddItem "Muhasebeci"
    Combo1.AddItem "Egitimci"
    Combo1.AddItem "Programci"
    Combo1.AddItem "Teknisyen"
    Combo1.ListIndex = 0 ' ilk elemani kutuda göster
End Sub
Private Sub Combo1_LostFocus ()
    Dim i
    For i= 0 To Combo1.ListCount -1
        If Combo1.Text = Combo1.List (i) Then
            'Kutuda yazili olan listedeki elemanlardan biri ise
            'Degisim yok fonksiyondan çık
            Exit Sub
        End If
    Next
    'ComboBox'a giris yapilmis
    Combo1.AddItem Combo1.Text 'Girileni listeye ekle
End Sub

Private Sub Combo1.KeyPress (keyascii As Integer)
    If keyascii = 27 Then
        'ESC' ye basildi ise
        Combo1.Text = Combo1.List (0)
    End If
End Sub
```

Kullanıcının ComboBox'a herhangi bir deger girip girmediğini ComboBox'un

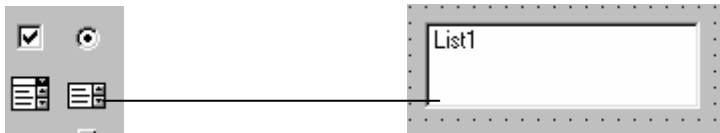
LostFocus olayında yazdığımız kodla anlıyoruz. LostFocus olayını kullanmamızın sebebi kontrol ComboBox'tan gittiğine göre yazılacak olan metninde bitmiş olacağındandır. Burada ComboBox kutusunda yazılı olanla (Combo1.Text) listedeki bütün elemanları karşılaştırıyoruz, kutudaki yazı listede var ise bu yeni eklenmiş bir şey değildir ve fonksiyondan çıkartıyoruz. Listedeki yok ise yeni girilmiş bir şeydir ve bunu listeye ekliyoruz. Ayrıca kullanıcının ComboBox'a yazdığı bir şeyi ESC ile iptal edebilmesi için de KeyPress olayındaki kod ile kutuyu içindeki yazı listedeki bir elemana esitlenmiştir.

ComboBox'lara kullanıcının yazdiklarını eklemek değişik yollarla da yapılabilir. Burada kullandığımız yöntem her zaman ise yaramayabilir. Örneğin ComboBox'ta bir şey yazdıktan sonra menüden bir seçeneğin seçilmesi ComboBox'ta LostFocus olayını gerçekleştirmeyecektir. Eğer seçilen menüde ComboBox'ın değerine ihtiyaç duyuyorsa doğru değeri alamayacaktır. Alternatif bir yöntem olarak yalnız KeyPress olayını aşağıdaki gibi kullanabiliriz. Bu durumda ise kullanıcının listeye eklemek istediği yeni değeri yazdıktan sonra Enter'e basması gerekir. Enter'e basmaması durumunda listeye eklenmeyecektir.

```
Private Sub Combo1.KeyPress (keyascii As Integer)
    Dim i
    If keyascii = 27 Then
        'ESC' ye basildi ise
        Combo1.Text = Combo1.List (0)
    End If
    If keyascii = 13 Then
        For i = 0 To Combo1.ListCount - 1
            If Combo1.Text = Combo1.List (i) Then
                'Kutuda yazili olan listedeki elemanlardan biri ise
                'Degisim yok fonksiyondan cik
                Exit Sub
            End If
        Next
        'ComboBox'a giris yapilmis
        Combo1.AddItem Combo1.Text 'Girileni listeye ekle
    End If
End Sub
```

9-LISTBOX:

Visual Basic'in sagladigi dizilerinizi gösterebileceginiz kontrollerdendir. Elemanlari listelemek,siralamak gibi özellikler sunan genel amaçli bir kontroldür.



compenent ve formda görünüsü

ListBox Nesnesini Özellikleri

Properties - List1

List1 ListBox

Alphabetic | Categorized

(Name)	List1
Appearance	1 - 3D
BackColor	<input type="checkbox"/> &H80000005&
CausesValidation	True
Columns	0
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	<input checked="" type="checkbox"/> &H80000008&
Height	450
HelpContextID	0
Index	
IntegralHeight	True
ItemData	(List)
Left	3960
List	(List)
MouseIcon	(None)
MousePointer	0 - Default
MultiSelect	0 - None
OLEDragMode	0 - Manual
OLEDropMode	0 - None
RightToLeft	False
Sorted	False
Style	0 - Standard
TabIndex	1
TabStop	True
Tag	
ToolTipText	
Top	1080
Visible	True
WhatsThisHelpID	0
Width	255

- Seçili nesnenin ismidir.
- Formdaki görüntüsünü belirler
- Validate olayının çalışmasını belirler
- Arka zemin rengini belirler
- Listenin kolon sayısını belirler
- Veritabanının bir alana bağlanmasını sağlar
- Veri biçimini belirler
- Veritabanında bağlanacak tabloyu belirler
- Sürükleme olayında mouse'nin şeklini belirler
- Sürükleme olayının nasıl olacağını belirler
- Clicklenebilirliğini belirler
- Yazı tipini belirler
- Yazı rengini belirler
- Nesnenin yüksekliğini belirler
- Yardım dosyasındaki konu numarası belirlenir
- Index'i belirler
- Elemanlar tam gösterilebilmek için boyutlanır
- Veri tabanı alanına seçenek eklemek
- Sola hizayı belirler
- Listeye seçenek eklemek
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki şekil
- Listeden birkaç seçeneği seçmeyi sağlar
- Seçenekleri siraya dizer
- ComboBox'ın çalışma stilini belirler
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- Gizli mesaj vermek için kullanılır
- Üstten hizalama
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

LISTBOX'IN METOTLARI

LISTBOX'in farklı metotları şunlardır:

Columns:Bu özellik ile liste kutusu birkaç kolon yapılabilir.

0 ise:Liste kutusu tek sütundur ve listenin ekranda görülen kısmının dolmasıyla listeye dikey scrollbar eklenir.

0 değilse:Listenin genişliği verilen sayıda sütuna bölünür ve bir sütunun dolmasıyla ikinci sütuna geçilir. Listenin görülen kısmındaki sütunların dolmasıyla listeye yatay scrollbar eklenir. Burada her kolonun genişliği listenin genişliğinin verilen kolon sayısına oranıdır. Örneğin Columns=5 ve Width=3000 ise her kolonun genişliği $3000/5=600$ olacaktır.

MultiSelect:Bu özellik bir kutu içinde birden fazla elemanı seçme imkanı verir. İki değişik modu vardır.

0:Birden fazla eleman seçimi yapılamaz.

1:Mouse ile tiklanan her eleman seçilir veya seçilmişse seçilmişliği kaldırılır.

2:Bu modda Shift veya Ctrl tuşu basılı tutularak birden fazla seçim yapılabilir.

Bu özellik sadece tasarım zamanında properties penceresinden değiştirilebilir.

Çalışma zamanında yapılacak atama ile listenin MultiSelect özelliği değiştirilemez. Bu özellik aktif hale getirilirse seçili olan elemanları öğrenmek için SelCount ve Selected özellikleri kullanılır.

Selcount,Selected(Index):MultiSelect özelliği 0 olmayan listelerde birden fazla elemanı seçilebileceği için SelCount özelliği ile seçili eleman sayısı,Selected(Index) özelliği ile de Index numaralı elemanın seçili olup olmadığı öğrenilebilir.

LISTBOX'IN OLAYLARI

LISTBOX'in farklı olayları şunlardır:

List1_MouseDown(Button As Integer,Shift As Integer X As Single, Y As Single):ListBox üzerine Mouse'un herhangi bir tuşu ile clicklendiğinde çalışır.Button basılan tuşun değerini verir.Shift ise mousela birlikte klavyeden basılan tuşun değerini verir. X ve Y ise mouse'un bulunduğu koordinatlarını verir.

List 1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single):Mouse etiket üzerindeyken çalışır.

List11_MouseUp(Button As Integer, Shift As Integer,X As Single, Y As Single):Mouse tuslari birakildigi anda çalisir.

List11_ItemCheck(Item As Integer):Listenin Style özelligine 1-CheckBox verilmissse listedeki elemanlarin birer CheckBox gibi olacaktır. Bu durumda listeden bir eleman isaretlendiginde ItemCheck olayi meydana gelir. Burdaki Item paremetresi isaretlenen elemanin listedeki Index numarasidir.

ÖRNEK:Genel bir örnek olarak personel bilgilerinin girilebilecegi bir program yazalım. Örneğimiz için asagidaki formu olusturun.

Örneğimizde:**Ekle** düğmesi (Command1) ile Text kutularına girilenleri ayrı ayrı listelerin sonuna ekleyeceğiz.

Sil düğmesi (Command2) ile Listedenden seçilen personeli sileceğiz.

Bul düğmesi (Command3) ile Text1'e ismi girilen elemanı listede bulup seçeceğiz.

Değistir düğmesi (command4) ile listede seçilen elemanı Text kutularına girilen yeni değerlerle değistireceğiz.

Araya Ekle düğmesi (Command5) ile Text kutularına girilen personeli, Listedende seçin personelin önüne ekleyeceğiz.

Listelerden birinde seçilen elemanın diğer listelerde de seçilmesi için ListBox'ların Click olayları aracılığı ile ListIndex özelliklerinin, seçilen listedeki ile aynı olmasını sağlayacağız. Ve programdan çıkarken listeleri kaydedip girerken yükleyeceğiz.

Microsoft Visual Basic 6.0

```
Private Sub Form_Load ()
    Combo1.AddItem "Mühendis"
    Combo1.AddItem "İşçi"
    Combo1.AddItem "Programci"
    Combo2.AddItem "Üretim"
    Combo2.AddItem "Pazarlama"
    Combo2.AddItem "Reklam"
    Combo2.AddItem "Satis"
    Dim x, y, z
    'Daha önce kaydedilmiş dosya varsa aç
    If Dir ("per.dat") <> "" Then 'Dosya varsa
        Open "pers.dat" For Input As #1
        While Not EOF (1) 'Dosya sonuna kadar
            Input #1, x, y, z
            List1.AddItem x
            List2.AddItem y
            List3.AddItem z
        Wend
        Close #1
    End If
End Sub

Private Sub Command1.Click () 'Ekle Düğmesi
    List1.AddItem Text1
    List2.AddItem Combo1.Text
    List3.AddItem Combo2.Text
    Label6 = List1.ListCount
End Sub

Private Sub Command2.Click () 'Sil Düğmesi
    If List1.ListIndex < 0 Then
        MsgBox ("Önce silinecek elemani seçiniz")
        Exit Sub
    End If
    Dim ind, c
    c = MsgBox (List1.List (ind) & "silinsinmi", vbYesNo +
VbExclamation + vbDefaultButton2, "Sil")
    Ind = List1.ListIndex
    If c = vbNo Then Exit Sub
    List1.RemoveItem ind
    List2.RemoveItem ind
    List3.RemoveItem ind
    Label6 = List1.ListCount
End Sub
```

```
Private Sub Command3.Click () 'Bul Düğmesi
    Dim i
    For i = 0 To List1.ListCount - 1
        If Ucase (List1.List (i)) = Ucase (Text1) Then
            'Bulundu ise seç
            List1.ListIndex = i
            Exit Sub
        End If
    Next
    MsgBox (Text1 & "bulunamadi")
End Sub

Private Sub Command4.Click () 'Degistir Düğmesi
    Dim ind
    If List1.ListIndex < 0 Then
        MsgBox ("Önce degistirilecek elemani seçiniz")
        Exit Sub
    End If
    Ind = List1.ListIndex
    List1.List (ind) = Text1
    List2.List (ind) = Combo1.Text
    List3.List (ind) = Combo2.Text
End Sub

Private Sub Command5.Click () 'Araya ekle Düğmesi
    Dim ind
    If List1.ListIndex < 0 Then
        MsgBox ("Önce elemanın nereye ekleneceğini seçiniz")
        Exit Sub
    End If
    Ind = List1.ListIndex
    List1.AddItem Text1, ind
    List2.AddItem Combo1.Text, ind
    List3.AddItem Combo1.Text, ind
    Label6 = List1.ListCount
End Sub

Private Sub List1_Click ()
    'Birinde seçilene diğerlerinde de seç
    List2.ListIndex = List1.ListIndex
    List3.ListIndex = List1.ListIndex
    List2.TopIndex = List1.TopIndex
    List3.TopIndex = List1.TopIndex
    Label8 = List1.ListIndex + 1
    Text1 = List1.Text
End Sub
```

```
Private Sub List2_Click ()
    List1.ListIndex = List2.ListIndex
    List3.ListIndex = List2.ListIndex
    List1.TopIndex = List2.TopIndex
    List3.TopIndex = List2.TopIndex
    Label8 = List2.ListIndex + 1
    Combo1.Text = List2.Text
End Sub

Private Sub List3_Click ()
    List2.ListIndex = List3.ListIndex
    List1.ListIndex = List3.ListIndex
    List2.TopIndex = List3.TopIndex
    List1.TopIndex = List3.TopIndex
    Label8 = List3.ListIndex + 1
    Combo2.Text = List3.Text
End Sub

Private Sub Form_Unload (Cancel As Integer)
    'Çıkışta listeleri pers.dat dosyasına kaydet
    Dim x, y, z, i
    Open "pers.dat" For Output As #1
    For i = 0 To List1.ListCount - 1
        x = List1.List (i)
        y = List2.List (i)
        z = List3.List (i)
        Write #1, x, y, z
    Next
    Close #1
End Sub
```

Personel Takip

Personelin

Adı Soyadı: Selahatin KOCABAŞ

Mesleği: Pazarlama

Çalıştığı Birim: Mühendis

Ekle Sil Bul

Değiştir Araya Ekle

Perihan AKKIŞI	Muhasebeci	Üretim
Selahatin KOCABAŞ	Mühendis	Pazarlama
Erkan ATAL	Mühendis	Pazarlama
İsmet KAYA	Mühendis	Pazarlama
Mehmet Ali ŞAVLI	Mühendis	Pazarlama
Gülhan ULUMAN	Mühendis	Pazarlama
Gülhan BAĞCI	Mühendis	Pazarlama

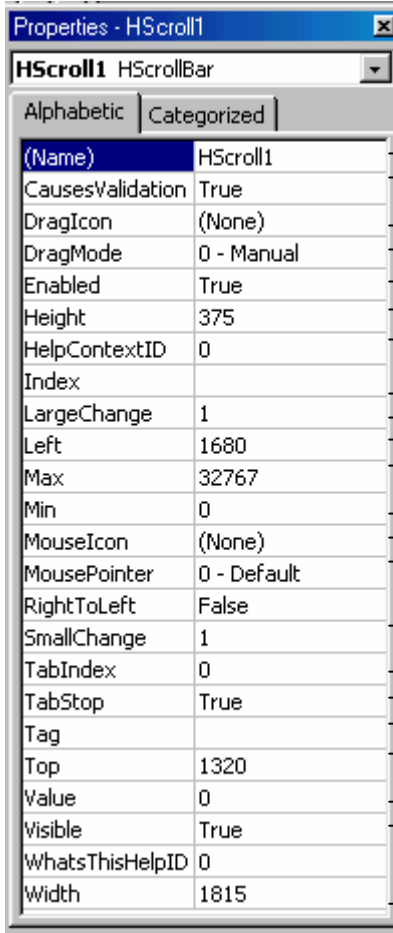
Personel Sayısı 9 Sıra No 2

10-H.SCROLLBAR:

Yatay olarak sağa sola okların çalışmasını sağlar.



component ve formda görünüşü



H.ScrollBar Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Validate olayının çalışmasını belirler
- Sürükleme olayında mouse'nin şeklini belirler
- Sürükleme olayının nasıl olacağını belirler
- Clicklenebilirliğini belirler
- Nesnenin yüksekliğini belirler
- Yardım dosyasındaki konu numarası belirlenir
- Index'i belirler
- Kaydırma çubuğuna tıklandığında ne kadarlık değişeceği
- Sola hizayı belirler
- Kaydırma çubuğunun alacağı max değer
- Kaydırma çubuğunun alacağı min değer
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki şekil
- Kaydırma çubuğunu ikitarafta ki oklarla ne kadarlık değişeceği
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- Üstten hizalama
- Kaydırma çubuğunun temsil ettiği değer
- Nesnenin görünüp görünmeyeceği

H.SCROLL'IN METOTLARI

H.SCROLL'in farklı metodları şunlardır:

LargeChange:Kaydırma çubuğu üzerine tıklanması durumunda H.Scroll'un ne kadarlık bir değişime tabi tutulacağını belirler.

Max,Min:Kaydırma çubuğunun alabileceği maksimum ve minimum değerdir.

SmallChange:Kaydırma çubuğunu iki kenarındaki oklarla H.Scroll'un ne kadarlık bir değişime tabi tutulacağını belirler.

Value:Kaydırma çubuğunun temsil ettiği değeri gösterir. Bu değer max ile min arasında bir sayıdır ve kaydırma çubuğu üzerinde değişik şekillerde değiştirilebilir.

H.SCROLL'IN OLAYLARI

H.SCROLL'in farklı olayları şunlardır:

H.Scroll1_Change():Nesne içeriği üzerinde değişiklik yapıldığı zaman çalışır.

H.Scroll1_DragDrop(Source As Control, X As Single, Y As Single):Olay sürükleme olayı metin kutusunda bittğinde çalışır.Soruce sürükleme olayının başladığı nesneyi temsil eder.X ve Y mouse'un koordinat değerlerini verir.

H.Scroll1_DragOver(Source As Control, X As Single, Y As Single,State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürükleme olayının hangi aşamada olduğu değerini verir.State özelliği üç değer tasir.

H.Scroll1_GotFocus():Form üzerinde nesne aktif nesneyken çalışır.

H.Scroll1_KeyDown(Key Code As Integer, Shift As Integer):Olay veri girişi esnasında klavyenin tusuna basıldığı an çalışır.KeyCode klavyeden basılan tusun değerini tasir.Shift Ctrl, Alt ve Shift tuslarına karşılık değer üretir.Bunlar:1-Shift, 2-Ctrl ve 4-Alt değerleridir.Beraber basıldıklarında gelen değer toplamlarından oluşur.Örneğin Ctrl+Alt=6'dir.

H.Scroll1_KeyPress(KeyAsciiAs Integer):Olay veri girişi esnasında KeyDown olayından sonra çalışır.KeyAscii klavyeden basılan tusların ASCii karşılığını verir.

H.Scroll1_KeyUp(KeyCode As Integer,Shift As Integer):Olay KeyPress olayından sonra, klavyede basılan tus bırakıldıktan sonra çalışır.

H.Scroll1_LostFocus():Nesne aktif olmadiginda yani terkedildiginde çalisir.

H.Scroll1_Scroll():Farkli liste elemani seçildiginde çalisir.

11-V.SCROLLBAR:

Dikey olarak oklari yukari asagi çalismasini saglar.



compenent ve formda görünüsü

Properties - VScroll1	
VScroll1 VScrollBar	
Alphabetic Categorized	
(Name)	VScroll1
CausesValidation	True
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Height	615
HelpContextID	0
Index	
LargeChange	1
Left	2040
Max	32767
Min	0
MouseIcon	(None)
MousePointer	0 - Default
RightToLeft	False
SmallChange	1
TabIndex	0
TabStop	True
Tag	
Top	2040
Value	0
Visible	True
WhatsThisHelpID	0
Width	1695

V.ScrollBar Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Validate olayinin çalismasini belirler
- Sürükleme olayinda mouse'nin seklini belirler
- Sürükleme olayinin nasil olacagini belirler
- Clicklenebilirliğini belirler
- Nesnenin yüksekliğini belirler
- Yardim dosyasindaki konu numarası belirlenir
- Index'i belirler
- Kaydırma çubuguna tiklandiginda ne kadarlik degisecegi
- Sola hizayi belirler
- Kaydırma çubugunun alacagi max deger
- Kaydırma çubugunun alacagi min deger
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki sekil
- Kaydırma çubugunu ikitaraftaki oklarla ne kadarlik degisecegi
- Kaçinci TAB ta ulasilacagini belirler
- TAB ile seçilemeyecegini belirler
- Mesaj saklamak için kullanilir
- Üstten hizalama
- Kaydırma çubugunun temsil ettiigi deger
- Nesnenin görünüp görünmeyecegi

V.SCROLL'IN METOTLARI

V.SCROLL'in farklı metotları şunlardır:

LargeChange:Kaydırma çubuğu üzerine tıklanması durumunda H.Scroll'un ne kadarlık bir değişime tabi tutulacağını belirler.

Max,Min:Kaydırma çubuğunun alabileceği maksimum ve minimum değerdir.

SmallChange:Kaydırma çubuğunu iki kenarındaki oklarla H.Scroll'un ne kadarlık bir değişime tabi tutulacağını belirler.

Value:Kaydırma çubuğunun temsil ettiği değeri gösterir. Bu değer max ile min arasında bir sayıdır ve kaydırma çubuğu üzerinde değişik şekillerde değiştirilebilir.

V.SCROLL'IN OLAYLARI

V.SCROLL'in farklı olayları şunlardır:

V.Scroll1_Change():Nesne içeriği üzerinde değişiklik yapıldığı zaman çalışır.

V.Scroll1_DragDrop(Source As Control, X As Single, Y As Single):Olay sürükleme olayı metin kutusunda bittiginde çalışır.Soruce sürükleme olayının başladığı nesneyi temsil eder.X ve Y mouse'un koordinat değerlerini verir.

V.Scroll1_DragOver(Source As Control, X As Single, Y As Single,State As Integer):Olay sürükleme esnasında Mouse nesne üzerinden geçerken çalışır.State ise sürükleme olayının hangi aşamada olduğu değerini verir.State özelliği üç değer tasir.

V.Scroll1_GotFocus():Form üzerinde nesne aktif nesneyken çalışır.

V.Scroll1_KeyDown(Key Code As Integer, Shift As Integer):Olay veri girişi esnasında klavyenin tusuna basıldığı an

çalışır.KeyCode klavyeden basılan tusun değerini tasir.Shift Ctrl, Alt ve Shift tuslarına karsilik deger üretir.Bunlar: 1-Shift, 2-Ctrl ve 4-Alt degerleridir.Beraber basildiklarinda gelen deger toplamlarindan olusur.Örneğin Ctrl+Alt=6'dir.

V.Scroll1_KeyPress(KeyAscii As Integer):Olay veri girişi esnasında KeyDown olayından sonra çalışır.KeyAscii klavyeden basılan tusların ASCii karşılığını verir.

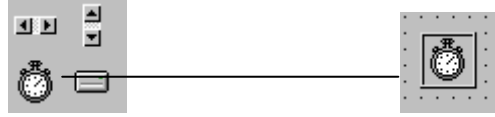
V.Scroll1_KeyUp(KeyCode As Integer,Shift As Integer):Olay KeyPress olayından sonra, klavyede basılan tus bırakıldıktan sonra çalışır.

V.Scroll1_LostFocus():Nesne aktif olmadığında yani terkedildiğinde çalışır.

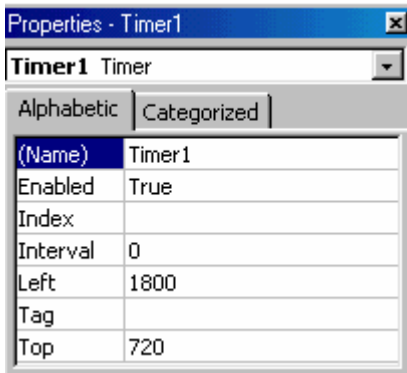
V.Scroll1_Scroll():Farklı liste elemanı seçildiğinde çalışır

12-TIMER:

Programda belirli süre aralıklarla aktif hale gelip belirli işleri yapabilmek amacıyla kullanılan bir kontroldür. Bu kontrol tasarım zamanı ekranda görülmesine rağmen çalışma esnasında görülmez.



compenent ve formda görünüşü



Timer Nesnesini Özellikleri

çili nesnenin ismidir
Timer'i etkinleştirme
Index'i belirler

Timer'in gerçekleştireceği zaman aralığı
Sola hizayı belirler
Mesaj saklamak için

Üstten hizalama

TIMER'IN METOTLARI

TIMER'in methotlari sunlardir:

Enabled:False verilerek Timer nesnesinin alısmasi durdurulur. Tekrar True yapilincaya kadar Timer olayi meydana gelmez.

Interval:Timer olayinin gereklesecegi milisaniye cinsinden zaman periyodudur. Alabilecegi degerler 1-65535 arasidir. 0 degeri Timer'i pasif hale getirir.

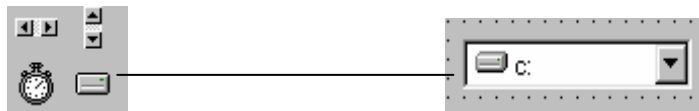
TIMER'IN OLAYLARI

TIMER'in olayi sundur:

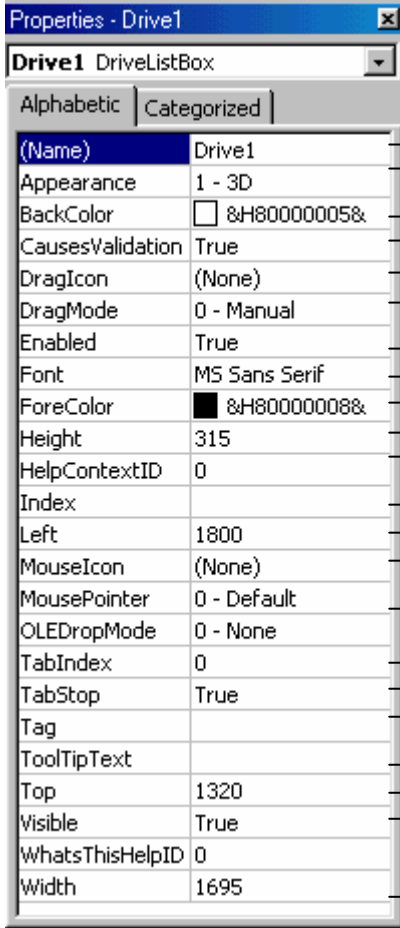
Timer1_Timer():Timer kontrolünün interval özelliği ile belirtilen süre içerisinde periodik olarak bu olay meydana gelir. Bu olay içerisinde yazılacak kodun hızlı olması gerekir. Bu olay periyodik olarak sürekli meydana geleceği için bu olay içerisinde yazılan kodun uzun olması Windows altında çalışan diğer programlarında yavaşlamasına sebep olacaktır. Ayrıca Timer olayına yazacağınız kodun çalışması interval özelliğine verdiğiniz süreden daha uzun sürerse bu arada meydana gelmesi gereken timer olayları meydana gelmez.

13-DRIVELISTBOX:

Sistemde bulunan sürücülerini listelemeye yarayan,ComboBox kontrolünden türemiş bir kontrol elemanıdır. Bunun vasıtasıyla programın çalışması esnasında istenilen sürücü seçimi yapılır. Bu seçim sürücüye geçişi sağlamaz.



compenent ve formda görünüşü



DriveListBox Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Formdaki görüntüsünü belirler
- Arka zemin rengini belirler
- Validate olayının çalışmasını belirler
- Sürükleme olayında mouse'nin şeklini belirler
- Sürükleme olayının nasıl olacağını belirler
- Clicklenebilirliğini belirler
- Yazı tipini belirler
- Yazı rengini belirler
- Nesnenin yüksekliğini belirler
- Yardıma dosyasındaki konu numarası belirlenir
- Index'i belirler
- Sola hizayı belirler
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki şekil
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- Gizli mesaj vermek için kullanılır
- Üstten hizalama
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

DRIVELISTBOX'IN METOTLARI

DRIVELISTBOX'in farklı metotları vardır:

Drive: Bu özellik kullanılarak kontrolün gösterdiği aktif sürücü öğrenilebilir veya değiştirilebilir. Drive özelliğinin gösterdiği değer disket sürücüler için sürücü harfi ve : isareti iken, harddisk sürücülerde buna sürücü etiketide dahildir.

DRIVELISTBOX'IN OLAYLARI

DRIVELISTBOX'in farklı olayları vardır:

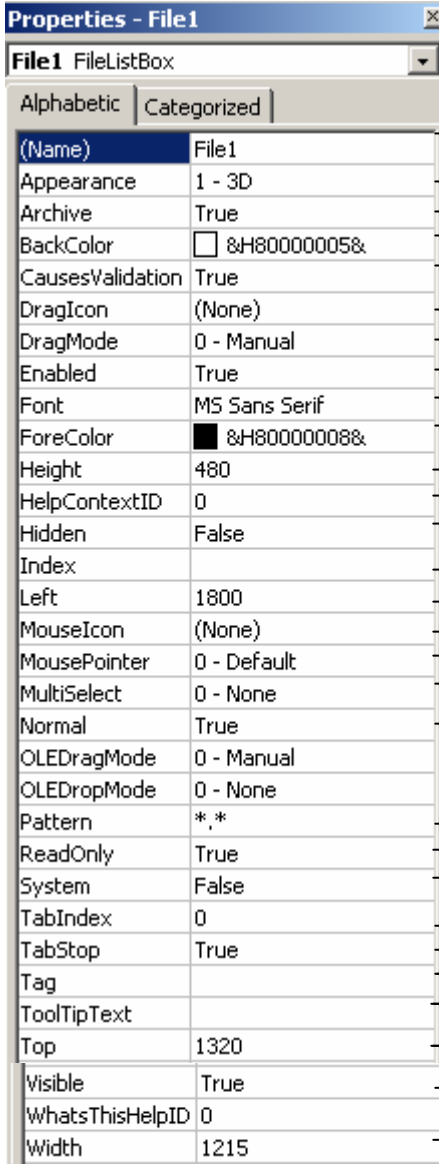
DriveListBox1_Change(): Kontrolde bir sürücü seçilmesi ile bu olay meydana gelir. Sürücü seçimi yapıldığında yapılması gereken kod buraya yazılmalıdır.

14-FILELISTBOX:

Herhangi bir dizindeki dosyaları listelemeye yarayan, ListBox kontrolünden türemiş bir kontrol elemanıdır.



compenent ve formda görünüşü



FileListBox Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Formdaki görüntüsünü belirler
- Arsiv dosyalarının görüntülenmesi
- Arka zemin rengini belirler
- Validate olayının çalışmasını belirler
- Sürükleme olayında mouse'nin şeklini belirler
- Sürükleme olayının nasıl olacağını belirler
- Clicklenebilirliğini belirler
- Yazı tipini belirler
- Yazı rengini belirler
- Nesnenin yüksekliğini belirler
- Yardım dosyasındaki konu numarası belirlenir
- Saklı dosyaların görüntülenmesi
- Index'i belirler
- Sola hizayı belirler
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki şekil
- Listeden birkaç seçenek seçmeyi sağlar
- Normal dosyaların görüntülenmesi
- Listelenecek dosyalar filtrelenir
- Sadece okunur dosyalar görüntülenir
- Sistem dosyaları görüntülenir.
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- Gizli mesaj vermek için kullanılır
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

FILELISTBOX'IN METOTLARI

FILELISTBOX'ın farklı metotları şunlardır:

Archive,Hidden,Normal,ReadOnly,System: Bu özellikler kullanarak dosya listeleme kutusunda Arsiv,Saklı,Sadece Okunur,Normal ve System özelliklerine sahip dosyaların görüntülenmesi veya görüntülenmemesi

saglanır. Bu özelliklerden herhangi birisine Trie degeri verildiginde o özellige sahip dosyalar listelenir,False degeri verildiginde listelenmez.

FileName:Bu özellik ile seçili dosyanin ismi öğrenilebilir. Geri dönecek deger dosyanin adi ve uzantisidir.

Path:Bu özellik ile listelenmek istenen sürücü ve dizin ismi belirlenir. Örneğin Windows dizinindeki dosyalari göstermesini istiyorsak File1.Path="C:\windows" atamasini yapabiliriz.

Pattern:Bu özellik ile listelenmek istenen dosyalar filtrelenir. Bu işlem bildigimiz joker karakterle(?,*) yapılır.

Örneğin yalnız EXE uzantili dosyalari görüntülemek için:

File1.Pattern="*.EXE" atamasi yapılır.

Birden fazla tip belirlemek için ";" karakteri kullanilir. EXE ve COM dosyalarini listelemek için;

File1.Pattern="*.EXE;*.COM" seklinde kullanilir.

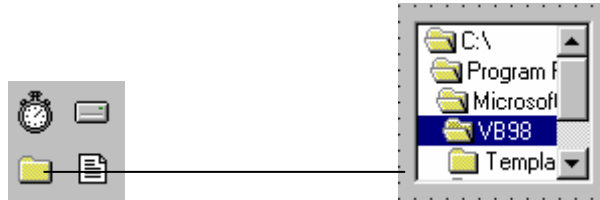
FILELISTBOX'IN OLAYLARI

FILELISTBOX'in farklı olaylari şunlardır:

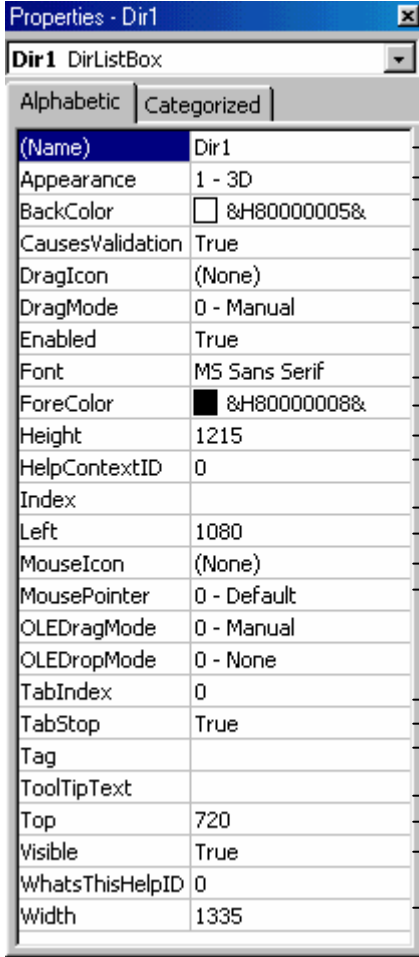
FileListBox1_PathChange():Bu olay Path özelliğinin değiştirilmesi sonucu meydana gelir. Yani FileListBox farklı bir yeri göstermeye başladığında meydana gelir.

FileListBox1_PatternChange():Bu olay Pattern özelliğinin değiştirilmesi sonucu meydana gelir. Yani FileListBox farklı türlerdeki dosyalari listelemeye başladığında bu olay meydana gelir.

15-DIRLISTBOX:Sistemde bulunan sürücülerdeki dizinleri listelemeye yarayan,ListBox kontrolünden türemiş bir kontrol elemanıdır.



compenent ve formda görünüşü



DirListBox Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Formdaki görüntüsünü belirler
- Arka zemin rengini belirler
- Validate olayının çalışmasını belirler
- Sürükleme olayında mouse'nin şeklini belirler
- Sürükleme olayının nasıl olacağını belirler
- Clicklenebilirliğini belirler
- Yazı tipini belirler
- Yazı rengini belirler
- Nesnenin yüksekliğini belirler
- Yardım dosyasındaki konu numarası belirlenir
- Index'i belirler
- Sola hizayı belirler
- Mouse'un nesne üzerindeki resim
- Mouse'un nesne üzerindeki şekil
- Kaçıncı TAB ta ulaşılacağını belirler
- TAB ile seçilemeyeceğini belirler
- Mesaj saklamak için kullanılır
- Gizli mesaj vermek için kullanılır
- Üstten hizalama
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

DIRLISTBOX'IN METOTLARI

DIRLISTBOX'in farklı metotları vardır:

Path: Bu özellik kullanılarak seçilmiş olan dizin öğrenilebilir ve değiştirilebilir. Bu özellikten dönen değer sürücü ismi dahil tam yoldur. Kontrolün göstermesi istenen sürücüde, yine sürücü isminin bu özelliğe atanması ile gerçekleştirilir.

DIRLISTBOX'IN OLAYLARI

DIRLISTBOX'in farklı olayları vardır:

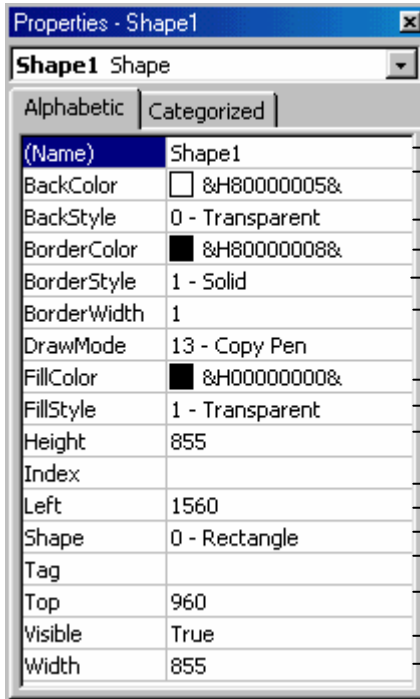
DirListBox1_Change(): Listedeki bir dizin seçilmesi ile bu olay meydana gelir.

16-SHAPE:

Shape grafiksel bir kontrol elemanı olup dikdörtgen,kare,elips,çember,oval kare ve oval dikdörtgen şekillerini oluşturmaya yarar.



compenent ve formda görünüsü



Shape Nesnesini Özellikleri

- Seçili nesnenin ismidir.
- Arka zemin rengini belirler
- Zeminin görünüp görünmeyeceği
- Çizgi rengini belirtir
- Çizginin durumunu belirler
- Çizgi kalınlığını belirler
- Seklin iç boyama rengi
- Seklin içini boyamak için kullanılacak desen
- Nesnenin yüksekliğini belirler
- Index'i belirler
- Sola hizayı belirler
- Sekli belirler
- Mesaj saklamak için kullanılır
- Üstten hizalama
- Nesnenin görünüp görünmeyeceği
- Nesnenin genişliğini belirler

SHAPE'İN METOTLARI

SHAPE'in farklı metotları şunlardır:

Shape: Shape kontrolünün çizeceği şekli belirler. 0-5 arası bir değer alabilir.

- | | |
|---------------|--------------------|
| 0: Dikdörtgen | 3: Çember |
| 1: Kare | 4: Oval dikdörtgen |
| 2: Elips | 5: Oval kare |

BorderStyle:Bu özellik nesnenin çerçeve biçimini belirler. Çerçeve biçimleri ve BorderStyle özelliğinin aldığı değerler aşağıda gösterilmiştir.

BorderStyle	Çerçeve Biçimi
0	Zemin rengiyle uyumlu,görülmez
1	Solid (tam çerçeve)
2	Dash (çizgi)
3	Dot (nokta)
4	Dash dot (çizgi,nokta)
5	Dash dot dot (çizgi,nokta,nokta)
6	Inside solid (şekil ile çerçeve kenarları çakışık)

BorderColor:Bu özellik nesnenin çerçeve rengini belirler.

BorderWidth:Bu özellik çerçeve kalınlığını ifade eder.1 ile 8192 arasındaki değerleri alabilir.BorderStyle özelliğinin sadece 1-Solid ve 6-Inside solid biçimlerindeyken etkisi görülür.

FillColor:Seklin iç boyama rengini belirler.

FillStyle:Seklin içini boyamak için kullanılacak deseni belirler.

FillStyle	Boyama Sekli
0	Tam Dolu
1	Üzerinde bulunduğu nesnenin zemin rengi
2	Yatay Çizgili
3	Dikey Çizgili
4	Sola Eğik
5	Sağa Eğik
6	Kareli
7	Çapraz

BackStyle:Bu özellik nesnenin arka zemin şeklini belirler.

ÖRNEK:Aşağıda verilen örnekte Timer aracılığıyla her 1sn'de bir şekil çizdirilmektedir. Bu olay programa eklenen bir kodla sürekli tekrarlatılmaktadır. Programda dikey kaydırma çubuğu,Shape ve Timer kullanıldı.

```
Private Sub Form_Load ()  
    Shape1.BorderStyle = 1  
    Shape1.BorderWidth = 5 'çizgi kalınlığı 5  
    Timer1.Interval = 1000  
    VScroll1.Min = 0  
    VScroll1.Max = 5  
End Sub
```

```
Private Sub VScroll1_Change ()
    Shape1.Shape = VScroll1.Value
End Sub
```

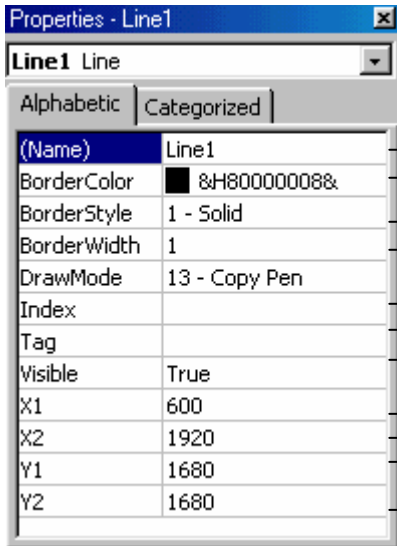
```
Private Sub Timer1_Timer ()
    VScroll1.Value = VScroll1.Value + 1
    'Tüm şekiller gösterildi ise gösterime bastan itibaren devam et
    If VScroll1.Value >= 5 Then VScroll1.Value = 0
End Sub
```

17-LINE:

Bu kontrol elemanı form üzerine bir çizgi şeklinde alınır. Alınan bu çizgiyi yatay, dikey ve eğik bir biçimde göstermek, istenilen boyuta getirmek mümkündür.



component ve formda görünüşü



Line Nesnesini Özellikleri

Seçili nesnenin ismidir.
Çizgi rengini belirtir

Çizginin durumunu belirler
Çizgi kalınlığını belirler

Index'i belirler
Mesaj saklamak için kullanılır
Nesnenin görünüp görünmeyeceği

X1 koordinatını belirler
X2 koordinatını belirler
Y1 koordinatını belirler

LINE'IN METOTLARI

LINE'in farklı metotları şudur:

X1,X2,Y1,Y2: X1 ve Y1 noktaları çizgi kontrol elemanlarının başlangıç koordinatlarını, X2 ve Y2 ise bitiş koordinatlarını belirler. Yatay kontrolleri X1 ve X2, dikey kontrolleri ise Y1 ve Y2 belirler.

18-Image Kontrolü

Bu eleman grafiksel bir kontrol elemanı olup resimleri görüntülemek, boyutlandırmak ve tasamak için kullanılır. PictureBox' dan daha hızlı ve daha az sistem kaynağı kullanmasına rağmen PictureBox kadar gelişmiş özelliklere sahip değildir. PictureBox' dan farklı olarak gruplandırma yapamaz ve metodları kullanılarak yazım ve çizim yapamaz.

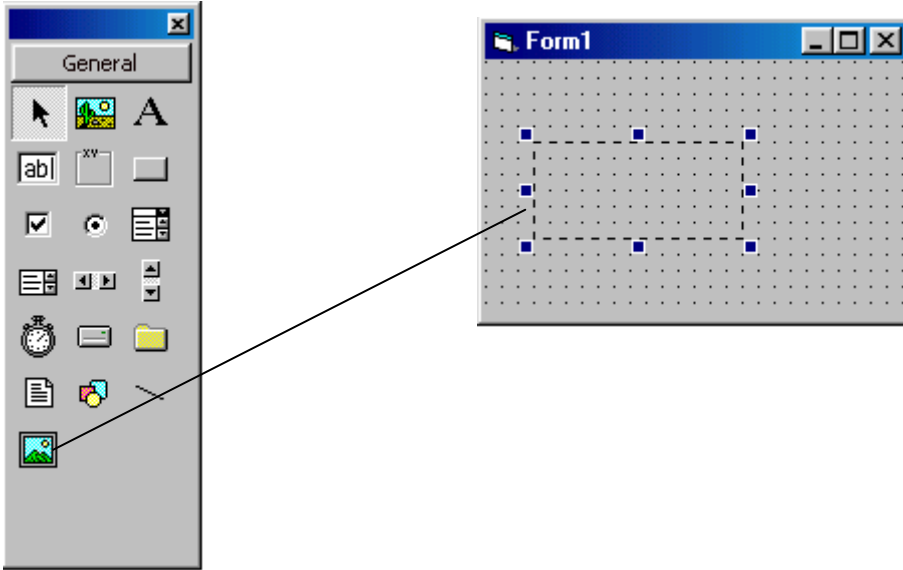


Image Properties (Özellikler)

Image1 Image	
Alphabetic Categorized	
(Name)	Image1
Appearance	1 - 3D
BorderStyle	0 - None
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Height	735
Index	
Left	360
MouseIcon	(None)
MousePointer	0 - Default
OLEDragMode	0 - Manual
OLEDropMode	0 - None
Picture	(None)
Stretch	False
Tag	
ToolTipText	
Top	600
Visible	True
WhatsThisHelpID	0
Width	1575
OLEDragMode Returns/Sets whether this object can act as an OLE drag/drop source, and whether this process is started	

Nesnenin programdaki ismini belirler.

Nesnenin formdaki görünüsünü belirler.

Nesneye ait çerçevenin tipini belirler.

Nesnenin veritabanındaki alanini belirler.

Veri biçimini belirler.

Daha önceden bağlanılan alan varsa bulur.

Nesnenin bağlı olduğu datanın adını belirtir.

(Drag/Drop) Sürükle bırak işlemi sırasında oluşacak fare imlecini belirler.

Sürükle/bırak işleminin otomatik mi yoksa el yordamı ile mi gerçekleştirileceğini belirler.

Nesnenin aktif olup olmadığını belirler.

Nesnenin boyunu belirler.

Aynı ad altında toplanan dizi nesnelerin indekslerini gösterir.

Nesnenin sol hizasını belirler.

Fare imlecini belirler.

Nesne üzerindeyken farenin alacağı imleci belirler.

Nesne içindeki resmi belirler.

Resmi nesne boyutuna gelip gelmeyeceğini belirler.

Nesne için gerekli olan ek metin bilgisini içerir.

Fare nesne üzerindeyken verilecek mesajı belirler.

Nesnenin form üzerinde üst hizasını belirler.

Nesnenin form üzerinde görünürlüğünü belirler.

"Bu nedir?" sorusunun referansı olan yardım dosyasına ait numarayı içerir.

Nesnenin genişliğini belirler.

Image Methods (Metodlar)

Appearance : Nesnenin görünüsü ile özelliklerin biçimini sağlar. Sadece tasarım asamasında kullanıcıya açıktır.

Örnek : Image1.Appearance = 0
Image1.Appearance = 1

BorderStyle : Nesneye ait çerçevenin tipini belirler.

Örnek : Image1.BorderStyle = 0
Image1.BorderStyle = 1

Container : Nesne içeriğinin ne tip bir veri yapısı olduğunu tanımlar.

Örnek : Image1.Container = Picture1

DataChanged : Herhangi bir işlem sonrası, nesne data kontrolüne bağlıysa içeriğinde bir değişiklik olup olmadığını bildirir.

Örnek : Image1.DataChanged = True / False

DataField : Kontrol data nesnesine bağlıysa, veritabanı dosyasındaki hangi alana (Field' a) bağlı olduğunu gösterir.

Örnek : Image1.DataField = "Alan Adı"

DataFormat : Nesnenin veri biçimini belirler.

Örnek : Image1.DataFormat = Picture

DataMember : Daha önceden üye olunan bir data varsa buradan belirlenir.

Örnek : Image1. DataMember = "Products"

DataSource : Nesnenin bağlı olduğu Data kontrolünü içerir.

Örnek : Image1.Datasource = Data1

Drag : Sürükleme işleminin başlatılmasını, sonlandırılmasını ve iptalini belirler.

Örnek : Image1.Drag = 0
 Image1.Drag = 1
 Image1.Drag = 2

DragIcon : Sürükle/birak islemi sırasında fare imlecinin hangi sembolü göstermesi gerektiğini belirler.

Örnek : Image1.DragIcon = Load Picture ()

DragMode : Drag / Drop (sürükle-birak) işleminin el yordami ile mi yoksa otomatik olarak mı gerçekleştirileceğini belirtir.

Örnek : Image1.DragMode = 0 (Manual)
 Image1.DragMode = 1 (Automatic)

Enabled : Nesnenin aktif olup olmayacağını belirleyen metoddur.

Örnek : Image1.Enabled = True / False

Height : Nesnenin boyunu gösterir.

Örnek : Image1.Height = 1440

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Örnek : Image1.Index = index no

Left : Nesnenin, üzerinde bulunduğu nesneden yatay olarak ne kadar uzakta olduğunu gösterir. Bu özellik, daima nesnenin sol kenarına bağlı x koordinatını verir.

Örnek: Image1.Left = 360

MouseIcon : Fare imlecini belirler. Bu metodun 99' a esitlenmesi ise farenin programcı tarafından tanımlayacağı özel bir imleci göstereceğini yansıtır.

Örnek : Image1.MouseIcon = LoadPicture ("c:\...")

MousePointer : Farenin, nesne üzerinde dolasırken hangi imleci kullanacağını belirler.

Örnek : Image1.MousePointer = 0 – 15

Move : Nesnenin ekrandaki konumlarında degisiklik yapmasina izin veir.

Örnek: Image1.Move left,top

Name : Nesnenin adini gösterir.

Örnek : Image1.Name = resim1

OleDrag : Sürükle bırak isleminin otomatik mi yoksa el yordamiyla mi olacagini belirler.

Örnek: Image1.OleDrag = 0 (Manual)
 Image1.OleDrag = 1 (Automatic)

OleDragMODE : Sadece sürükleme isleminin hareket tipi için seçenekler sunar.

Örnek : Image1.OleDragMODE = 0 / 1

OleDropMODE : Sadece bırak isleminin hareket tipi için seçenekler sunar.

Örnek : Image1.OleDropMODE = 0 / 1 / 2

Parent : Nesnenin hangi nesne üzerinde bulunduğunu veya hangi nesneye bağlı olduğunu bildirir.

Örnek : Image1.Parent.MousePointer = 4

Picture : Nesne içeriği olan resim dosyasının bulunduğu adresi içerir.

Örnek : Image1.Picture =LoadPicture ("C:\...")

Refresh : Nesnenin güncelleştirilmesini sağlar.

Örnek : Image1.Refresh

ShowWhatsThis : Bu nesir imlecini gösterir.

Örnek: Image1.ShowWhatsThis

Stretch : Resmin nesne boyutuna gelip gelmeyeceğini belirler.

Örnek: Image1.Stretch = True / False

Tag : Nesne için gereksinim duyulan ek metin bilgisini gösterir.

Örnek: `Image1.Tag = ("c:\...")`

ToolTipText : Fare nesne üzerindeyken verilecek mesajı belirler.

Örnek : `Image1.ToolTipText = "mesaj"`

Top : Nesnenin üzerinde bulunduğu ana nesneden düşey olarak ne kadar uzakta olduğunu belirtir.

Örnek : `Image1.Top = deger`

Visible : Nesnenin form çalışırken görünüp görünmeyeceğini belirler.

Örnek : `Image1.Visible = True / False`

Width : Nesnenin genişliğini belirler.

Örnek : `Image1.Width = 3000`

ZOrder : Nesnenin diğer nesnelere göre ne kadar üstte ya da arkada olduğunu belirler. Nesneler arası geçiş öncelik tanımlamaları için ideal bir metod.

Örnek : `Image1.ZOrder = 0 (ön)`
 `Image1.ZOrder = 1 (arka)`

Image Events (Olaylar)

Private Sub Image1_Click () : Olay image nesnesi üzerine fare ile bir kere tıklandığında gerçekleşir.

Private Sub Image1_DbClick () : Olay image nesnesi üzerine fare ile çift tıklandığında gerçekleşir.

Private Sub Image1_DragDrop () : Olay sürükleme olayı image nesnesi üzerinde bittiginde çalışır.

Private Sub Image1_DragOver () : Olay sürükleme esnasında fare nesne üzerinden geçerken gerçekleşir.

Private Sub Image1_MouseDown () : Olay image nesnesi üzerinde farenin herhangi bir butonuna tiklanmasiyla gerçekleşir.

Private Sub Image1_MouseMove () : Olay farenin nesne üzerinde gezindigi tüm süre boyunca gerçekleşir. LostFocus olayi ile bu olay geçerliliğini yitirir.

Private Sub Image1_MouseUp () : Butonuna basili olan farenin butonunun bırakilmasiyla gerçekleşir.

Private Sub Image1_OLECompleteDrag () : Sürükleme islemi tamamlandi-ginda ya da iptal edildiginde gerçekleşen olaydir.

Private Sub Image1_OLEDragDrop () : OLEDropMODE özelligine Manuel verilmissse bırakma islemi gerçeklestiginde meydana gelen olaydir.

Private Sub Image1_OLEDragOver () : OLEDropMODE özelligine Manuel verilmemisse sürükleme islemi gerçeklestiginde meydana gelen olaydir.

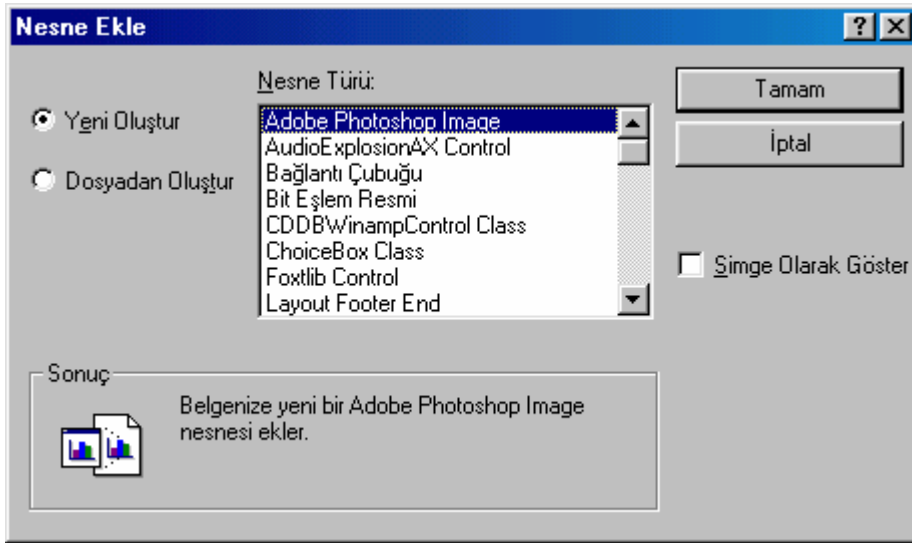
Private Sub Image1_OLEGiveFeedBack () : Bu olay, OLE kaynaginın sürükle ve bırak islemi sirasinda fare seklinin degismesi gerektiği durumlarda gerçekleşir.

Private Sub Image1_OLEStartDrag () : Sürükleme islemi basladiginda bu olay gerçekleşir.

19-OLE Kontrolü

OLE teknolojisini destekleyen iki program arasında bağlantı kurmayı sağlar. Bağlantının sonucunda veri transferi yapmak ya da verileri kullanmak mümkün olmaktadır.

OLE nesnesini form üzerine aldiginizda standart OLE diyalog kutusu karsimiza otomatik olarak çıkacaktır.



Bu pencerede OLE' yi destekleyen programların listesi yer almaktadır. Buradan herhangi birisi seçildiği zaman o programla direkt bağlantı kurularak programın kaynağına ulaşılır. Bu listedeki OLE' yi destekleyen programlardan biri seçilerek o uygulama ile bağlantı kurulabileceği gibi Dosyadan Yarat düğmesi kullanılarak da daha önce kaydedilmiş bir dosya ile bağlantı kurulabilir.

OLE işlemlerinde bir nesne iki türlü olabilir. Bunlar Linked (Bağlantılı) ve Embedded (Gömülü) olarak adlandırılır. Linked olarak kurulan bağlantılarda uygulama ile nesne arasında paylaşılabilecek veri alınır. Embedded olarak kurulan ilişkilerde ise nesnenin paylaşılabilecek verisi ile birlikte onu açıp düzenleyecek kodda alınır. Linked olarak kurulan bağlantılarda sadece veri alındığı için daha az yer kaplayacaktır ancak bu tip bağlantılarla oluşturulmuş dosyaların başka bilgisayara götürülmesi durumunda orada da çalışacağı garanti edilemez. Çünkü bağlantının kurulduğu program o bilgisayarda bulunmayabilir. Embedded olarak ilişki kurulduğunda ise veri ile birlikte programı düzenleyecek kodda nesneye dahil olduğu için bu tip dosyalar rahatlıkla diğer bilgisayarlarda götürülüp çalıştırılabilir. Embedded olarak kaydedilen OLE dosyaları Linked olanlara göre daha çok yer kaplayacaktır.

OLE Properties (Özellikler)

Properties - OLE1	
OLE1 OLE	
Alphabetic Categorized	
(Name)	OLE1
Appearance	1 - 3D
AutoActivate	2 - DoubleClick

AutoVerbMenu	True
--------------	------

BackColor	<input type="checkbox"/> &H80000005&
BackStyle	1 - Opaque
BorderStyle	1 - Fixed Single
CausesValidation	True
Class	
DataField	

DataSource	
------------	--

DisplayType	0 - Content
DragIcon	(None)

DragMode	0 - Manual
----------	------------

Enabled	True
Height	975
HelpContextID	0

HostName	
----------	--

Index	
Left	600
MiscFlags	0
MouseIcon	(None)
MousePointer	0 - Default
OLEDropAllowed	False
OLETypeAllowed	2 - Either

Programda kullanılan ismini belirler .
Nesnenin görünüşü ile ilgili özellikler belirlenir
Nesnenin aktif hale gelmesi için kullanacağı
metodu belirler
OLE' yi destekleyen uygulamaların Verb
menülerini belirler.
Nesnenin zemininin rengini belirler.
Nesnenin arka stilini belirler.
Nesneye ait çerçevenin tipini belirler.

Nesnenin sınıfı belirlenir.
Data nesnesine bağlı bir kontrolün, veritabanı
dosyasındaki hangi alana bağlı olduğunu
gösterir.

Nesnenin bağlı olduğu data kontrolünün adını
içerir.
Nesnenin gösterim şekli belirlenir.
Sürükle/bırak işlemi sırasında farenin hangi
imleci alacağını belirler.

Sürükle bırak işleminin el yordamı ile mi yoksa
otomatik olarak mı gerçekleştirileceğini belirler
Nesnenin etkin olup olmadığını belirler.
Nesnenin boyunu belirler.
İçerik bağımlı yardım dosyaları için, ilişki
kurulması istenen başlık numarasını içerir.

Nesnenin hangi program tarafından açıldığını
belirtmek için kullanılan isim.

İndeksli kullanımda indeksini bildirir.
Nesnenin sol hizasını belirler.

Fare imlecini belirler.
Nesne üzerine geldiğinde fare imlecini belirler.
Nesnenin fare ile hareketini belirler.

Nesnenin desteklediği bağlantı şeklini belirler.

SizeMode	0 - Clip
SourceDoc	
SourceItem	
TabIndex	0
TabStop	True
Tag	
Top	720
UpdateOptions	0 - Automatic
Verb	0
Visible	True
WhatsThisHelpID	0
Width	2655
MiscFlags	
Returns or sets a value that determines access to one or more additional features of the OLE	

Nesnenin nasıl boyutlandırılacağını belirler
Nesneyi dosya olarak kaydeder.

TAB tuşu ile kısayol sektirmelerinde nesnenin kaçinci sırada olduğunu gösterir.
Nesnenin TAB ini aktif eder.
Nesnenin ek metin bilgisini içerir.
Nesnenin üstten hizasını belirler.
Nesne içindeki veriyi yeniden günceller.
İşlem uygulamak için seçenekler sunar
Nesnenin görünürliğini belirler.
Bu nedir dosyasına ait numarayı içerir.
Nesnenin genişliğini belirler.

OLE Methodları

Action : Bu metoda değerler verilerek bazı olaylar gerçekleştirilir.

Örnek : OLE1.Action = 0 (Gömülü nesneyi oluşturur.)
 OLE1.Action = 1 (Dosya içindeki nesne ile bağlantı kurar.)
 OLE1.Action = 4 (Bir nesneyi panoya kopyalar.)
 OLE1.Action = 1 – 14

Appearance : Nesnenin görünüşü ile özelliklerin biçimini sağlar. Sadece tasarım aşamasında kullanıcıya açıktır.

Örnek : OLE1.Appearance = 0
 OLE1.Appearance = 1

AppIsRunning : OLE kontrolü içindeki uygulamanın o anda çalışıp çalışmadığını belirler.

Örnek : OLE1.AppIsRunning = True / False

AutoActivate : OLE kontrolünü aktif hale getirmek için kullanacağı metodları belirler.

Örnek : OLE1.AutoActivate = 0 (Manual)

OLE1.AutoActivate = 1 (Klavye kontrolü)

AutoVerbMenu : OLE kontrolü üzerinde iken sag fare tusunun kullanilmasi halinde Verb menüsünün otomatik olarak açilip açilmayacagi belirlenir.

Örnek : OLE1.AutoVerbMenu = True / False

BackColor : Nesnenin zemininin rengini belirler.

Örnek: OLE1.BackColor = &H0000C000&

BackStyle : Nesnenin arka stilini belirler.

Örnek : OLE1.BackStyle = 0 (Tranparent)
OLE1.BackStyle = 1 (Opaque)

BorderStyle : Nesneye ait çerçevenin tipini belirler.

Örnek : OLE1.Borderstyle = 0
OLE1.BorderStyle = 1

Class : Embedded object in sınıf ismini belirler.

Örnek : OLE1.Class = midfile

Close : Sadece gömme sonucundaki baglantilar üzerinde etkili olan bu metod OLE nesnesi ile kaynak arasindaki baglantiyi keser.

Örnek: OLE1.Close

Container : Nesne içeriğinin ne tip bir veri yapisi oldugunu tanımlar.

Örnek : OLE1.Container = Picture1

Copy : OLE elemani içindeki veriyi panoya kopyalar.

Örnek : OLE1.Close

CreateEmbed : Gömülü bir nesne olusturur.

Örnek : OLE1.CreateEmbed " ", class

CreateLink : Baglantili bir nesne olusturur.

Örnek : OLE1.CreateLink SourceDoc , SourceItem

Data : Nesneyi data kontrolüne bağlar.

Örnek : OLE1.Data

DataChanged : Herhangi bir işlem sonrası, nesne data kontrolüne bağlıysa içeriğinde bir değişiklik olup olmadığını bildirir.

Örnek : OLE1.DataChanged = True / False

DataField : Kontrol data nesnesine bağlıysa, veritabanı dosyasındaki hangi alana (Field' a) bağlı olduğunu gösterir.

Örnek : OLE1.DataField = "Alan Adı"

Delete : İlgili nesneyi silerek o nesne için ayrılan hafızayı boşaltır.

Örnek : OLE1.Delete

DisplayType : OLE kontrolünün form üzerinde nasıl gösterileceğini belirler.

Örnek : OLE1.DisplayType = 0

DoVerb : Nesneye girilen işlemi uygular.

Örnek : OLE1.DoVerb = -1 (Nesneyi düzenlemek için açar.)
 OLE1.DoVerb = -2 (Nesneyi kendi penceresinde açar.)
 OLE1.DoVerb = -3 (Gömülü uygulamayı gizler.)

Drag : Sürükleme işleminin başlatılmasını, sonlandırılmasını ve iptalini belirler.

Örnek : OLE1.Drag = 0
 OLE1.Drag = 1
 OLE1.Drag = 2

DragIcon : Sürükle/birak işlemi sırasında fare imlecinin hangi sembolü göster-mesi gerektiğini belirler.

Örnek : OLE1.DragIcon = Load Picture ()

DragMode : Drag / Drop (sürükle-birak) işleminin el yordamı ile mi yoksa otomatik olarak mı gerçekleştirileceğini belirtir.

Örnek : OLE1.DragMode = 0 (Manual)
 OLE1.DragMode = 1 (Automatic)

Enabled : Nesnenin aktif olup olmayacağını belirleyen metoddur.

Örnek : OLE1.Enabled = True / False

FetchVerbs : Nesnenin desteklediği fiilleri günceller.

Örnek: OLE1.FetchVerbs

FileNumber : Nesnenin kayıtlı olduğu dosya numarasını verir.

Örnek: OLE.FileNumber

Format: Hangi formatın kullanılacağı belirlenir.

Örnek : OLE1.Format = " "

Height : Nesnenin boyunu belirler.

Örnek : OLE1.Height = 1440

HostName: Nesnenin hangi program tarafından açıldığını belirtmek için kullanılır.

Örnek: OLE1.HostName = "Uygulama1"

HelpContextID : İçerik bağımlı yardım dosyaları için, ilişki kurulması istenen başlık numarasını içerir.

Örnek : OLE1.HelpContextID

hWnd : Nesneyi referans gösteren handle' i içerir. Bu özellik, API destekli ileri düzey Visual Basic programlamada kullanılmaktadır.

Örnek : OLE1.hWnd

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Örnek : OLE1.Index = index no

InsertObjDlg : Nesne yerleştirme diyalog kutusunu görüntüler.

Örnek : OLE1.InsertObjDlg

Left : Nesnenin, üzerinde bulunduğu nesneden yatay olarak ne kadar uzakta olduğunu gösterir. Bu özellik, daima nesnenin sol kenarına bağlı x koordinatını verir.

Örnek: OLE1.Left = 360

MouseIcon : Fare imlecini belirler. Bu metodun 99' a esitlenmesi ise farenin programcı tarafından tanımlayacağı özel bir imleci göstereceğini yansıtır.

Örnek : OLE1.MouseIcon = LoadPicture ("c:\...")

MousePointer : Farenin, nesne üzerinde dolasırken hangi imleci kullanacağını belirler.

Örnek : OLE1.MousePointer = 0 – 15

Move : Nesnenin ekrandaki konumlarında değişiklik yapmasına izin verir.

Örnek: OLE1.Move left,top

Name : Nesnenin adını gösterir.

Örnek : OLE1.Name = resim1

ObjectGetFormats : Nesnenin desteklediği formatı belirler.

Örnek : OLE1.ObjectGetFormats

ObjectGetFormatsCount : Nesnenin desteklediği formatı belirler.

Örnek: OLE1.ObjectGetFormatsCount -1

ObjectVerbs : OLE uygulamasında kullanılacak verb ler (fiil - işlem) belirlenir. Çoğunlukla bir menüde göstermek için kullanılır.

Örnek : OLE1.ObjectVerbs (0)

ObjectVerbsCount : OLe uygulamasının desteklediği verb sayısı belirlenir.

Örnek : OLE1.ObjectVerbsCount - 1

ObjectVerbFlags : ObjectVerb özelliği ile ilgili işlemlerin kontrolü bu metodla belirlenir.

Örnek : OLE1.ObjectVerbFlags (1)

OLEType : Nesnenin içerdigi bilginin tipi bu metodla belirlenir.

Örnek : OLE1.OLEType = 0 / 1 / 3

OLETypeAllowed : OLE nesnesinin desteklediği bağlantı şekli bu metodla belirlenir.

Örnek : OLE1.OLEType = 0 / 1 / 3

Parent : Nesnenin hangi nesne üzerinde bulunduğunu veya hangi nesneye bağlı olduğunu bildirir.

Örnek : OLE1.Parent.MousePointer = 4

PasteOK : Panodaki bilginin OLE nesnesi tarafından desteklenip desteklenmediği bu metodla belirlenir.

Örnek : OLE1.PasteOK = True

PasteSpecialDlg : Özel yapıştır diyalog kutusunu görüntüler.

Örnek : OLE1.PastespecialDlg

Picture : Nesne içeriği olan resim dosyasının bulunduğu adresi içerir.

Örnek : OLE1.Picture =LoadPicture ("C:\...")

ReadFromFile: Bir dosyaya kaydedilen nesneyi yükler.

Örnek : OLE1.ReadFromFile dosyano

Refresh : Nesnenin güncelleştirilmesini sağlar.

Örnek : OLE1.Refresh

SaveToFile : Bir nesneyi OLE formatında kaydeder.

Örnek: OLE1.SaveToFile

SetFocus : Konumlanma noktası.

Örnek : OLE1.SetFocus

ShowWhatsThis : Bu nesir imlecini gösterir.

Örnek: OLE1.ShowWhatsThis

Stretch : Resmin nesne boyutuna gelip gelmeyeceğini belirler.

Örnek: OLE1.Stretch = True / False

Tag : Nesne için gereksinim duyulan ek metin bilgisini gösterir.

Örnek: OLE1.Tag = ("c:\...")

ToolTipText : Fare nesne üzerindeyken verilecek mesajı belirler.

Örnek : OLE1.ToolTipText = "mesaj"

Top : Nesnenin üzerinde bulunduğu ana nesneden düzey olarak ne kadar uzakta olduğunu belirtir.

Örnek : OLE1.Top = deger

Update : OLE kontrolü içindeki veriyi yeniden günceller.

Örnek: OLE1.Update

Visible : Nesnenin form çalışırken görünüp görünmeyeceğini belirler.

Örnek : OLE1.Visible = True / False

Width : Nesnenin genişliğini belirler.

Örnek : OLE1.Width = 3000

ZOrder : Nesnenin diğer nesnelere göre ne kadar üstte ya da arkada olduğunu belirler. Nesneler arası geçiş öncelik tanımlamaları için ideal bir metod.

OLE Events (Olaylar)

Private Sub OLE1_Click () : Olay OLE nesnesi üzerine fare ile bir kere tiklandığında gerçekleşir.

Private Sub OLE1_DbClick () : Olay OLE nesnesi üzerine fare ile çift tiklandığında gerçekleşir.

Private Sub OLE1_DragDrop () : Olay sürükleme olayı OLE nesnesi üzerinde bittiginde çalışır.

Private Sub OLE1_DragOver () : Olay sürükleme esnasında fare nesne üzerinden geçerken gerçekleşir.

Private Sub OLE1_GotFocus () : Nesneye konumlanma anı.

Private Sub OLE1_KeyDown () : Klavyeden bir tusa basıldığı an.

Private Sub OLE1_KeyPress () : Herhangi bir tustan bir değer alındığında yapılması istenilen iş.

Private Sub OLE1_KeyUp () : Klavyede basılı tustan elin çekildiği an.

Private Sub OLE1_LostFocus () : Nesne üzerinden ayrılma anı.

Private Sub OLE1_MouseMove () : Olay farenin nesne üzerinde gezindiği tüm süre boyunca gerçekleşir. LostFocus olayı ile bu olay geçerliliğini yitirir.

Private Sub OLE1_MouseUp () : Butonuna basılı olan farenin butonunun bırakılmasıyla gerçekleşir.

Private Sub OLE1_ObjectMove() : OLE kontrolü taşındığında veya boyutlandırıldığında bu olay gerçekleşir.

Private Sub OLE1_Resize() : Nesnenin ilk kez görüntülendiğinde ya da boyutunda bir değişiklik yapıldığı an.

Private Sub OLE1_Updated() : OLE kontrolünün içeriği güncellendiğinde bu olay gerçekleşir.

20-DATA Kontrolü

Veritabanı dosyaları ile Visual Basic nesneleri arasında iletişimi sağlayan ve yönetim kontrolüdür.

DATA Properties

Alphabetic	Categorized
(Name)	Data1
Align	0 - None
Appearance	1 - 3D
BackColor	<input type="checkbox"/> &H80000005&
BOFAction	0 - Move First
Caption	Data1
Connect	Access
DatabaseName	
DefaultCursorType	0 - DefaultCursor
DefaultType	2 - UseJet
DragIcon	(None)

DragMode	0 - Manual
----------	------------

Enabled	True
EOFAction	0 - Move Last
Exclusive	False
Font	MS Sans Serif
ForeColor	<input checked="" type="checkbox"/> &H80000008&
Height	735
Index	
Left	1080
MouseIcon	(None)
MousePointer	0 - Default

Negotiate	False
OLEDropMode	0 - None
Options	0
ReadOnly	False
RecordsetType	1 - Dynaset
RecordSource	
RightToLeft	False
Tag	

Programda kullanılan ismini belirler .
Data nesnesinin hizasını belirler.
Nesnenin görünüşü ile ilgili özellikler belirlenir
Nesnenin zemininin rengini belirler.
Dosya başı hareketini belirler.
Data kontrolünün ismini belirler.
Veritabanı programı belirlenir.
Kullanılacak veritabanının ismi seçilir.

Bağlanılacak arabirimin tipini belirler.
Sürükle/birak işlemi sırasında farenin hangi imleci alacağını belirler.
Sürükle bırak işleminin el yordamı ile mi yoksa otomatik olarak mı gerçekleştirileceğini belirler
Nesnenin etkin olup olmadığını belirler.
Dosya sonu hareketini belirler.

Yazı tipini belirler.
Yazı rengini belirler.
Nesnenin boyunu belirler.
İndeksli kullanımda indeksini verir.
Nesnenin sol hizasını belirler.
Fare imlecini belirler.
Nesne üzerine geldiğinde fare imlecini belirler.

Fareyi bırak modu.
Seçenekler.
Sadece okunabilir özelliğini belirler.
Kayıt tipi.
Veritabanındaki kaydın adı.

Nesnenin ek metin bilgisini içerir.

ToolTipText	
Top	2280
Visible	True
WhatsThisHelpID	0
Width	2175

RecordSource
Returns/sets the underlying table, SQL statement, or QueryDef object for a

Fare nesne üzerindeyken verilecek mesajı belirler.

Nesnenin üstten hizasını belirler.

Nesnenin görünürlüğünü belirler.

Bu nedir dosyasına ait numarayı içerir.

Nesnenin genişliğini belirler.

DATA Methods

Align : Nesnenin form üzerindeki hizasını belirler.

Örnek : `Data1.Align = 1 (Align Top)`

Appearance : Nesnenin görünüşü ile özelliklerin biçimini sağlar. Sadece tasarım aşamasında kullanıcıya açıktır.

Örnek : `Data1.Appearance = 0`
`Data1.Appearance = 1`

BackColor : Nesnenin zemininin rengini belirler.

Örnek: `Data1.BackColor = &H0000C000&`

BOFAction : Dosya başı hareketini belirler.

Örnek : `Data1.BOFAction = 0 (MoveFirst)`
`Data1.BOFAction = 1 (BOF)`

Caption : Nesnenin adını belirler.

Örnek : `Data1.Caption = "veri1"`

Connect : Bağlanılacak veritabanı programı belirlenir.

Örnek : `Data1.Connect = Access`

Databasename : Bağlanılan veritabanının ismi .

Örnek : `Data1.Databasename = "tablo.mdb"`

Drag : Sürükleme işleminin başlatılmasını, sonlandırılmasını ve iptalini belirler.

Örnek : Data1.Drag = 0
 Data1.Drag = 1
 Data1.Drag = 2

DragIcon : Sürükle/birak işlemi sırasında fare imlecinin hangi sembolü göster-mesi gerektiğini belirler.

Örnek : Data1.DragIcon = Load Picture ()

DragMode : Drag / Drop (sürükle-birak) işleminin el yordami ile mi yoksa otomatik olarak mi gerçekleştirileceğini belirtir.

Örnek : Data1.DragMode = 0 (Manual)
 Data1.DragMode = 1 (Automatic)

Enabled : Nesnenin aktif olup olmayacağını belirleyen metoddur.

Örnek : Data1.Enabled = True / False

EOFAction : Dosya sonu hareketini belirler.

Örnek : Data1.EOFAction = 0

FontBold : Yazının kalınlığını belirler.

Örnek: Data1.FontBold = True

FontName : Yazının tipini belirler.

Örnek: Data.FontName = TimesNewRoman

Height : Nesnenin boyunu belirler.

Örnek : Data1.Height = 1440

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Örnek : Data1.Index = index no

Left : Nesnenin, üzerinde bulunduğu nesneden yatay olarak ne kadar uzakta olduğunu gösterir. Bu özellik, daima nesnenin sol kenarına bağlı x koordinatını verir.

Örnek: `Data1.Left = 360`

MouseIcon : Fare imlecini belirler. Bu metodun 99' a esitlenmesi ise farenin programcı tarafından tanımlayacağı özel bir imleci göstereceğini yansıtır.

Örnek : `Data1.MouseIcon = LoadPicture ("c:\...")`

MousePointer : Farenin, nesne üzerinde dolarken hangi imleci kullanacağını belirler.

Örnek : `Data1.MousePointer = 0 - 15`

Move : Nesnenin ekrandaki konumlarında değişiklik yapmasına izin verir.

Örnek: `Data1.Move left,top`

Name : Nesnenin programdaki adını gösterir.

Örnek : `Data1.Name = resim1`

OLEDropMode : Sürükle bırak işleminin bırakılmasının el yordamı ile mi yoksa otomatik mi olacağını belirler.

Örnek : `Data1.OLEDropMode = 1 / 2`

Parent : Nesnenin hangi nesne üzerinde bulunduğunu veya hangi nesneye bağlı olduğunu bildirir.

Örnek : `Data1.Parent.MousePointer = 4`

ReadOnly: Sadece okunabilirlik özelliğini belirler.

Örnek : `Data1.ReadOnly`

RecordSet : Kayıt ismi belirlenir.

Örnek : `Data1.RecordSet = "c:\..."`

RecordSetType : Kayıt tipini belirler.

Örnek : `Data1.RecordSetType = 0 / 1 / 2`

Refresh : Nesnenin güncelleştirilmesini sağlar.

Örnek : `Data1.Refresh`

ShowWhatsThis : Bu nesir imlecini gösterir.

Örnek: `Data1.ShowWhatsThis`

Tag : Nesne için gereksinim duyulan ek metin bilgisini gösterir.

Örnek: `Data1.Tag = ("c:\...")`

ToolTipText : Fare nesne üzerindeyken verilecek mesajı belirler.

Örnek : `Data1.ToolTipText = "mesaj"`

Top : Nesnenin üzerinde bulunduğu ana nesneden dikey olarak ne kadar uzakta olduğunu belirtir.

Örnek : `Data1.Top = deger`

UpdateRecord : Kayıtları günceller.

Örnek: `Data1.UpdateRecord`

Visible : Nesnenin form çalışırken görünüp görünmeyeceğini belirler.

Örnek : `Data1.Visible = True / False`

Width : Nesnenin genişliğini belirler.

Örnek : `Data1.Width = 3000`

ZOrder : Nesnenin diğer nesnelere göre ne kadar üstte ya da arkada olduğunu belirler. Nesneler arası geçiş öncelik tanımlamaları için ideal bir metod.

Örnek : `Data1.ZOrder = 0 (ön)`
 `Data1.ZOrder = 1 (arka)`

DATA Events (Olaylar)

Private Sub Data1_DragDrop () : Olay sürükleme olayı Data nesnesi üzerinde bittiginde çalışır.

Private Sub Data1_DragOver () : Olay sürükleme esnasında fare nesne üzerinden geçerken gerçekleşir.

Private Sub Data1_Error () : Bir hata meydana geldiği an.

Private Sub Data1_MouseMove () : Olay farenin nesne üzerinde gezindiği tüm süre boyunca gerçekleşir. LostFocus olayı ile bu olay geçerliliğini yitirir.

Private Sub Data1_MouseUp () : Butonuna basılı olan farenin butonunun bırakılmasıyla gerçekleşir.

Private Sub Data1_OLECompleteDrag () : Sürükleme işlemi tamamlandı-ğında ya da iptal edildiğinde gerçekleşen olaydır.

Private Sub Data1_OLEDragDrop () : OLEDropMODE özelliğine Manuel verilmişse bırakma işlemi gerçekleştiğinde meydana gelen olaydır.

Private Sub Data1_OLEDragOver () : OLEDropMODE özelliğine Manuel verilmemişse sürükleme işlemi gerçekleştiğinde meydana gelen olaydır.

Private Sub Data1_OLEGiveFeedBack () : Bu olay, OLE kaynağının sürük ve bırak işlemi sırasında fare şeklinin değişmesi gerektiği durumlarda gerçekleşir.

Private Sub Data1_OLEStartDrag () : Sürükleme işlemi başladığında bu olay gerçekleşir.

Private Sub Data1_Resize() : Nesnenin ilk kez görüntülendiğinde ya da boyutunda bir değişiklik yapıldığı an.

Private Sub Data1_Validate() : Farklı bir kayıt aktif kayıt anına geçmeden önceki an.

Private Sub Image1_Click () : Olay image nesnesi üzerine fare ile bir kere tıklandığında gerçekleşir.

Private Sub Image1_DbClick () : Olay image nesnesi üzerine fare ile çift tiklandığında gerçekleşir.

Private Sub Image1_DragDrop () : Olay sürükleme olayı image nesnesi üzerinde bittiginde çalışır.

Private Sub Image1_DragOver () : Olay sürükleme esnasında fare nesne üzerinden geçerken gerçekleşir.

Private Sub Image1_MouseDown () : Olay image nesnesi üzerinde farenin herhangi bir butonuna tiklanmasıyla gerçekleşir.

Private Sub Image1_MouseMove () : Olay farenin nesne üzerinde gezindiği tüm süre boyunca gerçekleşir. LostFocus olayı ile bu olay geçerliliğini yitirir.

Private Sub Image1_MouseUp () : Butonuna basılı olan farenin butonunun bırakılmasıyla gerçekleşir.

Private Sub Image1_OLECompleteDrag () : Sürükleme işlemi tamamlandı-gında ya da iptal edildiğinde gerçekleşen olaydır.

Private Sub Image1_OLEDragDrop () : OLEDropMODE özelliğine Manuel verilmişse bırakma işlemi gerçekleştiğinde meydana gelen olaydır.

Private Sub Image1_OLEDragOver () : OLEDropMODE özelliğine Manuel verilmemişse sürükleme işlemi gerçekleştiğinde meydana gelen olaydır.

Private Sub Image1_OLEGiveFeedBack () : Bu olay, OLE kaynağının sürüklen ve bırak işlemi sırasında fare şeklinin değişmesi gerektiği durumlarda gerçekleşir.

Private Sub Image1_OLEStartDrag () : Sürükleme işlemi başladığında bu olay gerçekleşir.

B-Microsoft Common Dialog Kontrolü

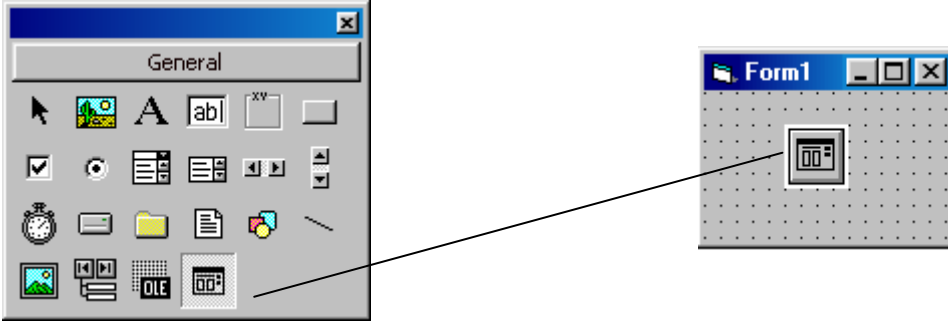
Windows tarafından sağlanan standart diyalog kutularının kullanımını sağlayan kontroldür. Bu kutular "Aç", "Yeni Adla Kaydet", "Renk", "Help", "Yazdır" ve "Font" diyalog kutularıdır.

Diyalog kutularının modal olması sebebiyle bir seçim yapılmadan form üzerinde işlem yapılamaz.

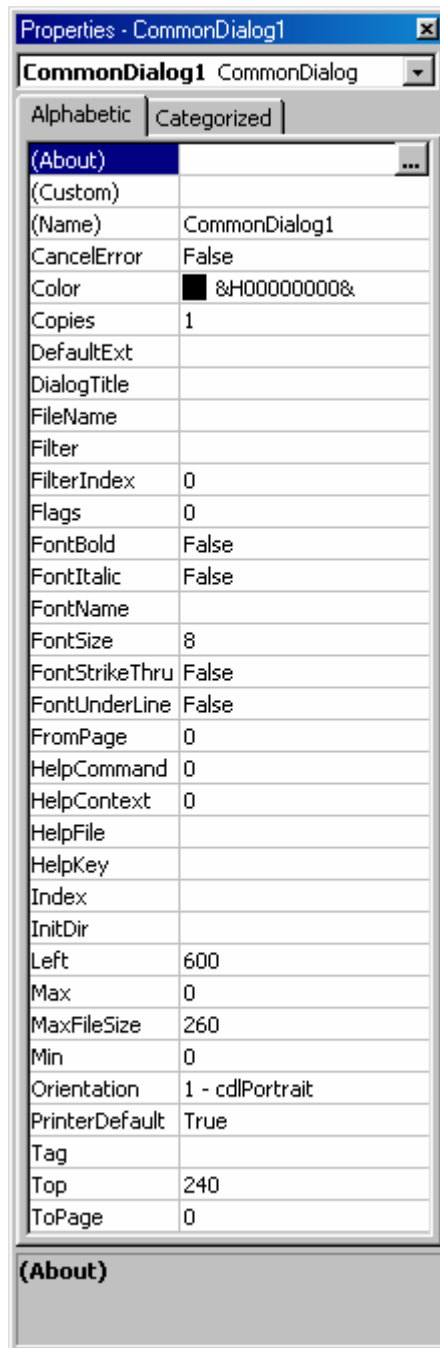
Aşağıda anlatılacak olan bu kontroller form üzerinde tasarım zamanı görülmesine rağmen çalışma esnasında görülmez. Bu kontroller, **Action** özelliklerine de atanmasıyla veya ilgili metod kullanılarak aktif hale gelirler.

CMDialog kontrolünde "İptal" düğmesinin seçilmesi halinde hata oluşturulup oluşturulmayacağını kullanıcının seçimine bırakmıştır. Olusacak hataların programlanmasından kaynaklanan bir hata (BUG) değildir. Sadece "İptal" düğmesinin seçildiğini programa belirtmek için düşünülmüş bir yöntemdir.

Toolbox

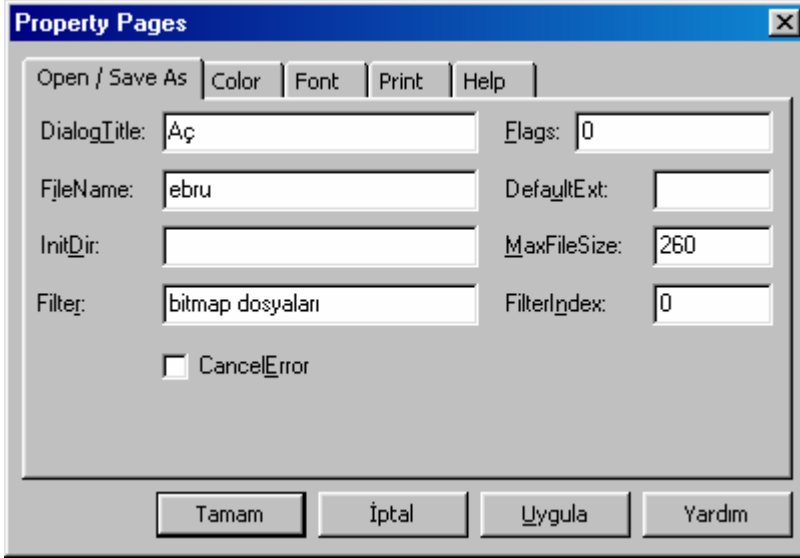


CommonDialog Properties (Özellikleri)



About: Diyalog penceresi telifi hakkında bilgi verir.

Custom: Altı diyalog kutusunun bazı önemli özellikleri buradan belirlenebilir. Custom özelliğine tiklandığında su pencere karsımıza çıkar.



Action: Kod sayfasında 1 – 6 arası bir deger verilerek ilgili diyalog kutusu açılır.

1: “Aç” diyalog kutusu

```
CommonDialog1.Action = 1  
veya  
CommonDialog1.ShowOpen
```

2: “Yeni Adla Kaydet” diyalog kutusu

```
CommonDialog1.Action = 2  
veya  
CommonDialog1.ShowSave
```

3: “Renk” diyalog kutusu

```
CommonDialog1.Action = 3  
veya  
CommonDialog1.ShowColor
```


4: "Font" diyalog kutusu

```
CommonDialog1.Action = 4  
veya  
CommonDialog1.ShowFont
```

5: "Yazdır" diyalog kutusu

```
CommonDialog1.Action = 5  
veya  
CommonDialog1.ShowPrint
```

6: Windows Help programini çalıştırır.

```
CommonDialog1.Action = 6  
veya  
CommonDialog1.ShowHelp
```

Name : Diyalog penceresinin program içerisindeki adını belirler.

Cancel Error: Diyalog pencerelerinde "İptal" düğmesinin seçilmesi halinde yakalanabilir bir hata oluşturulup oluşturulmayacağı bu özellik ile belirlenir.

True : "İptal" düğmesi seçilirse hata oluşur.

False : Hata oluşturma.

ÖRNEK: Cancel Error özelliğini True yaparak kullanıcının İptal düğmesini seçmesi halinde yazacağımız kod aşağıdaki gibi olacaktır.

```
Private Sub Form_Click ( )  
    On Local Error GoTo iptal  
    CommonDialog1.Action = 1  
    CommonDialog1.CancelError = True  
    MsgBox ( CommonDialog1.filename & " doayası seçildi " )  
Exit Sub  
iptal:  
    MsgBox ( " Dosya seçme işlemi iptal edildi " )  
    Exit Sub  
End Sub
```

Dialog Title : Açılacak pencerelerin başlığında yazılması istenen metni belirler. Bir değer atanmazsa standart başlık yazısı görüntülenir.

Aç ve Yeniden Kaydet Diyalog Penceresi

FileName : Diyalog pencerelerinde **Dosya Adı** kutusunda yazılması istenen veya kullanıcı tarafından yazılan değerdir. Bu özelliğe, diyalog kutusu aktif hale getirilmeden bir değer atanması durumunda bu değer Dosya Adı kutusuna yazılacak default değer olur. Diyalog kutusu kapandıktan sonra ise, bu özelliğin aldığı değer kullanıcının seçtiği veya yazdığı dosya adıdır. Bu özelliğin ifade ettiği dosya adına sürücü ve yol dahildir.

CommonDialog1.Action=1 veya CommonDialog1.Action=2 satırları ile diyalog penceresini açtıktan sonra kullanıcının seçtiği dosyanın adını öğrenmek için bu özellik kullanılabilir.

FileTitle : Filename özelliğinden farklı olarak dosya adına yol dahil değildir. Sadece seçilen dosyanın ismi ve uzantısını gösterir.

DefaultExt : Kullanıcı dosya isminde dosya uzantısı yazmazsa bu özellik ile belirlenen uzanti dosya adına eklenir.

CommonDialog1.DefaultExt = "TXT" 'Default uzanti TXT

Diyalog penceresinde **Dosya Adı** kısmına kullanıcı DENEME yazarsa CommonDialog1.FileTitle = "DENEME.TXT" olacak, DENEME.X yazarsa CommonDialog1.FileTitle = "DENEME.X" olacaktır. Yani bir uzanti belirtmezse **DefaultExt** uzantisi ile belirlenen uzanti dosyaya eklenecektir.

Yaptığınız programın kullandığı standart bir dosya uzantısı varsa, bu özelliğe o uzantiyi vererek, her seferinde uzanti yazma zahmetinden kurtarabilirsiniz.

Filter : Aç ve farklı kaydet diyalog pencerelerinde de görüldüğü gibi



kutularına listelenecek dosyaların tiplerini seçmeye yarayacak açıklamalar koymaya yarar.

FilterIndex: Filtre özelliği ile listeye eklenen seçimlerden hangisinin default olarak etkin olacağını belirleyen bir numaradır. Listedeki ilk elemanın numarası 1'dir.

InitDir : Diyalog kutularının ilk açıldığında listeleyeceği dizini belirlemeyi sağlar. Verilmezse aktif dizin kabul edilir.

Flags : Açılacak diyalog penceresinin bazı özelliklerini belirleyen bir özelliktir. Bu özelliğin alacağı değerler ve etkileri aşağıda verilmiştir. Bu değerler OR işlemine tabi tutularak birden fazla özellik verilebileceği gibi hangi özelliğin seçili olduğu da aşağıdaki değerlerle AND işlemine tabi tutularak öğrenilir.

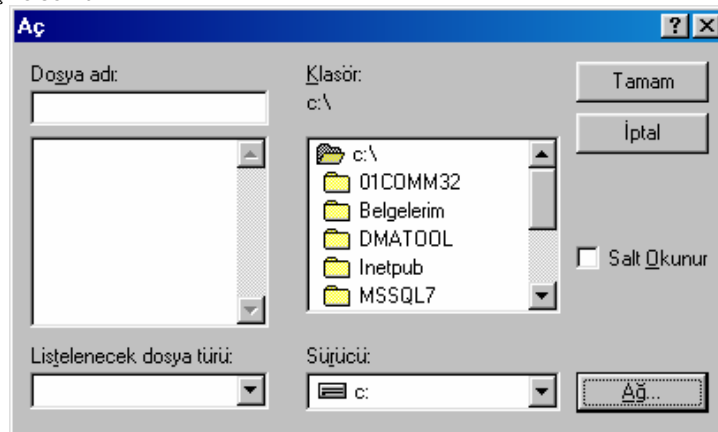
&H200, cdiOFNAllowMultiselect : Bu değer **Yeni Adla Kaydet** diyalog kutusunda bir etkisi yoktur. **Aç** diyalog kutusunda ise birden çok dosyanın seçilebilmesini sağlar. Bu seçme işlemi Shift veya Ctrl tuş kombinasyonlarıyla yapılır. Bu durumda FileName parametresi boş döner, **FileName** parametresinde ise, dosya isimleri arasında da boşluk olarak döner.

CommonDialog1.FileName = "Sürücüİsmi:Dizinİsmi\ Birinci Dosya İkinci Dosya Üçüncü Dosya..."

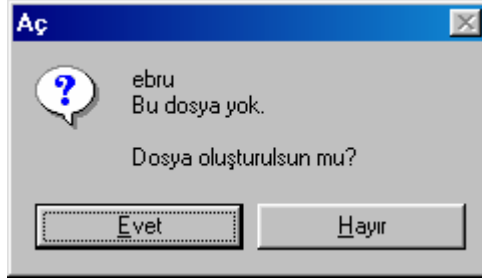
CommonDialog1.FileName = "C:\WINDOWS\ win.com msd.exe" şeklinde olacaktır.

Eğer bir tane dosya seçilmişse dizin ismi ile dosya ismi arasında boşluk bulunmaz.

Windows 95/98'de dosya isminde de boşluk olabileceği için Flags parametresine bu değer verildiğinde uzun dosya isimlerinin seçilebileceği dosya penceresi yerine 8 + 3 dosya isimlerinin seçilebileceği aşağıdaki dosya penceresi açılacaktır.

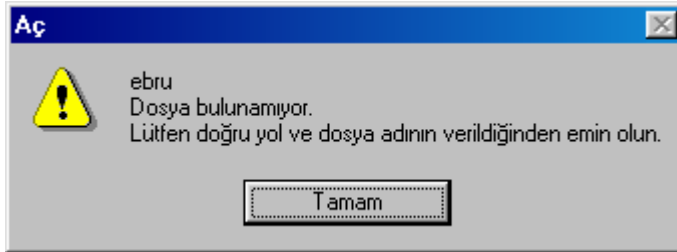


&H2000, cdIOFNCreatePrompt: Bu degerin de **Yeni Adla Kaydet** diyalog kutusunda bir etkisi yoktur. Aç diyalog kutusunda yazılan dosya isminin diskte olmaması durumunda Windows' un şöyle bir mesaj kutusu çıkarmasına sebep olur.

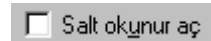


Burada "Hayır" seçilmesi halinde diyalog kutusu kapanmayacak ve yeni bir dosya seçmenizi sağlayacaktır. "evet" seçilmesi halinde ise gerçekte diskte olmayan dosya ismi geri gönderilir.

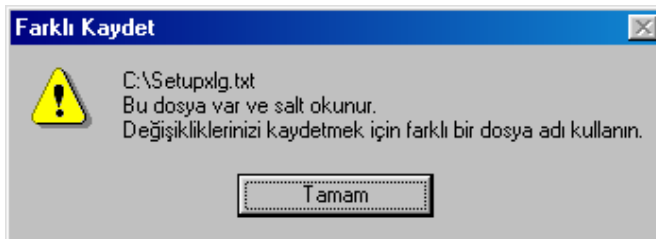
&H1000, cdIOFNExplorer: Bu degerin de Yeni Adla Kaydet diyalog kutusunda bir etkisi yoktur. Aç diyalog kutusunda yazılan dosya isminin diskte olmaması durumunda Windows' un şöyle bir mesaj kutusu çıkarmasına sebep olur.



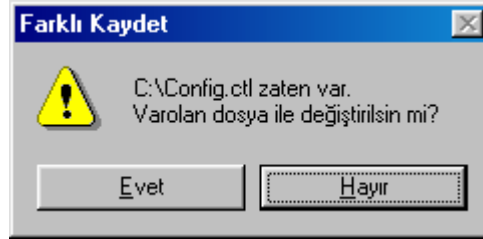
&H4, cdIOFNHideReadOnly: Diyalog kutusundaki Checkbox' inin görüntülenmemesini sağlar.



&H8000, cdIOFNNoReadOnlyReturn: Diyalog kutusunda Read Only (Sadece okunabilir) dosya seçilmesini önler. Böyle bir dosya seçilmesi durumunda Windows' un aşağıdaki mesajı görüntülemesini sağlar.



&H2, cdIOFNOverwritePrompt: Bu degerin **Aç** diyalog kutusunda bir degeri yoktur. **Yeni Adla Kaydet** diyalog kutusunda yazılan dosya ismiyle ayni isimli bir dosyanin diskte olmasi durumunda Windows' un asagidaki mesajla uyarimasini saglar.



&H800, cdIOFNPathMustExist: Girilen dosya isminin yolu dogru degilse Windows' un asagidaki gibi bir mesaj vermesini saglar.

&H1, cdIOFNReadOnly: Diyalog penceresi ☐ Salt okunur aç ☐ Salt okunur aç saglar. açilirken Checkbox' inini isaretlenmis olmasini saglar. Ayni zamanda kullanicinin bu seçenegi isaretleyip isaretlemedigi de Flags özelliginin bu degeri ile AND islemine tabi tutularak anlasilir.

&H4000, cdIOFNShareAware: Normalde diyalog kutulari, bir baska uygulama tarafından açık tutulan bir dosyanin seçilmesi durumunda bir hata mesajı ile kullanıcıyı uyarır. Eger flags özelligine bu deger verilirse diger programlar tarafından açık tutulan dosyaların da seçilebilmesini saglar.

&H10, cdIOFNHelpButton : Diyalog kutusunda **Yardım** komut düğmesinin görüntülenmesini saglar. Yardım düğmesinin etkili olması için uygulamanın Help dosyası ve HelpConTextID özelligi ile ilgili yardım konusunun belirlenmiş olması gerekir.

&H8, cdIOFNNoChangeDir: Normalde diyalog kutulari sonraki seferlerde açıldığında en son kalınan dizindeki dosyaları gösterir. Eger her seferinde ayni dizini göstermek istiyorsanız **Flags** parametresine bu deger verebilirsiniz.

Font diyalog Penceresi

Color: Diyalog penceresi içinde bulunan renk kutusunda seçilen rengin numarasıdır.

FontBold, FontItalic, FontStrikeThru, FontUnderline: True veya False degeri alabilen bu özellikler diyalog penceresinde Koyu, İtalik, Üstü Çizili ve Altıçizgili özelliklerinin durumunu belirler.

```
CommonDialog1.Flags = cdICFEffects or cdICFBoth  
CommonDialog1.ShowFont  
Text1.FontBold = CommonDialog1.FontBold  
Text1.FontItalic = CommonDialog1.FontItalic  
Text1.FontUnderline = CommonDialog1.FontUnderline
```

Fontname: Diyalog kutusunda seçilen/seçilecek fontun sistemdeki ismidir. Bu deger herhangi bir kontrolün font name özelligine atanabilir.

```
CommonDialog1.Flags = cdICFEffects Or cdICFBoth  
CommonDialog1.ShowFont  
Text1.FontName = CommonDialog1.FontName
```

FontSize: Seçilen fontun büyüklüğünü verir.

```
CommonDialog1.Flags =cdICFEffects Or cdICFBoth  
CommonDialog1.ShowFont  
Text1.FontSize =CommonDialog1.FontSize
```

Min,max: Diyalog penceresindeki boyut listesinde gösterilecek font boyutlarının minimum ve maximum degerlerini belirler.

Flags: Açılacak diyalog penceresinin bazı özelliklerini belirleyen bir deger alır. Alacağı degerler ve anlamları aşağıda verilmiştir. Bu degerler **OR** işlemine tabi tutularak birkaç özellik birden aktif hale getirilebilir.

&H100, cdICFEffects : Diyalog penceresinde etkiler ve renk kutularının da görüntülenmesini sağlar.

&H1, cdICFScreenFonts : Ekran fontlarının listede görülmesini sağlar.

&H2, cdICFPrinterFonts : Yazıcı fontlarının da listede görülmesini sağlar.

&H3, cdICFBoth : Hem ekran hem de yazıcı fontlarının listede görülmesini sağlar.

&H20000, cdICFScalableOnly : Yalnız ölçeklenebilir fontların listede görülmesini sağlar. (True Type, vektör fontlar vb.). Bu fontların özelliği büyütülüp/küçültüldüğünde şekillerinin bozulmamasıdır.

&H40000, cdICFTTOnly : Yalnız True type fontların listede görülmesini sağlar.

&H8000, cdICFWYSIWYG : Yalnız hem ekran hem de yazıcıda kullanılabilen fontların gösterilmesini sağlar. Bu flags değeri özellikle yazıcıdan alınacak çıktıyla ekranda görülen yazının aynı kalitede olması istendiği durumlarda kullanılmalıdır. Diğer durumda da ekranda görülen yazı yazıcıdan aynen çıkar ancak ekran ile yazıcının çözünürlükleri aynı olmadığı için yazım kalitesi düşük olacaktır.

Bu değerin etkili olması için &H1, &H2, ve &H20000 değerlerinin de verilmiş olması gerekir. Yani flags özelliğinin &H28003 olması gerekir.

&H400, cdICFANSIOnly : Symbol fontları gibi yazı içermeyen fontların listede görülmemesini sağlar.

&H800, cdICFNoVectorFonts : Vektör fontlarının seçilmemesini sağlar.

&H4000, cdICFFixedPitchOnly : Yalnız "Fixed Pitched" fontların görüntülenmesini sağlar.

Windows'ta kullanılan fontlardan çoğunluğunun (Variable Length Font-Times New Roman gibi) harf genişlikleri sabit değildir. "M" harfi "I" harfinden daha geniş bir alan kaplayacaktır. Fontların diğer bir kısmında ise (Fixed Pitched Fontlar – Courier New gibi) bu genişlik her harf için sabittir (DOS' ta olduğu gibi). Font genişliğinin sabit olması çirkin bir görüntü oluştururken özellikle tablolarda kullanılması verilerin aynı hizaya gelmelerini sağlayacaktır.

&H10000, cdICFForceFontExist : Kullanıcının listede olmayan bir font ismi yazması durumunda Windows' un aşağıdaki mesajı görüntülemesini sağlar ve geçerli bir font ismi seçilmesi için kontrolü tekrar diyalog kutusuna bırakır.

&H2000, cdICFLimitSize : Diyalog kontrolünün Max ve Min özellikleriyle belirlenen sınırlar haricinde bir font büyüklüğü seçilmesine izin vermez.

Renk diyalog penceresi

Windows tarafından sağlanan standart renk diyalog kurusunun kullanılmasını sağlayan bir kontroldür. Bu kontrolle sistem renklerinin veya kullanıcının tanımlayacağı özel renklerin kullanılması mümkündür.

CommonDialog kontrolünün **Action** özelliğine üç vererek veya ShowColor metodunu kullanarak renk diyalog penceresi aktif hale getirilir.

CommonDialog1.Action = 3
veya

CommonDialog1.ShowColor



Color : Kullanıcının seçtiği renk numarasıdır. Herhangi bir kontrolün BackColor, ForeColor gibi renk özelliklerinden birine atanarak seçili renk aktif hale getirilebilir.

Flags : Açılacak diyalog penceresinin bazı özelliklerini belirleyen bir değer alır. Alacağı değerler ve anlamları aşağıda verilmiştir. Bu değerler **OR** işlemine tabi tutularak birkaç özellik birden aktif hale getirilebilir.

&H2, cdICCIFullOpen : Diyalog penceresinin normalde sadece renk kısmı açılır. Kullanıcının **Özel Renk Tanımla** komut düğmesini tıklamasıyla da pencerenin diğer yarısı açılır. Eğer pencerenin tamamının program tarafından açılmasını istiyorsanız **Flags** özelliğinin bu değerini kullanmalısınız.

&H4, cdICCPreventFullOpen : Diyalog penceresinde Özel Renk tanımla komut düğmesinin pasif yapılmasını ve kullanıcı tarafından seçilememesini sağlar.

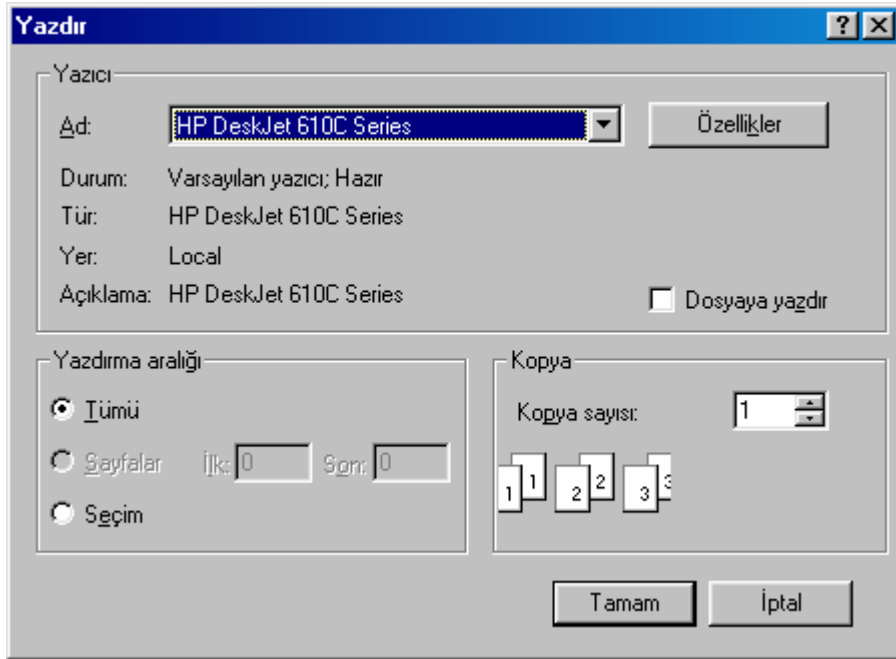
Yazdırma Diyalog Penceresi

Windows tarafından sağlanan standart **Yazdır** diyalog kutusunun kullanılmasını sağlayan bir kontroldür. Bu kontrolle; baskı kalitesini, basılacak

sayfa araligini, kopya sayisini ayarlamak ve **Yazici Ayarlari** diyalog kutusunu kullanmak mümkündür.

CommonDialog kontrolünün Action özelliğine 5 vererek veya ShowPrint metodunu kullanarak Yazdir diyalog penceresi aktif hale getirilir.

```
CommonDialog1.Action = 5  
veya  
CommonDialog1.ShowPrinter
```



PrinterDefault : Bu özelliğe **True** degeri verilirse kullanıcının seçtiği yazıcı ayarlari varsayılan ayar olarak kaydedilir.

Copies : Diyalog penceresinde **kopya sayisi** kutusuna yazılan degeri gösterir.

FromPage, ToPage : Diyalog kutusundaki **İlk** ve **Son** sayfa numaralarını belirten sayıları gösterir. Bu degerlerin -1 olması bütün sayfalar anlamına gelir.

Min, Max : Diyalog kutusundaki **Ilk** ve **Son** text kutularina girilebilecek minimum ve maximum siniri belirler. Kullanıcının bu sinir disinda bir deger girmesi halinde Windows asagidaki gibi bir mesaj görüntüler.

Flags : Açılacak diyalog penceresinin bazi özelliklerini belirleyen bir deger alır. Alacagi degerler ve anlamlari asagida verilmistir. Bu degerler OR islemine tabi tutularak birkaç özellik birden aktif hale getirilebilir.

&H0, cdIPDAIIPages : Yazdir diyalog penceresinde Tümü seçeneginin seçili olmasini saglar veya tümü seçeneginin seçili olup olmadigi bu degerle öğrenilir.

&H1, cdIPDSelection : Yazdir diyalog penceresinde Seçili Kisim seçeneginin seçili oldugunu gösterir.

&H2, cdIPDPPageNums : Yazdir diyalog penceresinde Sayfalar seçeneginin seçili oldugunu gösterir. Hangi sayfaların girildigi ise Max ve Min özellikleri ile öğrenilir.

&H20, cdIPDPrintToFile : Yazdir diyalog penceresinde Dosyaya Yazdir seçeneginin seçili oldugunu gösterir.

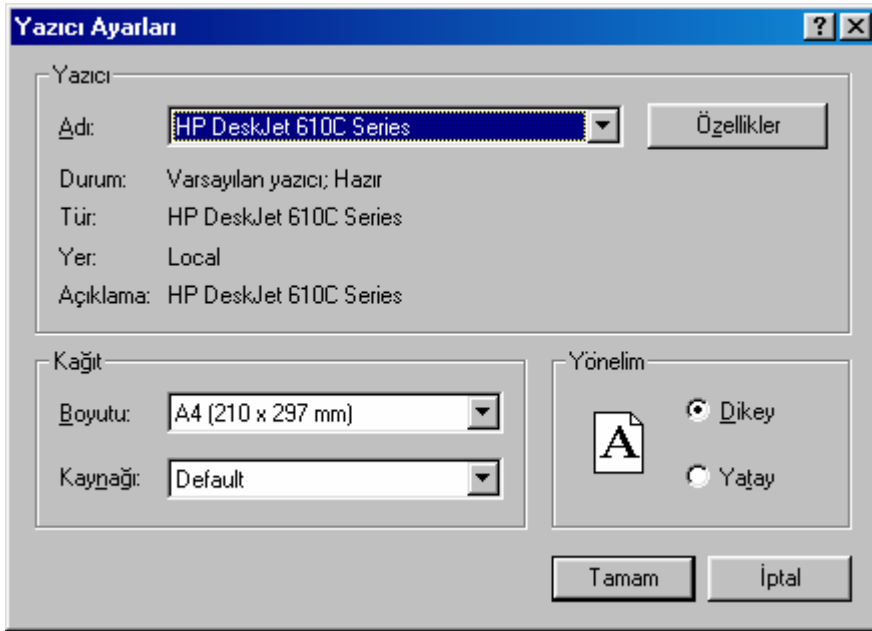
&H80000, cdIPDDisablePrintToFile : Yazdir diyalog penceresinde **Dosyaya Yazdir** seçeneginin pasif (seçilemez) olmasini saglar.

&H100000, cdIPDHidePrintToFile : Yazdir diyalog penceresinde **Dosyaya Yazdir** seçeneginin görüntülenmemesini saglar.

&H8, cdIPDNopageNums : Yazdir diyalog penceresinde **Sayfalar** seçeneginin, **Ilk** ve **Son** text kutularinin pasif olmasini saglar.

&H4, cdIPDNoSelection : Yazdir diyalog penceresinde **Seçili Kisim** seçeneginin pasif olmasini saglar.

&H40, cdIPDPrintSetup : Yazdir diyalog penceresi yerine **Yazici Ayarlari** diyalog penceresinin görüntülenmesini saglar.



Bu pencere aracılığı ile bir ayar yapıldıktan sonra bu ayarların etkili olabilmesi için **Printer.EndDoc** komutu kullanılmalıdır.

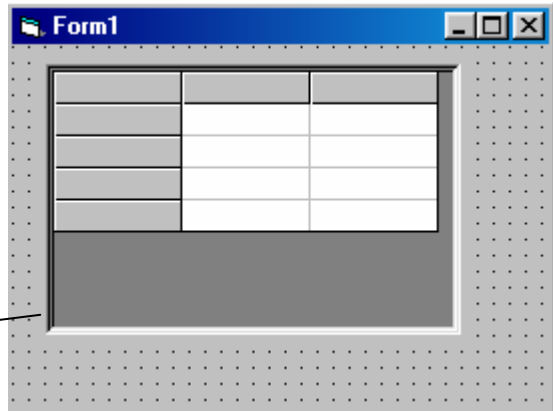
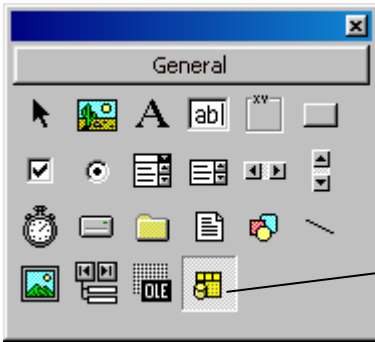
```
CommonDialog1.Flags = &H40  
CommonDialog1.Action = 5  
Printer.EndDoc
```

C-Microsoft MSFlexGrid (Izgara Kontrolü)

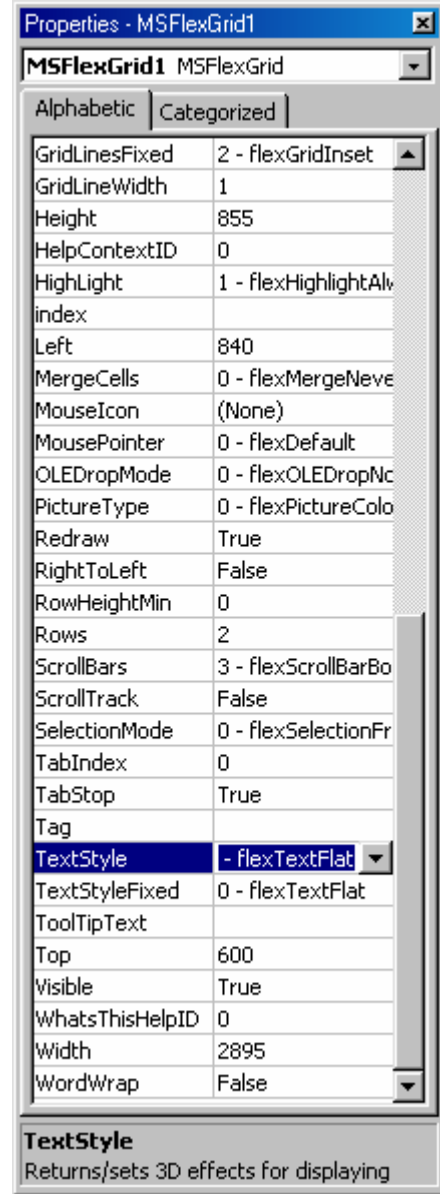
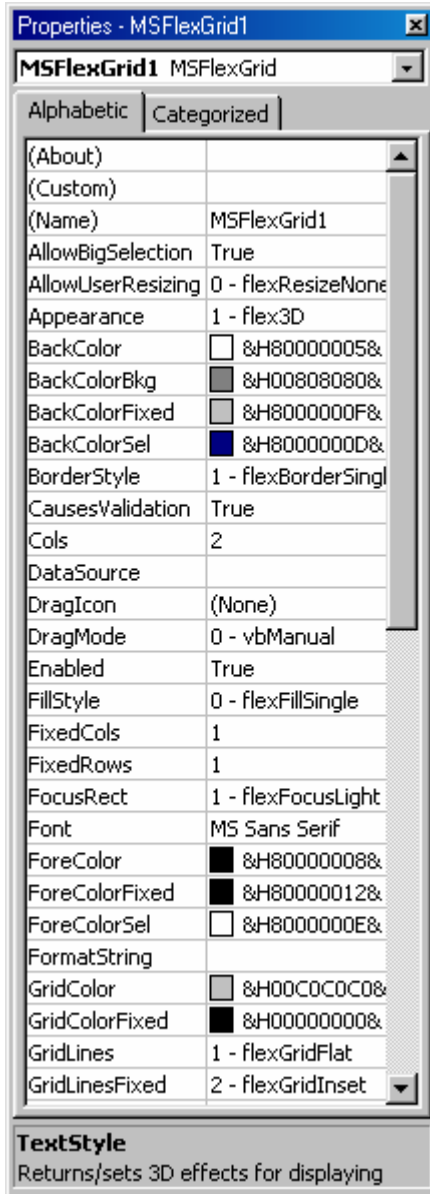
Grid kontrolü gibidir ancak bazı gelişmiş özelliklere sahiptir. Aşağıdaki özellikleri gride olduğu gibi kullanılır.

1. Cols ve Rows özellikleri ile satır ve sütun sayısı belirlenir.
2. Fixedcols ve FixedRows özellikleri ile başlık olarak kullanılacak satır ve sütunlar belirlenir.
3. Col ve Row özellikleri ile aktif kolon öğrenilebilir veya değiştirilebilir.
4. Text özelliği ile aktif hücrenin içeriği öğrenilebilir ve değiştirilebilir.
5. Clip özelliği ile seçili bölgenin içeriği öğrenilebilir ve değiştirilebilir.
6. ColWidth ve RowHeight özellikleri ile sütun genişliği ve satır yüksekliği belirlenebilir.
7. ColAlignment özelliği ile kolonların alignment'leri değiştirilebilir.

Toolbox



MSFlexGrid Properties (MSFlexGrid Özellikleri)



About : MSFlexGrid telifi hakkında bilgi verir.

Custom : MSFlexGrid kontrolünün özellikleri pencereler halinde bölümlere ayrılmıştır. Buradan özellikler sanki kısa yolmuş gibi ayarlanabilir.

Name : Kontrolün programda kullanılan ismidir.

AllowBigselecion : Bir sütun ya da satirin basına tiklandığında tiklanan satir ya da sütun boyunun tamamının seçilip seçilmemesine izin verir.

AllowUserResizing : Kullanıcının fare yardımıyla satir ya da sütunların tekrar büyütölüp büyütölmemesine seçenekler sunar.

Appearance : Kontrolün formdaki görüntüsünü belirler.

BackColor : Kontrolün satir ve sütunlarının rengini belirler.

BackColorBkg : Kontrolün üzerinde bulunduğu arka plan rengini belirler.

BackColorFixed : Kontroldeki satir ve sütun başlıklarının rengini belirler.

BackColorSel : Satir ve sütunların seçildiğinde alacağı rengi belirler.

BorderStyle : Kenar çizgilerini belirler.

CausesValidation : Kontrol üzerinde en son odaklanan noktanın geçerli olup olamayacağını belirler.

Cols : Kontroldeki sütun sayısını belirlemeye yarar.

DataSource : Veritabanındaki bağlanacak tabloyu belirler.

DragIcon : Sürükleme – bırakma olayı başladığında farenin alacağı şekli belirler.

DragMode : Fareyi sürükle–bırak olayının manual ya da otomatik olacağını belirler.

Enabled : kontrol üzerinde fare ile tiklanıp tiklanamayacağını belirler. True ise click olayı çalışır.

FillStyle : Şekilleri doldurma biçimi.

FixedCols : Kontrol üzerinde sabit sütunların sayısını belirler.

FixedRows : Kontrol üzerinde sabit satırların sayısını belirler.

FocusRect : Flex Grid kontrolü üzerinde, hücre üzerine tiklandığında hücrenin etrafında oluşacak çerçeve seçeneklerini belirler.

Font : Yazı tipini belirler.

ForeColor : Açıklama yazısının rengini belirler.

ForeColorFixed : Flexgrid' in köşe başlığının rengini belirler.

ForeColorSel : Hücreler içindeki yazının rengini belirler.

FormatString : FlexGrid' in köşe başlığı yazısını belirler.

GridColor : FlexGrid kontrolünde hücreler arasındaki çizgilerin rengini belirler.

GridcolorFixed : FlexGrid kontrolünde en son satır ve en son sütun çizgilerinin rengini belirler.

GridLines : Hücreleri ayıran çizgi tipini belirleyen seçenekler sunar.

GridLinesFixed : Fixed kısmın çizgi stilini belirleyen seçenekler sunar.

GridLinesWidth : Grid çizgisinin kalınlığını belirler.

Height : Nesnenin yüksekliğini belirler.

HighLight : Konumlanılan hücrenin belirgin olup olmaması ile ilgili seçenekler sunar.

Index : Nesne indeksli kullanılıyorsa indeksini belirler.

Left : Nesnenin sol hizasını belirler.

MergeCells : İstenirse Grid' de, yan yana veya üst üste aynı bilgileri içeren hücrelerin birleştirilerek tek bir hücre halinde gösterilmesi sağlanabilir. Bu özelliğin değeri normalde 0 dir ve herhangi bir birleştirme işlemi uygulanmaz. Birleştirme isteniyorsa bu özelliğe bir değeri verilebilir.

Bu özelliğe 1 değeri verildikten sonra hangi satır veya sütunlarda birleştirme uygulanacağı MergeRow veya MergeCol özellikleriyle belirlenir.

MouseIcon : Fare nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer : Fare nesnenin üzerindeyken hangi şekli alacağını belirler.

PictureType : Resim tipi ile ilgili özellikler belirlenir.

Redraw : Tekrar çizimin aktif ya da pasif olması belirlenir.

RowHeightMin : Bütün kontroller için satirin minimum yüksekliğini belirler.

Rows : Satir sayisini belirler.

ScrollBars : Kaydırma çubuklari ile ilgili seçenekler sunar.

TabIndex : TAB tusuna basildiginda kaçinci tabda kendine ulasilacagini belirler.

TabStop : TAB ile seçilmesi sonlandırılır.

Tag : Herhangi bir veriyi saklamak için kullanılır.

TextStyle : Yazıları yazma biçimi.

ToolTipText : Fare nesnenin üzerindeyken olusacak mesajı belirler.

Top : Nesnenin üstten hizasını belirler.

Visible : Form aktifken nesnenin görünür olu olmayacağını belirler.
True = görünür False = görülmez

Width : Nesnenin genişliğini belirler.

WhatThisHelpID : Bağlam numarasinin birlestirilmesini belirler.

MSFlexGrid Metodlari (Methods)

Bu nesnenin bütün özellikleri metod olarak da kullanılır. Özelliklerden farklı olan metodlar ve kod sayfasında kullanım şekilleri aşağıda verilmistir.

AddItem : Grid' e yeni bir satir eklemek için kullanılır.

MSFlexGrid1.AddItem

CellAlignment : Aktif hücrenin yatay ve dikey yerlesimi (sola, üste, ortaya vb.) bu metodla belirlenebilir.

MSFlexGrid1.CellAlignment(1) = 0

CellBackColor : Hücrenin arka fon rengini belirleme metodudur.

MSFlexGrid1.CellBackColor = &HFF00&

CellFontBold : Hücre içindeki yazının kalın olup olmasini belirler.

MSFlexGrid1.CellFontBold = True / False

CellFontItalic : Hücre içindeki yazının italic olup olmasini belirler.

MSFlexGrid1.CellFontItalic = False

CellFontName : Hücre içindeki yazının tipini belirler.

MSFlexGrid1.CellFontName = Verdana

CellFontSize : Hücre içindeki yazının boyutunu belirler.

MSFlexGrid1.CellFontSize = 10

CellFontStrikeThrough : Hücre içindeki yazının üstü çizili olup olmayacagini belirler.

MSFlexGrid1.CellFontStrikeThrough = True / False

CellFontUnderline : Hücre içindeki yazının altı çizili olup olmayacagini belirler.

MSFlexGrid1.CellFontUnderline = True / False

CellFontWidth : Hücre içindeki yazının genisligini belirler.

MSFlexGrid1.CellFontWidth = 15

CellForeColor : Hücre içindeki yazının rengini belirler.

MSFlexGrid1.CellForeColor = &HHff00&

CellHeight : Hücrenin yüksekligini belirler.

MSFlexGrid1.CellHeight = Verdana

CellLeft : Hücrenin solundan yazmaya baslar.

MSFlexGrid1.CellLeft

CellPicture : Hücre içine resim eklemek için kullanılan metoddur.

Microsoft Visual Basic 6.0

MSFlexGrid1.CellPicture= Load Picture (" ")

CellPictureAlignment : Hücrenin içindeki resmin sağ, sol, üst vb. yerini belirler.

MSFlexGrid1.PictureAlignment = justify

CellTextStyle : Hücrenin içindeki yazının görünme stilini belirler.

MSFlexGrid1.CellTextStyle = flexTextFlat

CellTop : Hücre içinde üste yaz.

MSFlexGrid1.CellTop

CellWidth : Hücrenin genişliğini belirler.

MSFlexGrid1.CellWidth

CellClear : Hücrenin içeriğini temizler.

MSFlexGrid1.Clear

CellClip : Bir ızgara içindeki seçilmiş olan bölgenin içeriğini verir. Ayrıca birçok hücreye birden atama imkanı verir. Aynı satırdaki hücreler arasında chr(13) karakteri, bir sonraki sütuna geçiş içinde Chr(9) karakteri bulunur.

MSFlexGrid1.Clip = "1,1. hücre" + Chr(9) + "1,2. hücre" + Chr(13) + "2,1. hücre"

CellColVisible : Sütunun görünürlüğünü belirler.

MSFlexGrid1.ColVisible = True / False

CellColWidth : Sütunun genişliğini belirler.

MSFlexGrid1.ColWidth

FixedAlignment: Fixed olarak adlandırılan gri kolonların alignmentlarını ayarlar.

MSFlexGrid1.fixedAlignment =

hWnd : Windows altında çalışan kontrollerin handle diye adlandırılan tanıtıcı bir numarası vardır. Windows bu numarayı kullanarak kontrolleri tanır. Visual

Basic' ta bu numara nesnenin hWnd özelliği ile programın çalışması esnasında öğrenilebilir. hWnd özelliği genellikle Windows API çağrılarında gereklidir.

MergeCol : İstenilen sütunların birleştirme işlemi yapılır.

MSFlexGrid1.MergeCol (1) = True

MergeRow : İstenilen satırların birleştirme işlemi yapılır.

MSFlexGrid1. MergeRow (1) = True

Picture : FlexGrid' in görüntüsünü resim olarak belirlemeye yarar.

MSflexGrid1.Picture = LoadPicture (" ")

Refresh : Yenileme - güncelleme

MSFlexGrid1.Refresh

RemoveItem : Index numarası verilen satırı siler ve yukarıya doğru kaydırır.

MSFlexGrid1.RemoveItem 1

Row : Satır sayısını belirler.

RowHeight : Satır yüksekliğini belirlemeye yarar.

RowVisible : İstenilen satırın görünürliğini belirler.

SetFocus : Odaklanma noktası.

ShowWhatsThis : Bu nedir simgesini ekler.

Sort : Aşağıdaki seçeneklerden biri seçilerek Grid içindeki sütunlardan birine göre sıralanabilir. Eğer seçili bir alan varsa sıralama işlemi sadece o alanda uygulanır. Seçili alan yoksa aktif sütuna göre sıralama yapılır.

flexSortNone	0	Sıralama yok
flexSortGenericAscending	1	Artan Sıralama
flexSortGenericDescending	2	Azalan Sıralama
flexSortNumericAscending	3	Artan Sayısal Sıralama
flexSortNumericDescending	4	Azalan Sayısal Sıralama
flexSortstringNoCaseAscending	5	Büyük küçük harf ayrımı yapmayan, Artan Sıralama

flexSortNoCseDescending	6	Büyük küçük harf ayrimi yapmayan, Artan Siralama
flexSortStringAscending	7	Büyük küçük harf ayirimi yapan, Artan Siralama
flexSortStringDescending	8	Büyük küçük harf ayirimi yapan, Azalan Siralama
flexSortCustom	9	Özel siralama. Siralamanin nasıl olacağı Compare olayına yazılacak kodla yapılır.

Text : Satir ve sütunların içine yazı yazmaya yarar.

TextArray : Yazıları düzenlemeye yarar.

TextMatrix : Satir ve sütun parametreleri ile belirlenen hücredeki bilgi bu özel-likle öğrenilebilir veya değiştirilebilir.

MSFlexGrid1.TextMatrix (1,1) = "Visual Basic"

Zorder : Grafik düzeyi belirler.

MSFlexGrid Events (Olaylar)

Click : Farenin sol tusuna bir kez tiklandığı an.

DBClick : Farenin sol tusunun iki kez tiklandığı an.

Dragdrop : Farenin sürüklenip bırakılması anı.

DragOver : Farenin nesne üzerinden geçerken oluşan an.

EnterCell : Bir hücrenin aktif hale geldiği an.

GotFocus : Nesnenin üzerine gelinip aktif olduğu an.

KeyDown : Klavyenin tusuna basıldığı an.

KeyPress : Klavyeden gelen herhangi bir tustan bir değer alındığında yapılan olay.

KeyUp : Key down olayının tam tersi.

LeaveCell : Hücrenin aktiviteyi kaybettiği an.

LostFocus : Nesne üzerinden ayrılma anı.

MouseDown : Herhangi bir olay üzerindeyken farenin tuslarına tiklandığı an.

MouseMove : Herhangi bir olay üzerindeyken farenin hareket ettirildiği an.

MouseUp : Mousedown olayının tam tersi .

OLECompleteDrag : Drag – Drop işlemi tamamlandığında veya iptal edildiğinde bu olay meydana gelir.

OLEGiveFeedBack : bu olay, OLE kaynağının sürüklenip bırak işlemi esnasında fare şeklinin değişmesi gerektiği durumlarda meydana gelir.

OLEStartDrag : Sürüklenme (drag) işlemi başladığında gerçekleşecek olan olaydır.

RowColChange : Aktif olan hücre kontrolünün başka bir hücreye geçmesi halinde yani o hücrenin aktif olması halinde gerçekleşen olaydır.

SelChange : Seçili alanın değişmesi halinde gerçekleşen olaydır.

Scroll : Bu olay hareket çubuğundaki (scrollbar) kutucuğu hareket ettirdiği zaman gerçekleşen olaydır.

Validate : Farklı bir kayıt aktif haline gelmeden önce , Update, Delete, Unload, Close metodlarından önce meydana gelen olaydır.

D-Microsoft Forms 2.0 object Library:

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden Microsoft Forms 2.0 object Library seçeneği seçilerek istedigimiz Component' i ekleriz.

MultiPage: Bilginin çok ekranlarini tek set olarak sunar. Çesitli siniflar içine siralanilabilen çok miktarda bilgi ile çalisdiginizda multipage kullanılabilir.

Name: Program içerisindeki ismini belirler.

Back color: Arka plan rengini belirler.

CausesValitation: Controlde odaklanmasini kaybetmis olan olusumlarini tekrar döndürme ya da kurma islemini yapar.

DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon Nesnesi geri döndürür.

DragIcon: Sürükleme olayi basladiginda farenin alacagi sekli belirler.

DragMode: Sürükleme olayinin otomatik veya manuel olacagini belirler.

Enable: Clicklenebiliriligini belirler. Tru ise click olayi çalisir.

Font: Yazı tipini belirler.

ForeColor: Açıklama yazisinin rengini belirler.

Height: Nesnenin yüksekligini belirler.

HelpContextID: Indexli kullanimda indexini belirler.

Index: Index numarasini belirler.

Left: Sola hizasini belirler.

Properties - MultiPage1	
MultiPage1 MultiPage	
Alphabetic Categorized	
(Name)	MultiPage1
BackColor	<input type="color"/> &H8000000F&
CausesValidation	True
DataBindings	
DragIcon	(None)
DragMode	0 - vbManual
Enabled	True
Font	MS Sans Serif
ForeColor	<input type="color"/> &H80000012&
Height	495
HelpContextID	0
Index	
Left	360
MultiRow	False
Style	0 - fmTabStyleTabs
TabFixedHeight	0
TabFixedWidth	0
TabIndex	0
TabOrientation	0 - fmTabOrientation
TabStop	True
Tag	
ToolTipText	
Top	1080
Value	0
Visible	True
WhatsThisHelpID	0
Width	735

Value

MultiRow: Kontrolün listede daha çok ön siraya sahip olduğunu belirler.

Style: Belirtilmiş listedeki karışık uzunluk ve genişliği belirler.

TabFixedHeight: Belirtilmiş listedeki uzunluğu belirtir.

TabFixedWidth: Belirtilmiş listedeki genişliği belirler.

TabIndex: TAB'a basıldığında kaçinci atlamada kendine ulaşacağını belirler.

TabOrientation: Listelerin yerini belirler.

Tabstop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

Multipage Fonksiyonları:

MultiPage1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

MultiPage1.Drag: Herhangi bir kontrolün sürükleme çalışmasını iptal eder çizgi menü şekli zama nlayıcı veya commondialog kontrol eder .

MultiPage1.Move: Mdi form form hareketini kontrol eder.

MultiPage1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

MultiPage1.Parent: Kontrol veya başka nesneyi kapsayan form nesneyi geri döndürür.

MultiPage1.SetFocus: Belirlenmiş kontrol veya forma thefocus hareket eder

MultiPage1.ShowWhatsThis: Pencere 95nin yanında yardım sağlanan bu popup ne oldugunun kullanarak yardım dosyasinda seçilmiş konuyu göster .

ToogleButton: Bilginini seçim durumunu gösterir. Bilginin seçildigini göstermek için togglebuttonu kullanilir. togglebutton veri kaynagini sinirlendirabisilir. Herhangi bir veri kaynaginın güncel degerini gösterir. Bir gurupla ilgili bilgilerin bir ya da birden çoğunu seçmek için yapı içinde togglebuttonu aynı zamanda kullanabiliriz .

Name: Program içerisindeki ismini belirler.

Accelerator: Kontrol için accelerator anahtarini kurar ya da tekrar geri döndürür.

AutoSize: Genisligini üzerinde bulunan yazıya ayarlar.

Back color: Arka plan rengini belirler.

BackStyle: Üzerinde bulunduğu nesnenin Gözüküp gözükmeyecegini belirtir.

Caption: Üzerinde görünecek yazıyı belirler.

CausesValitation: Controlde odaklanmasını kaybetmiş olan olusumlarını tekrar döndürme ya da kurma işlemini yapar.

DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon Nesnesi geri döndürür.

Properties - ToggleButton1	
ToggleButton1 ToggleButton	
Alphabetic Categorized	
(Name)	ToggleButton1
Accelerator	
AutoSize	False
BackColor	8H8000000F&
BackStyle	1 - fmBackStyleOpaque
Caption	
CausesValidation	True
DataBindings	
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - vbManual
Enabled	True
Font	MS Sans Serif
ForeColor	8H80000012&
Height	255
HelpContextID	0
Index	
Left	1440
Locked	False
MouseIcon	(None)
MousePointer	0 - fmMousePointerDefault
Picture	(None)
PicturePosition	7 - fmPicturePositionAboveCenter
TabIndex	1
TabStop	True
Tag	
TextAlign	2 - fmTextAlignCenter
ToolTipText	
Top	1920
TripleState	False
Value	False
Visible	True
WhatsThisHelpID	0
Width	495
WordWrap	True

DataField: Veritabanına bağlantı için alan seçilmesini sağlar.

DataFormat: Veritabanından alınacak bilginin formatını belirler.

DataMember: Data bağlantısı kurar ve bir değer belirtir.

DataSource: Veritabanının tablosunu belirler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manuel olacağını belirler.

Enable: Clicklenebilirliğini belirler. Tru ise click olayı çalışır.

Font: Yazı tipini belirler.

ForeColor: Açıklama yazısının rengini belirler.

Height: Nesnenin yüksekliğini belirler.

HelpContextID: Indexli kullanımda indexini belirler.

Index: Index numarasını belirler.

Left: Sola hizasını belirler

Locked: Veri girişine izin verilip verilmeyeceğini belirler.

MouseIcon: Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Mouse nesnenin üzerindeyken hangi şekli alacağını belirler.

Picture: Bir resme bağlanmayı sağlar.

PicturePosition: Belirlenen resimle arasındaki bağlantıyı belirler.

TabIndex: TAB'a basıldığında kaçinci atlamada kendine ulaşacağını belirler.

Tabstop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

TextAlign: Kontrole texttin nasıl bağlanacağını belirtir.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

TriState: The Null State için bir kontrol kutusunun ya da ToogleButtonun kullanıcı tarafından belirtilip belirtilmeyeceğini sağlar.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

WordWrap: İçerdiği bilgiyi forma kelime bölerek ya da bölmeden getirileceğini belirler.

ToggleButton Fonksiyonları:

ToggleButton1.Drag: Herhangi bir kontrolün sürükleme çalışmasını iptal eder çizgi menü şekil zamanlayıcı veya commdialog kontrol eder .

ToggleButton1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

ToggleButton1.Object: : Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

ToggleButton1.Parent: Kontrol veya başka nesneyi kapsayan form nesneyi geri döndürür.

ToggleButton1.SetFocus: Belirlenmiş kontrol veya forma thefocus hareket eder .

ToggleButton1.ShowWhatsThis: Pencere 95nin yanında yardım sağlanan bu popup ne olduğunun kullanarak yardım dosyasında seçilmiş konuyu göster .

Tabstrip: Görülebilir grup olarak bir takım ilgili kontrolleri sunar. İlgili kontroller için bilginin farklı setlerini görmek için tabstrip' i kullanabiliriz.

Name: Program içerisindeki ismini belirler.

Back color: Arka plan rengini belirler.

CausesValitation: Controlde odaklanmasını kaybetmiş olan oluşturmalarını tekrar döndürme ya da kurma işlemini yapar.

DataBindings: Geliştiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon nesnesi geri döndürür.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manuel olacağını belirler.

Enable: Clicklenebilirliğini belirler. True ise click olayı çalışır.

Font: Yazı tipini belirler.

ForeColor: Açıklama yazısının rengini belirler.

Height: Nesnenin yüksekliğini belirler.

HelpContextID: Indexli kullanımda indexini belirler.

Index: Index numarasını belirler.

Left: Sola hizasını belirler.

MouseIcon: Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.



MousePointer: Mouse nesnenin üzerindeyken hangi şekli alacağını belirler.

MultiRow: Kontrolün listede daha çok ön siraya sahip olduğunu belirler.

Style: Belirtilmiş listedeki karışık uzunluk ve genişliği belirler.

TabFixedHeight: Belirtilmiş listedeki uzunluğu belirtir.

TabFixedWidth: Belirtilmiş listedeki genişliği belirler.

TabIndex: TAB'a basıldığında kaçinci atlamada kendine ulaşacağını belirler.

TabOrientation: Listelerin yerini belirler.

TabStop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

TabStrip Fonksiyonları:

TabStrip1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

TabStrip1.Drag: : Herhangi bir kontrolün sürükleme çalışmasını iptal eder çizgi menü şekli zamanlayıcı veya commondialog kontrol eder .

TabStrip1.Move: Mdi form form hareketini kontrol eder.

TabStrip1.Object: : Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

TabStrip1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

TabStrip1.SetFocus: Belirlenmis kontrol veya forma thefocus hareket eder .

SpinButton: Artislar ve karar vermeleri sayar. Spinbuttonu tiklayarak sadece spinbuttonun degerini degisiririz. Baska kontrolun gösterilen degerini güncellestirmek için spinbuttonu kullanarak kod yazabiliriz. Örneğin siz tarihte gösterilen ay, gün veya yili degismek için spinbuttonu kullanabilirsiniz.

Name: Program içerisindeki ismini belirler.

Back color: Arka plan rengini belirler.

CausesValitation: Controlde odaklanmasini kaybetmis olan olusumlarini tekrar döndürme ya da kurma islemini yapar.

DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon Nesnesi geri döndürür.

DataField: Veritabanina baglanti için alan seçilmesini saglar.

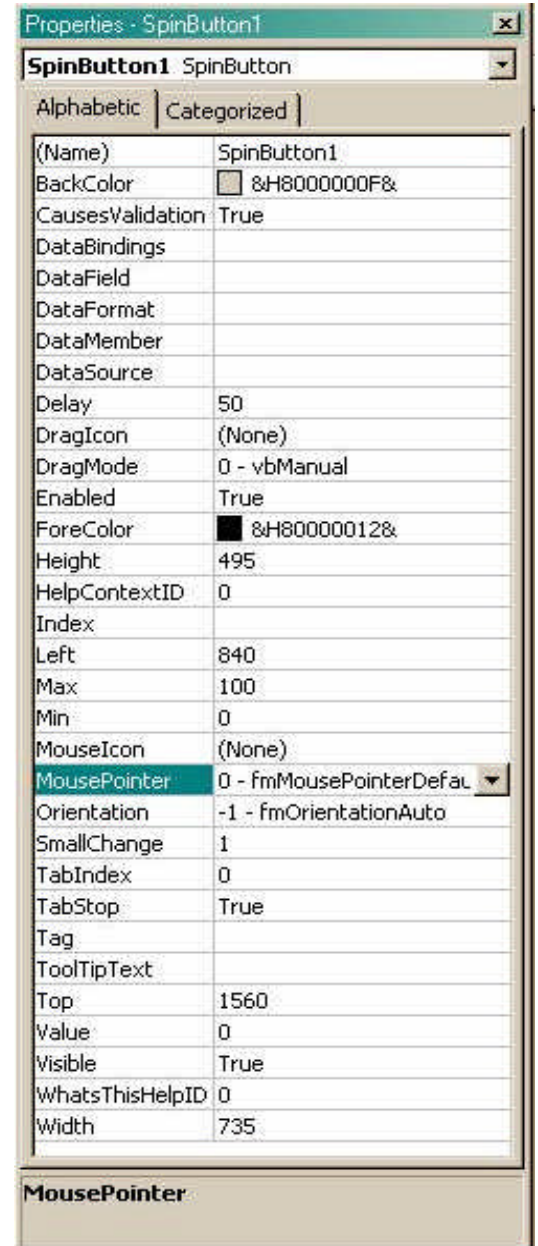
DataFormat: Veritabanından alınacak bilginin formatini belirler.

DataMember: Data baglantisi kurar ve bir deger belirtir.

DataSource: Veritabaninin tablosunu belirler.

Delay: Gecikmeyi belirler.

DragIcon: Sürükleme olayi basladiginda farenin alacagi sekli belirler.



DragMode: Sürükleme olayının otomatik veya manuel olacağını belirler.

Enable: Clicklenebilirliğini belirler. Tru ise click olayı çalışır.

ForeColor: Açıklama yazısının rengini belirler.

Height: Nesnenin yüksekliğini belirler.

HelpContextID: Indexli kullanımda indexini belirler.

Index: Index numarasını belirler.

Left: Sola hizasını belirler.

Max: ScrollBar ya da SpinButton değeri için max değeri belirler.

Min: ScrollBar ya da SpinButton değeri için min değeri belirler.

MouseIcon: Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Mouse nesnenin üzerindeyken hangi şekli alacağını belirler.

Oriantation: SpinButton ya da ScrollBar!ın karşıdaki bilgi ile ilgili olup olmadığını belirler.

Smallchange: ScrollBar ya da SpinButtondaki belgelerin kullanıcı tarafından girildiğinde oluşan hareketlerin oranını belirler.

TabIndex: TAB'a basıldığında kaçinci atlamada kendine ulaşacağını belirler.

Tabstop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

SpinButton Fonksiyonlari:

SpinButton1.Container: Formda kontrole ait mevcut atdesign zamanlamasi yapmaz.

SpinButton1.DataChanged: Thebound kontrolunde veri ondan baska bazi islemin tarafından degistirildigi degeri gösterir.

SpinButton1.Move: Mdiform form hareketini kontrol eder.

SpinButton1.Object: : Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

SpinButton1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

SpinButton1.SetFocus: Belirlenmis kontrol veya forma thefocus hareketi yapar.

E-Microsoft Hrerarchikal FilexGrid Control :

Project menüsünden Components (Ctrl+T) seçenegi seçilir. Gelen pencereden seçenegi Microsoft Hrerarchikal FilexGrid Control seçilerek istedigimiz Companent' i ekleriz.

MshflexGrid: Microsoft hiyerarsik flexgrid mshflexgrid gösterimleri kontrol eder ve tabular verisini isletir. Mshflexgrid' in herhangi bir hücresinde metin resim veya her ikisini yerlestirebiliriz.

Name: : Program içerisindeki ismini belirler.

AllowbigSelection: Kolon veya sıra basliginda, seçilmek için bütün kolon veya sirayi belirler.

AllowUserResizing: Kullanici, mshflexgrid' ta resize siralari ve kolonlarda fareyi kullanabilir .

Appearance: Formdaki görüntüsünü belirler.

Back color: Arka plan rengini belirler.

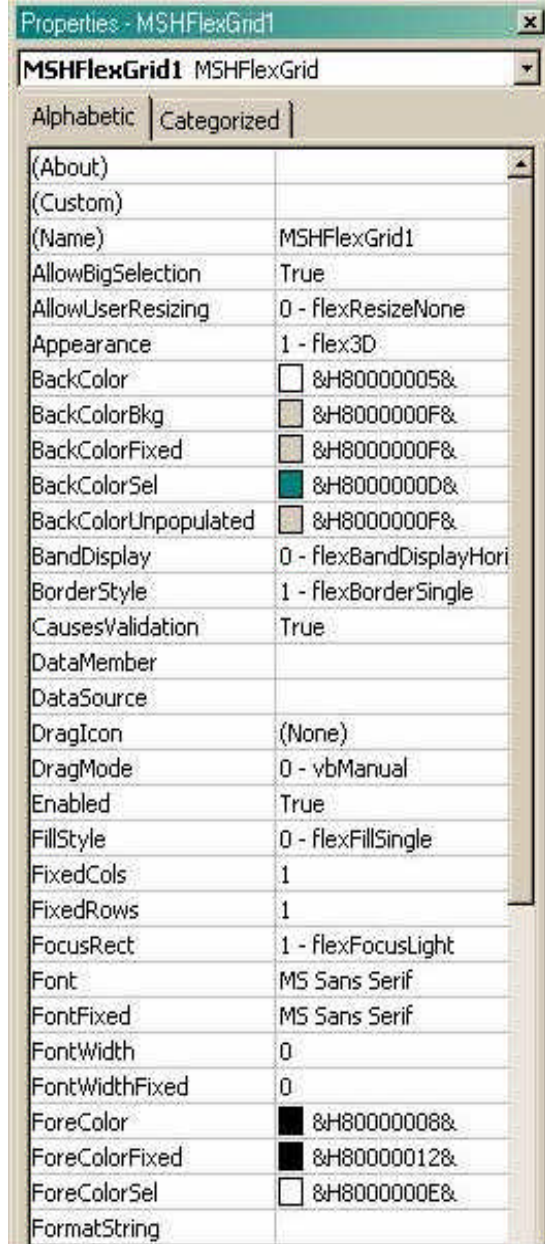
BackColorBkg: Mshflexgrid'in çeşitli elemanlarinin zemin rengini belirler.

BackColorFixed: Mshflexgrid'in çeşitli elemanlarinin zemin rengini belirler.

BackColorSel: Mshflexgrid'in çeşitli elemanlarinin zemin rengini belirler.

BackColorUnpoulated: Mshflexgri d'in çeşitli elemanlarinin zeminrengini belirler.

BandDisplay: Mshflexgrid içindeki batlari gösterdigini belirtir.



BorderStyle: Kenar çizgilerini belirler.

CausesValitation: Controlde odaklanmasını kaybetmiş olan oluşumlarını tekrar döndürme ya da kurma işlemini yapar.

DataMember: Data bağlantısı kurar ve bir değer belirtir.

DataSource: Veritabanının tablosunu belirler.

DataField: Veritabanına bağlantı için alan seçilmesini sağlar.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manuel olacağını belirler.

Enable: Clicklenebilirliğini belirler. Tru ise click olayı çalışır.

FillStyle: Şekilleri doldurma biçimini belirler.

FixedCols: Mshflexgrid içinde sabit kolonların toplam sayısını belirtir.

FixedRows: Mshflexgrid içinde sabit sıraların toplam sayısı belirtir.

FocusRect: Mshflexgrid güncel hücre etrafında odak dikdörtgeni çizmiş olması gerektiğini belirleyen değeri belirtir.

Font: Yazı tipini belirler.

FontFixed: Sabit hücrelerde metnin yazı tipini belirler.

GridColor	&H80000012&
GridColorFixed	&H00C0C0C0&
GridColorUnpopulated	&H00C0C0C0&
GridLines	1 - flexGridFlat
GridLinesFixed	2 - flexGridInset
GridLinesUnpopulated	0 - flexGridNone
GridLineWidth	1
GridLineWidthFixed	1
GridLineWidthUnpopulated	1
Height	735
HelpContextID	0
HighLight	1 - flexHighlightAlways
Index	
Left	600
MergeCells	0 - flexMergeNever
MouseIcon	(None)
MousePointer	0 - flexDefault
OLEDropMode	0 - flexOLEDropNone
PictureType	0 - flexPictureColor
Redraw	True
RowHeightMin	0
Rows	2
RowSizingMode	0 - flexRowSizeIndividu
ScrollBars	3 - flexScrollBarBoth
ScrollTrack	False
SelectionMode	0 - flexSelectionFree
TabIndex	0
TabStop	True
Tag	
TextStyle	0 - flexTextFlat
TextStyleFixed	0 - flexTextFlat
ToolTipText	
Top	600
Visible	True
WhatsThisHelpID	0
Width	1575
WordWrap	False

MousePointer

Returns or sets the type of mouse pointer displayed when over part of an object.

FontWidth: Mshflexgrid veya içinde gösterilen metin için kullanılan metnin genişliğini belirler.

FontWidthFixed: Fontwidth , fontwidthband , fontwidthfixed ve ontwidthheader özellikleri için metin dizim kuralları bu parçada belirlenir.

ForeColor: Açıklama yazısının rengini belirler.

FormatString: Metni sabitleştirilen kolon genişlikleri , siraya koyar ve mshflexgridin kolon metnini sabitleştirir .

GridColor: Hücreler arasında çizgi, bant ve başlıklar arasındaki rengini belirler.

GridLines: Çizilen çizgilerin kenar rengini belirler.

Highlight: Seçilmiş hücreler mshflexgrid içinde önemli olayları belirtir.

MergeCells: Belirtilen hücreleri belirleyen değer aynı içerik , sıraları veya kolonları kapsayan tek hücrede gruplaşmış değeri belirtir.

OLEDropMode: Hedef parçası çalışmalarını ele aldığını belirtir.

PictureType: Resim özelliği yoluyla üretilmek için resmin tipini belirler.

Redraw: Mshflexgridi belirleyen değer herbirisi değişdikten sonra mshflexgrid , otomatik olarak redrawn olmasını sağlar.

RowHeightmin: Bütün kontrol , intwips için minimum sıra yüksekliğini belirler.

Row: Satır ekleme sayısını belirler.

RowSizingMode: Mshflexgrid'ta sıraların size modunu tarif eden değeri belirler.

ScrollBars: Mshflexgrid' in yatay veya dikey barlarına sahip olduğunu belirleyen değeri belirtir.

ScrollTrack: Mshflexgrid kullanırken , barlar boyunca bar kutusunu hareket ettirir.

SelectionMode: Mshflexgrid'i kolonlar yoluyla sıralar veya seçim yoluyla düzenli hücre seçimini belirler.

TabIndex: TAB'a basildiginda kaçinci atlamada kendine ulasacagini belirler.

Tabstop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

TextStyle: Özel hücre veya hücrelerin içinde metin için üç boyutlu biçim belirler.

TextStyleFixed: Özel hücre veya hücrelerin içinde metin için üç boyutlu biçim belirler.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

WordWrap: İçerdiği bilgiyi forma kelime bölerek ya da bölmeden getirileceğini belirler.

MSHFlexGrid Fonksiyonları:

MSHFlexGrid1.BackColorBand: Mshflexgrid'in çeşitli elementlerinin geri rengini belirler.

MSHFlexGrid1.BandColIndex: Hücreyi kapsayan bant ile ilgili güncel hücrenin kolon sayısını belirler.

MSHFlexGrid1.BandLevel: Güncel hücreyi kapsayan bant sayısını belirler. Bant sayıları 0' da başlarlar.

MSHFlexGrid1.Bands: Mshflexgrid' ta bantların toplam sayısı belirler.

MSHFlexGrid1.CellAlignment: Güncel hücre içinde verinin yatay ve dikey sıraya konmasını belirleyen değerdir.

MSHFlexGrid1.CellFontBold: Güncel hücre metni için biçim özelliklerini belirler.

MSHFlexGrid1.CellPicture: Güncel hücre veya hücrelerde gösterilmek için görüntü özelliklerini belirler.

MSHFlexGrid1.Clear: Mshflexgrid' in içeriğini temizler. Bu bütün metin resimler ve hücre formatını içerir. Bu metod mshflexgrid içinde sıralar ve kolonların sayısını etkilemez.

MSHFlexGrid1.Clip: Mshflexgrid' in seçilmiş bölgesinde hücrelerin içeriği bu özellik ile belirlenir.

MSHFlexGrid1.ColAlignment: Kolonda verinin siraya konmasını sağlar. Bu başlık içinde bant veya kolon içinde standart kolon olabilir.

MSHFlexGrid1.ColData: Uzun ve her birisi sıralı kolon ile ilgili değerlendirmeleri veya bantları belirler.

MSHFlexGrid1.CollapseAll: Mshflexgrid içinde bütün belirlenmiş bantların sıralarını belirler.

MSHFlexGrid1.Sort: Seçilmiş kritere göre, seçilmiş sıraları sıralayan değeri belirler.

MSHFlexGrid1.MergeCells: Aynı içerikli hücreler çok sayıda sırayı veya kolonu kapsayan tek hücrede gruplama olması gerektiğini belirleyen değeri belirler.

MSHFlexGrid1.Recordset: Recordset veri kontrolü özellikleri yoluyla tanımlı değeri setler.

F-Microsoft Internet Transfer Control :

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden Microsoft Internet Transfer Control : seçeneği seçilerek istediğimiz Component' i ekleriz.

Inet : Internet transfer kontrolü, internet hypertext transfer protokol http ve dosya transfer protokol ftp de çok genişçe kullanılan protokollerin ikisinde uyarlamasını sağlar. Http protokolünü kullanarak html dokümanını geri alabilir, dünya çapında ağ hizmetçilerine bağlayabiliriz. Ftp protokolu ile, download' a ftp hizmetçilerine bağlanabiliriz.

Name : Program içerisindeki ismini belirler.

Accesstype: Kontrol internet ile haberleşmek kullanacağını erişimin tipini belirleyen değerdir .Eszamanlı olmayan talep , işleme tabi tutulur iken bu değer , değiştirilebilir ama , etki almayacak sonraki bağlantı , kurulana kadar .gçerlidir.

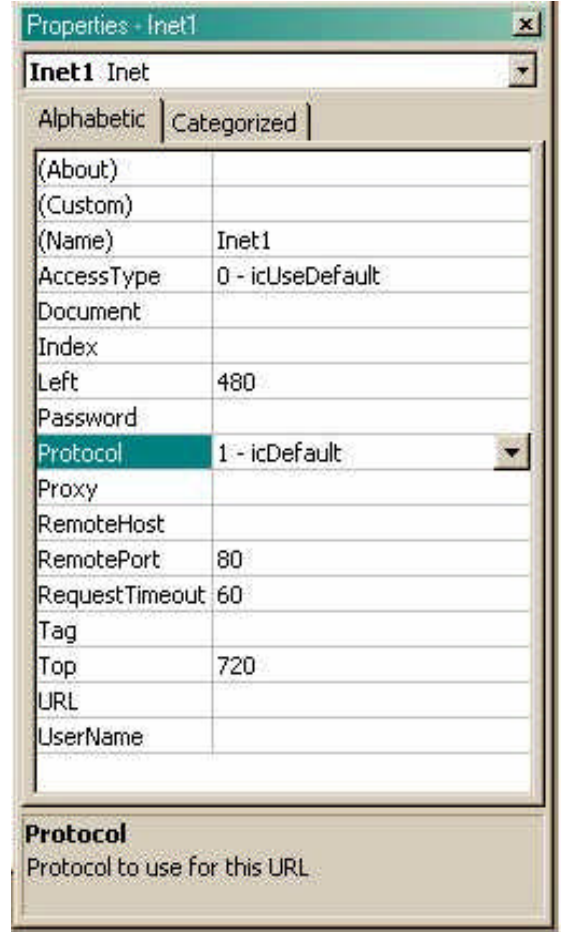
Document: Kullanılan dosya veya doküman , metodunu işletir. eğer bu özellik , belirtilmezse , hizmetçiden ön değer doküman ,istenmez . eğer hiçbir doküman , belirtilmezse çalışmalar yazmak için engellenir.

Password: Eğer bu özellik boş bırakılırsa kontrol ön değer şifreyi gönderecek .

Protocol: Kullanılan protokolu belirtilen değer ve metodta işletir .

Proxy: Accesstype özelliği sadece Icnamedproxye kurulduğunda kullanılır .

RemoteHost: Uzak bir bilgisayara bağlantı yapılacağını belirler.



RemotePort: Uzak bir bilgisayarda kullanılmak için Internet merkezini belirler.

RequestTimeout: Talepi tamamlamak için ikincinin sayisini bekler.

Url: Bu kontrol yoluyla URL' yi kullanir.

Tag: Herhangi bir mesaj saklamak için kullanilir.

Top: Üstten hizasini belirler.

UserName: Uzak bilgisayarlara talepler ile gönderilen ismi belirler. Eger bu özellik bos birakilirsa kontrol kullanıcı ismi olarak isimsizi gönderecektir.

Inet Fonksiyonlari:

Inet1.Execute: Uzak bilgisayar isteklerini isletir. Özellik protokolu için geçerli olan istekleri sadece gönderebilirsiniz .

Inet1.GetHeader: Getheader metodu http dosyasından baslik metnini geri almak kullanilir.

Inet1.hInternet: Temelde olan wininet-dll api'dan interneti tutar. Görülebilir temelden kontrole erisildiginde bu özellik kullanilmaz .

Inet1.OpenURL: Belirlenmis url' de doküman açar.

Inet1.ResponseCode: Iccerror durumu statechanged olayında gerçekleşdiginde baglanti hatasini belirler. Hatanin açıklaması için responseinfo özelligini kurar.

Inet1.ResponseInfo: Gerçeklesen son hatanın metnini belirler. Hata kodu için responsecode özelligini kurar.

G-Microsoft MAPI Controls:

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden Microsoft MAPI Controls seçeneği seçilerek istediğimiz Component' i ekleriz.

Mapi Session: Message uygulama programı ara birim mapi kontrolleri size görülebilir temel mapi uygulamaları sağlanan posta yaratmasına izin verirler. İki mapi kontrolü vardır.

Name: Program içerisindeki ismini belirler.

DownloadMail: Eğer herhangi bir yeni posta dowloaded olmalıysa belirtilen yeni oturum baslatıldığında dowloaded edilir. True ise edilir.

Index: Index numarasını belirler.

Left: Sola hizasını belirler.

LogonUI: Diyalog kutusunu belirtmek için kullanılır.

NewSession: Kullanılan geçerli oturum varsa bile yeni posta oturumu kurulmuş olması gerektiğini belirtir.

Password: Username özelliği ile ilgili hesap şifresini belirtir.

Tag: Herhangi bir mesaj saklamak için kullanılır.

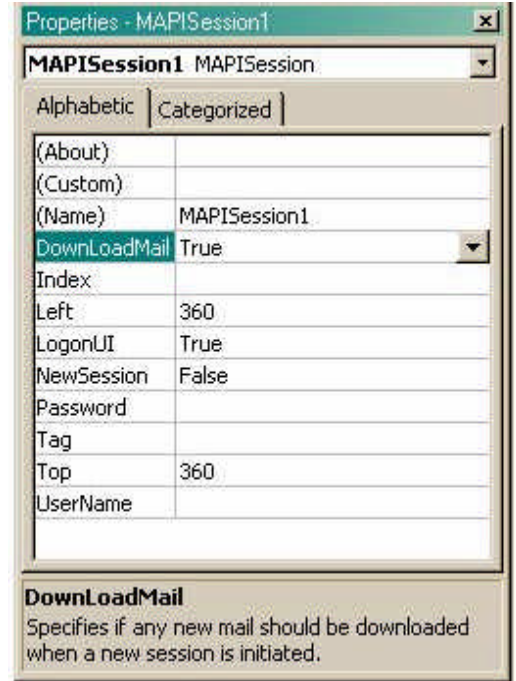
Top: Üstten hizasını belirler.

UserName: Hesap kullanıcı ismini belirler.

Mapi Fonksiyonları:

MAPI Session1.SessionID: Güncel message oturumunu belirler. Bu özellik tasarım zamanında mevcut değilse okunmaz.

MAPI Session1.SignOff: Message oturumu bit, username ve şifre özellikleri tarafından belirtilmeden dışarı gönderilmez.



MAPISession1.SignOn: Username ve sifre özellikleri tarafından belirtilen kullanıcı temelde olan mesaj alt sistemine oturumunu sağlar.

Mapi Messages: Mapimessages kontrolü message sistem fonksiyonunun değişikliğini gerçekleştirmek için kullanıcıya izin verir.

Name: Program içerisindeki ismini belirler.

AddressCaption: Adres kutusu üzerindeki ismini belirler.

AddressEditFieldCount: Adres dialog kutusunda kullanıcıya ait kontrolleri sayısını belirler.

AddressLabel: Adres kitabında kontrolü hazırlamak için görünüşlerini belirler.

AddressModifiable: Adres kitabının değiştirilebildiğini belirtir.

AddressResolveUI: Resolvename metodu belirtildiğinde dialog kutusu adres sırasında alıcı isim için gösterildiğini belirtir.

FetchMsgType: Mesaj tipini belirler.

FetchSorted: Inboxden mesajlar ile mesaj seti için mesaj düzenini belirler.

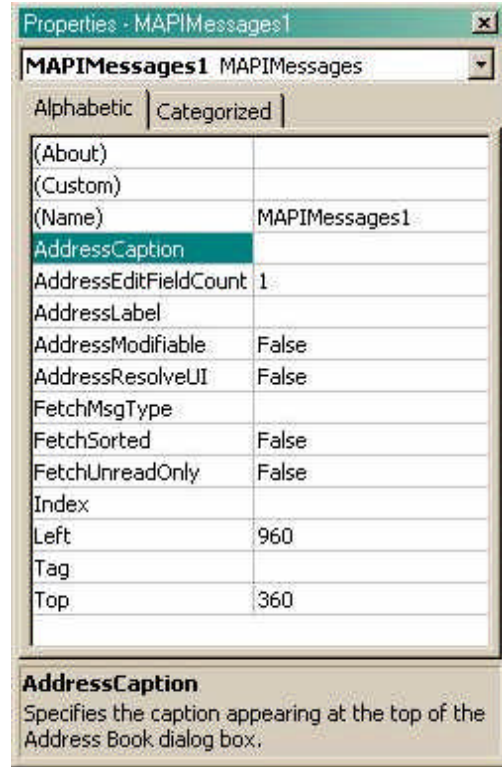
FetchUnreadOnly: Sadece unread mesajlarını mesaj setinde kısıtlaması gerektiğini belirtir.

Index: Index numarasını belirler.

Left: Sola hizasını belirler.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

Top: Üstten hizasını belirler.



Mapimessages Fonksiyonlari :

MAPIMessages1.Action: Mapimessages kontrolunun istenildiginde hareket etmesini belirler. Bu özellik tasarim zamaninda kullanilmaz.

MAPIMessages1.AttachmentCount: O anki indeksli mesaj ile ilgili baglantilarin toplam sayisi belirler. Bu özellik tasarim zamaninda kullanilmaz.

MAPIMessages1.AttachmentIndex: O anki indeksli baglantiyi belirtir. Bu özellik tasarim zamaninda kullanilmaz.

MAPIMessages1.AttachmentPathName: O anki indeksli baglantinin yol ismini belirtir. Bu özellik tasarim zamaninda kullanilmaz.

MAPIMessages1.AttachmentPosition: Mesaj bölümü içinde o anki indeksli baglanti pozisyonunu belirtir. Bu özellik tasarim zamaninda kullanilmaz. msgindex - 1 kurulmadukça read-only 'dir.

MAPIMessages1.Forward: Ileriye dogru mesaj gönderir.

MAPIMessages1.MsgConversationID: O anki indeksli mesaj için görüşme tanımlanmasi degerini belirtir. Msgindex - 1 kurulmadukça read-only' dir.

MAPIMessages1.MsgID: O anki indeksli mesajin belirtecini belirtir.

MAPIMessages1.MsgNoteText: Mesajin metin bölümünü belirtir. Bu özellik tasarim zamaninda kullanilmaz. Msgindex kurulmadukça read-only' dir.

MAPIMessages1.Reply: Mesaja cevap verir.

MAPIMessages1.ResolveName: O anki indeksli alicinin ismini çözer.

MAPIMessages1.Send: Mesaji gönderir.

MAPIMessages1.Show: Posta adres dialog kutusunu veya o anki indeksli alicinin detay bilgilerini gösterir.

H-Microsoft Masked Edit Control:

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden Microsoft Masked Edit Control seçeneği seçilerek istedigimiz Component' i ekleriz.

MaskedTextBox: Girilen veya gösterilen verinin tipi hakkında görülebilir bilgiler sağlar. Bu kontrol ikon olarak nasıl görüldüğüdür.

Name: Program içerisindeki ismini belirler.

Appearance: Formdaki görüntüsünü belirler.

Back color: Arka plan rengini belirler.

BorderStyle: Kenar çizgilerini belirler.

CausesValidation: Controlde odaklanmasını kaybetmiş olan oluşturmaları tekrar döndürme ya da kurma işlemini yapar.

ClipMode: esik yapıldığında girdi maskeleyişinde aynen karakterleri içermesi veya eğer dışlaması gereğine belirlenir veya kopya , kumanda eder .

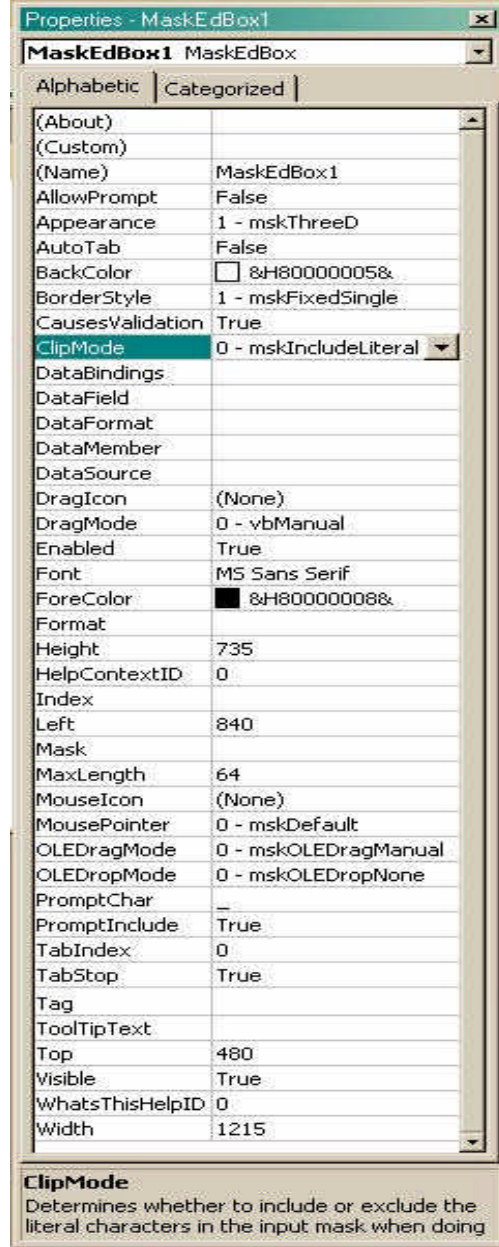
DataBindings: Geliştiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon Nesnesi geri döndürür.

DataField: Veritabanına bağlantı için alan seçilmesini sağlar.

DataFormat: Veritabanından alınacak bilginin formatını belirler.

DataMember: Data bağlantısı kurar ve bir değer belirtir.

DataSource: Veritabanının tablosunu



belirler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manuel olacağını belirler.

Enable: Clicklenebilirliğini belirler. Tru ise click olayı çalışır.

Font: Yazı tipini belirler.

ForeColor: Açıklama yazısının rengini belirler.

Height: Nesnenin yüksekliğini belirler.

HelpContextID: Indexli kullanımda indexini belirler.

Index: Index numarasını belirler.

Left: Sola hizasını belirler.

Mask: Kontrol için girdi maskeleymesini belirler .

MaxLength: maskelenmiş maksimum uzunluğu , kontrolü yayına hazırlar .

MouseIcon: Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Mouse nesnenin üzerindeyken hangi şekli alacağını belirler.

OLEDragMode: ya parça ya da programcı , ole drag/drop çalışmasını ele al .

PromptChar: karakter , girdi için kullanıcıyı harekete geçirdi .

PromptInclude: Çabuk karakterler metin özellik değerinde kapsanıldığını belirtir.

TabIndex: TAB'a basıldığında kaçinci atlamada kendine ulaşacağını belirler.

Tabstop: Tab ile seçilmesini sonlandırır.

Tag: Herhangi bir mesaj saklamak için kullanılır.

ToolTipText: Mouse nesne üzerindeyken mesaj verdirmek için kullanılır.

Top: Üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Bir nesne için belirtilmiş içerik numaralarını belirler.

Width: Nesnenin genişliğini belirler.

MaskedTextBox Fonksiyonları:

MaskedTextBox1.Container: Formda kontrolün mevcut atdesign zamanlamasını yapmaz.

MaskedTextBox1.HideSelection: Kontrol odagi kaybettiğinde metni seçtiğini belirleyen değeri belirler.

MaskedTextBox1.HWnd: Form veya kontrole ahandle kurar.

MaskedTextBox1.Refresh: Yenileme yapar.

MaskedTextBox1.SetFocus: Belirlenmiş kontrol veya forma the focus hareketini yapar.

I -Microsoft Windows Common Controls 6.0

Project menüsünden Component (Ctrl+T) seçeneği seçilir. Karşımıza gelen menüden **Microsoft Windows Common Controls 6.0** seçilerek, eklenir.

TOOLBAR : Toolbar kontrolü , düğme nesnesinin koleksiyonunu kapsar.

Name: Nesneyi tanımlamak için kullanılan isim.

Align: Nesnenin olduğu yeri belirleyen değeri setler.

AllowCustomize: Toolbar kontrolü ise değeri belirlemesi ve düzenlenmesi. son kullanıcı yoluyla, toolbar dialog kutusunu düzenler.

Appearance: Nesnenin görüntüsünü belirler.

BorderStyle: Kenar çizgilerini belirler.

ButtonHeight: Toolbar kontrolü düğmelerinin yüksekliği belirler.

ButtonWidth: Toolbar kontrolü düğmelerinin genişliği belirler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen değeri, Clicklenebilirliğini belirler. True ile Click olayı çalışır.

Height: Nesnenin yüksekliğini belirler.



HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

HelpFile: Yardım veya bağlantılı dokümantasyonu göstermek uygulamanız yoluyla kullanılan dosyalar.

Index: Indexli kullanımda indexini belirler.

Left: Nesnenin sola hizasını belirler.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Gösterilen fare işaret edicisinin tipini gösteren değeri belirler. Mouse nesne üzerindeyken hangi şekli alacağını belirler.

Negotiate: Siraya koyulabilen aktif nesneyi belirtir.

OLEDropMode: Hedef parçası çalışmalarını ele aldığını belirtir.

ShowTips: Tooltips nesne için gösterildiğini belirleyen değeri belirler.

Style: Nesnelerin biçimini belirler. Tipini ve çalışma düzeni gibi.

TabIndex: TAB tusuna basıldığında kaçinci TAB' da kendine ulaşacağını belirler.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

TextAlignment: Düğme ile ilgili metnin pozisyonunu belirleyen değer.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasını belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

Wrappable: Belirleyen değer toolbar ise, düğmeleri kontrol eder. pencere, resized olduğunda otomatik olarak sarar.

Toolbar Fonksiyonlari:

Toolbar1.Buttons: Düğme nesnesinin toolbar kontrolü koleksiyonuna ilişkin değerleri belirtir.

Toolbar1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

Toolbar1.Controls: Nesnede kapsanılan kontrollerin koleksiyonuna ilişkin değerleri belirtir.

Toolbar1.DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon nesnesini geri döndürür.

Toolbar1.DisabledImageList: Etkisiz kılınmış görüntüleri kullanmak üzere imagelist' i kontrol eder.

Toolbar1.HotImageList: Imleç clickable spotunda kaldığında hangi görüntüler için kullanılmak üzere imagelist kontrolü ve biçim özelliğini belirler, tbrtransparente kurulur.

Toolbar1.HWnd: Form veya kontrole ahandle kurar.

Toolbar1.ImageList: Baska kontrol ile ilgili olan herhangi bir imagelist 'i kontrol eder.

Toolbar1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

Toolbar1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

STATUS BAR : Statusbar kontrolü ve pencere sağlar. Ana formda konum verisinin çeşitlerini belirler.

Name : Nesneyi tanımlamak için kullanılan isim.

Align: Nesnenin olduğu yeri belirleyen değeri setler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen değer, Clicklenebilirliğini belirler. True ile Click olayı çalışır.

Font: RemoteData kontrolü güncel yazı tipini belirtir. Yazı tipini belirler.

Height: Nesnenin yüksekliğini belirler.

Index: Indexli kullanımda indexini belirler.

Left: Nesnenin sola hizasını belirler.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

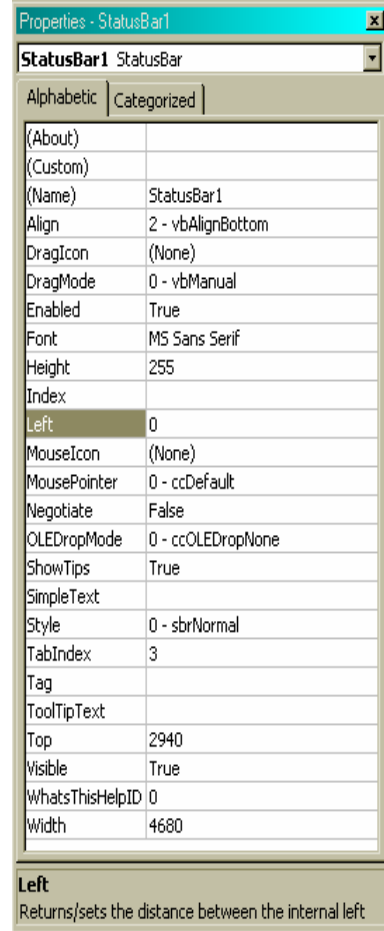
MousePointer: Gösterilen fare işaret edicisinin tipini gösteren değeri belirler. Mouse nesne üzerindeyken hangi şekli alacağını belirler.

Negotiate: Siraya koyulabilen aktif nesneyi belirtir.

OLEDropMode: Hedef parçası çalışmalarını ele aldığını belirtir.

Show tips: Tooltips nesne için gösterildiğini belirleyen değeri belirler.

Style: Nesnelerin biçimini belirler. Tipini ve çalışma düzeni gibi.



TabIndex: TAB tusuna basildiginda kaçinci TAB' da kendine ulasacagini belirler.

Tag: Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanilir.

Top: Nesnenin üstten hizasini belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyecegini belirler. True görünür.

WhatsThisHelpID: Nesne için birlesmis içerik sayisi setler.

Width: Nesnenin genisligini belirler.

StatusBar Fonksiyonlari:

StatusBar1.Container: Formda kontrole ait mevcut atdesign zamanlamasi yapmaz.

StatusBar1.HWnd: Form veya kontrole ahandle kurar.

StatusBar1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

StatusBar1.Panels: Panel nesnesinin koleksiyonuna iliskin degeri belirirler.

StatusBar1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

PROGRESS BAR :

Progressbar, soldan kısa ve kalın parçalar ile dikdörtgeni doldurarak uzun çalışmanın gelişmesini kontrol eder.

Name: Nesneyi tanımlamak için kullanılan isim.

Align: Nesnenin olduğu yeri belirleyen degeri setler.

Appearance: Nesnenin görüntüsünü belirler.

BorderStyle: Kenar çizgilerini belirler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen deger, Clicklenebilirliğini belirler. True ile Click olayı çalışır.

Height: Nesnenin yüksekliğini belirler.

Index: Indexli kullanımda indexini belirler.

Left: Nesnenin sola hizasını belirler.

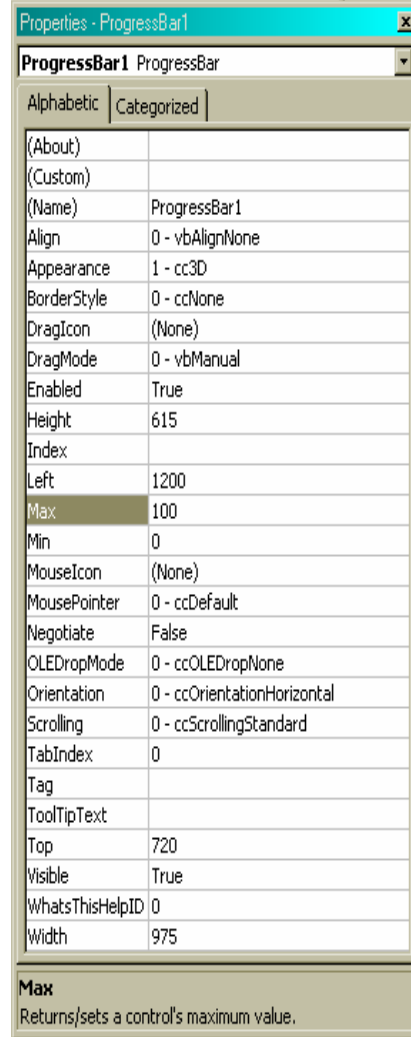
Max: ScrollBar ya da SpinButton degeri için max degeri belirler.

Min: ScrollBar ya da SpinButton degeri için min degeri belirler.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Gösterilen fare işaret edicisinin tipini gösteren degeri belirler. Mouse nesne üzerindeyken hangi şekli alacağını belirler.

Negotiate: Siraya koyulabilen aktif nesneyi belirtir.



OLEDropMode: Hedef parçası çalışmalarını ele aldığını belirtir.

Orientation: Nesneyi yönünü (yatay veya dikey) olarak belirleyen değer.

Scrolling: Gelisme gösterimini belirleyen değeri kurar.

TabIndex: TAB tusuna basıldığında kaçinci TAB' da kendine ulaşacağını belirler.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasını belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

ProgressBar Fonksiyonları:

ProgressBar1.Container: Formda kontrole ait mevcut at design zamanlaması yapmaz.

ProgressBar1.HWnd: Form veya kontrole ahandle kurar.

ProgressBar1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

ProgressBar1.Parent: Kontrol veya başka nesneyi kapsayan form nesneyi geri döndürür.

ProgressBar1.Value: Seçenek seçildiğinde true değeri alır.

TREEVIEW :

Treeview kontrol gösterimleri etiket ve isteğe bağlı bitmapdan oluşan her bir düğüm nesnesinin hiyerarsik listesi belirtir.

Name: Nesneyi tanımlamak için kullanılan isim.

Appearance: Nesnenin görüntüsünü belirler.

BorderStyle: Kenar çizgilerini belirler.

CausesValidation: Onaylamada kaybeden kontrolda gerçekleşdiği setleri gösterir.

Checkboxes: Checkboxes' i belirleyen değer.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen değer, Clicklenebilirliğini belirler. True ile Click olayı çalışır

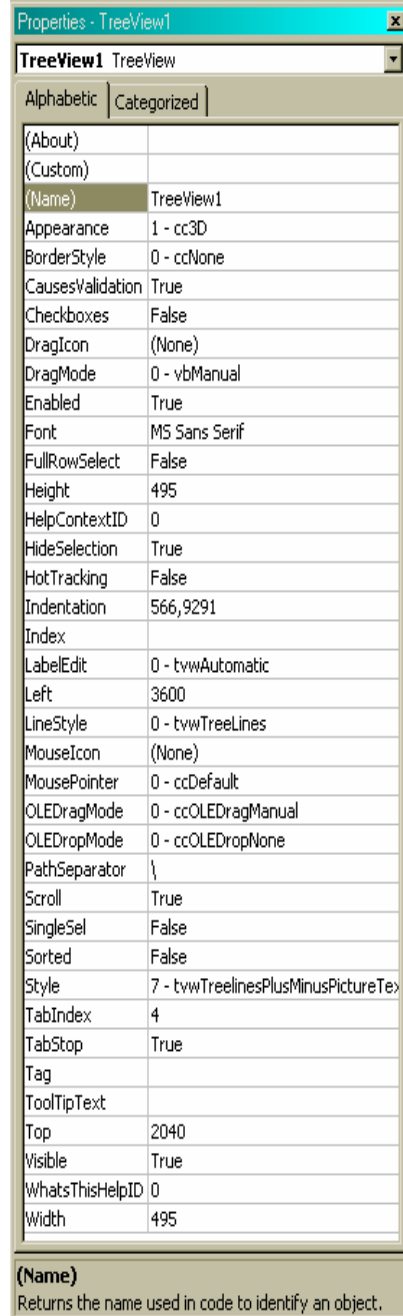
Font: RemoteData kontrolü güncel yazı tipini belirtir. Yazı tipini belirler.

FullRowSelect: Belirtilen bütün değeri sıra ile seçer.

Height: Nesnenin yüksekliğini belirler.

HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlıları belirler.

HideSelection: Kontrol odagi kaybettiginde metni seçtiğini belirleyen değeri belirler.



HotTracking: Mouse sensitive saglanildigini belirleyen deger.

Indentation: Kontrolde nesnelerin girinti genisligini belirler.

Index: Indexli kullanimda indexini belirler.

LabelEdit: Listview veya treeview kontrolundaki listitem veya düğüm nesnesinin etiketlerini yayina hazirlayabilir.

Left: Nesnenin sola hizasini belirler.

LineStyle: Düğüm arasında gösterilen çizgilerin biçimi belirler.

MouseIcon: Davranis biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacagini belirler.

MousePointer: Gösterilen fare isaret edicisinin tipini gösteren degeri belirler. Mouse nesne üzerindeyken hangi sekli alacagini belirler.

OLEDragMode: Bu kontrol OLE sürükleme olayinda kaynagi ve bu islem gibi davranabildigi belirtir.

OLEDropMode: Hedef parçasi çalismalarini ele aldigini belirtir.

PathSeperator: Delimiter karakterini yol için kullanir. Fullpath özelligi yoluyla geri döner.

Scroll: Scrollbars ile belirten degeri gösterir.

SingleSel: Belirten deger madde ise, genislik seçilir.

Sorted: İçinde bulunan seçenekleri siralar.

Style: Nesnelerin biçimini belirler. Tipini ve çalışma düzeni gibi

TabIndex: TAB tusuna basildiginda kaçinci TAB' da kendine ulasacagini belirler.

Tag: Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanilir.

Top: Nesnenin üstten hizasini belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyecegini belirler. True görünür.

WhatsThisHelpID: Nesne için birlesmis içerik sayisi setler.

Width: Nesnenin genisligini belirler.

TreeView Fonksiyonlari:

TreeView1.Container: Formda kontrol'e ait mevcut atdesign zamanlamasi yapmaz.

TreeView1.DropHighlight: Düğüm veya listemeye iliskin belirtilen , imleci hareket ettirildiginde sistem ile , rengi belirtir.

TreeView1.HWnd: Form veya kontrol'e ahandle kurar.

TreeView1.ImageList: Baska kontrol ile ilgili olan herhangi bir imagelist 'i kontrol eder.

TreeView1.Nodes: Treeview kontrol düğüm nesnesinin koleksiyonuna iliskin degeri belirler.

TreeView1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

TreeView1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

TreeView1.Selecteditem: Seçilmiş listitem ve düğüme iliskin listeleri belirler.

LISTVIEW :

Listview ,dört farklı görümlerin birini kullanan gösterimler maddelerini kontrol eder.

Name: Nesneyi tanımlamak için kullanılan isim.

Appearance: Nesnenin görüntüsünü belirler.

Arrange: Listview kontrolü ikonu veya smallicon görünümünde ikonlarının düzenlenildiğini belirleyen değer.

BackColor: Arka plan rengini belirler.

BorderStyle: Kenar çizgilerini belirler.

CausesValidation: Onaylamada kaybeden kontrolda gerçekleşdiği setleri gösterir.

Checkboxes: Checkboxes'i belirleyen değer.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen değer, Clicklenebilirliğini belirler. True ile Click olayı çalışır

FlatScrollBar: Nesnede scrollbarın görünüşlerini belirleyen değer.

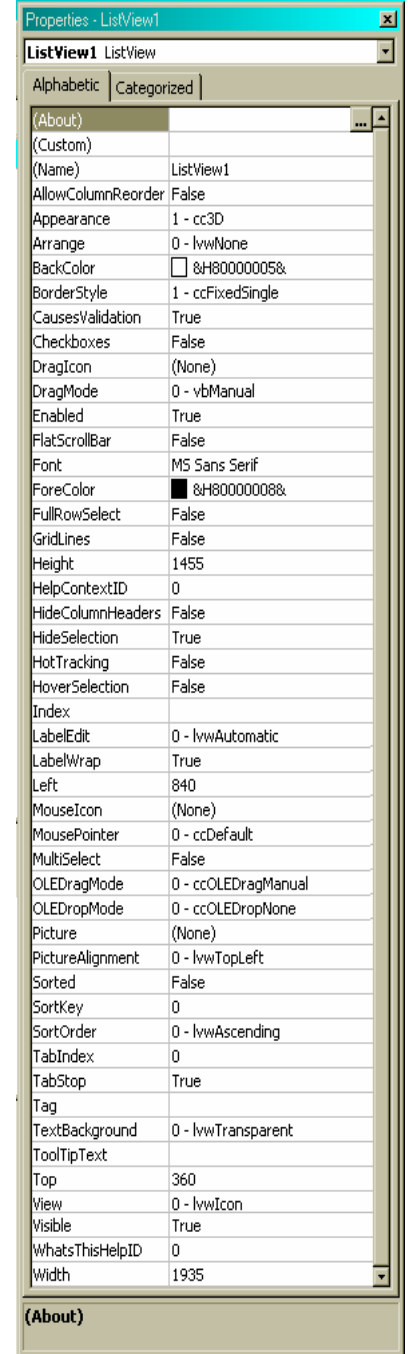
Font: RemoteData kontrolü güncel yazı tipini belirtir. Yazı tipini belirler.

ForeColor: Açıklama yazısının rengini belirler.

FullRowSelect: Belirtilen bütün değeri sıra ile seçer

GridLines: Çizilen çizgilerin kenar rengini belirler.

Height: Nesnenin yüksekliğini belirler.



HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

HideColumnSelection: Columnheader listview kontrolunda , rapor görüşünde saklanır .

HideSelection: Kontrol odagi kaybettiginde metni seçtigini belirleyen degeri belirler.

HotTracking: Mouse sensitive belirteni saglanildigini belirleyen deger belirlenir.

HoverSelection: Belirleyen deger listitem nesnesi ise, fare seçilen deger üstünde durur.

Index: Indexli kullanimda indexini belirler.

LabelEdit: Kullanici eger belirttigi deger, listview veya treeview kontrolunda ise listitem veya düğüm nesnesinin etiketlerini yayina hazirlayabilir .

LabelWrap: Etiket degerini koyar.

Left: Nesnenin sola hizasini belirler.

MouseIcon: Davranis biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacagini belirler.

MousePointer: Gösterilen fare isaret edicisinin tipini gösteren degeri belirler. Mouse nesne üzerindeyken hangi sekli alacagini belirler.

MultiSelect: Listedenden birden çok seçenegi seçmeyi saglayan özelliktir.3 seçenegi vardır.0 Çoklu seçim yok 1 Çoklu seçenek bir bütün halinde 2 toplu seçenekler ctrl tusu ile teker teker yapilir.

OLEDragMode: Bu kontrol OLE sürükleme olayinda kaynagi ve bu islem gibi davranabildigi belirtir.

OLEDropMode: Hedef parçasi çalismalarini ele aldigini belirtir.

Picture: Bir resme baglanmayi saglar.

PictureAlignment: Nesnenin resmi siraya koymasini belirleyen deger.

Sorted: İçinde bulunan seçenekleri siralar.

SortKey: Listitem listview kontrolünde belirtilen deger , siralanilir.

SortOrder: Listitem listview kontrolünde belirtilen değer , yukarı çıkmada veya inmede , sıralayıp düzenler.

TabIndex: TAB tusuna basıldığında kaçinci TAB' da kendine ulaşacağını belirler.

TabStop: TAB ile seçilmiş olanı sonlandırılır.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

TextBackground: Listitem nesnesi metin gerisi ise iki tane belirtilen seçenek vardır, 0 saydam 1 donuktur.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanılır. Top: Nesnenin üstten hizasını belirler.

View: Listview kontrolünde listitem nesnesinin görünüşlerini belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

ListView Fonksiyonları:

ListView1.ColumnHeaderIcons: Columnheaders koleksiyonu İçin görüntüleri sağlayan ImageList' i kontrol eder.

ListView1.ColumnHeaders: Columnheader nesnesinin koleksiyonuna ilişkin değeri belirler.

ListView1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

ListView1.DropHighlight: Düğüm veya listemeye ilişkin belirtilen , imleç hareket ettirildiğinde sistem ile , rengi belirtir.

ListView1.HWnd: Form veya kontrole ahandle kurar.

ListView1.Icons: Listview kontrolunda ikon ve smallicon görüsü ile ilgili imagelist kontrolünü belirler.

ListView1.ListItems: Listview kontrolunda listitem nesnesinin koleksiyonuna ilişkin degeri belirler.

ListView1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

ListView1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

ListView1.Selecteditem: Seçilmiş listitem ve düğüme ilişkin listeleri belirler.

ListView1.SmallIcons: Listview kontrolunda ikon ve smallicon görüsü ile ilgili imagelist kontrolünü belirler.

IMAGELIST :

Imagelist kontrolü, indeksi veya anahtar tarafından basvurulabilen listimage nesnesi, herbirin koleksiyonu kapsar .

Name: Nesneyi tanımlamak için kullanılan isim.

BackColor: Arka plan rengini belirler.

Index: Indexli kullanimda indexini belirler.

ImageHeight: Listimage nesnesinin yüksekligini kontrol eder.

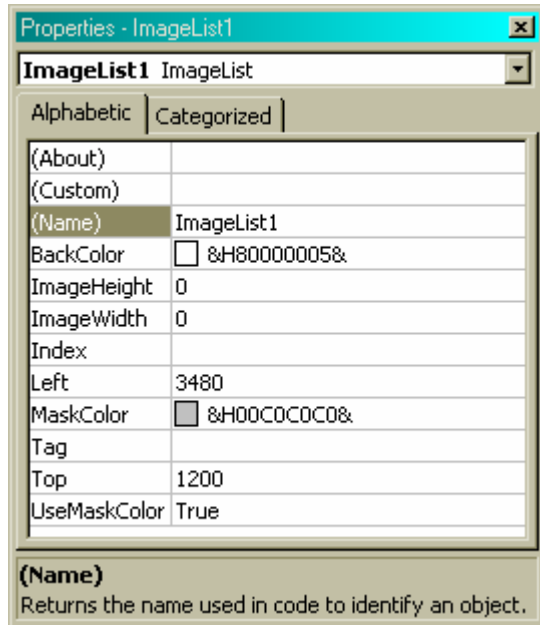
ImageWidth: Listimage nesnesinin genişligini kontrol eder .

Left: Nesnenin sola hizasini belirler.

MaskColor: Imagelist kontrolu için renkli maskelemeler yaratir.

Tag: Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

Top: Nesnenin üstten hizasini belirler.



UseMaskColor: Renk Maskcolor özelliginde tahsis etdigini belirleyen deger , maskeleme olarak kullanilir.

ImageList Fonksiyonlari:

ImageList1.hImageList: Imagelist kontrolü ile ilgili tutaç ayarlari yapar.

ImageList1.ListImages: Imagelist kontrolünde listimage nesnesinin koleksiyonuna degeri belirler.

ImageList1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

ImageList1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

SLIDER : Slider kontrolü, slideri kapsayan penceredir ve istege bagli isareti isaretler.

Name: Nesneyi tanımlamak için kullanılan isim.

BorderStyle: Kenar çizgilerini belirler.

CausesValidation: Onaylamada kaybeden kontrolda gerçekleşdigi setleri gösterir.

DragIcon: Sürükleme olayi basladiginda farenin alacagi sekli belirler.

DragMode: Sürükleme olayinin otomatik veya manual olacagini belirler.

Enabled: Nesneye karsilik verebildigini belirleyen deger, Clicklenebilirliğini belirler. True ile Click olayi çalışır

Height: Nesnenin yüksekliğini belirler.

Properties - Slider1	
Slider1 Slider	
Alphabetic Categorized	
(About)	
(Custom)	
(Name)	Slider1
BorderStyle	0 - ccNone
CausesValidation	True
DragIcon	(None)
DragMode	0 - vbManual
Enabled	True
Height	255
HelpContextID	0
Index	
LargeChange	5
Left	480
Max	10
Min	0
MouseIcon	(None)
MousePointer	0 - ccDefault
OLEDropMode	0 - ccOLEDropNone
Orientation	0 - ccOrientationHorizontal
SelectRange	False
SelLength	0
SelStart	0
SmallChange	1
TabIndex	6
TabStop	True
Tag	
TextPosition	0 - sldAboveLeft
TickFrequency	1
TickStyle	0 - sldBottomRight
ToolTipText	
Top	960
Value	0
Visible	True
WhatsThisHelpID	0
Width	735
(Name)	
Returns the name used in code to identify an object.	

HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

Index: Indexli kullanimda indexini belirler.

Left: Nesnenin sola hizasini belirler.

Max: ScroolBar ya da SpinButton degeri için max degeri belirler.

Min: ScroolBar ya da SpinButton degeri için min degeri belirler.

MouseIcon: Davranis biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacagini belirler.

MousePointer: Gösterilen fare isaret edicisinin tipini gösteren degeri belirler. Mouse nesne üzerindeyken hangi sekli alacagini belirler.

OLEDropMode: Hedef parçasi çalışmalarını ele aldığını belirtir.

Orientation: Nesneyi yönünü (yatay veya dikey) olarak belirleyen deger.

SelectRange: Eger slider kontrolü , seçilmiş alana sahip olabilirse belirlenir.

SelLength: Seçilen karakterlerin sayisini belirler.

SelStart: Eger hiçbir metin seçilmezse seçilen metnin baslangıç noktası pozisyonunu belirtir .

Smallchange: ScroolBar ya da SpinButtondaki belgelerin kullanıcı tarafından girildiginde olusan hareketlerin oranini belirler.

TabIndex: TAB tusuna basildiginda kaçinci TAB' da kendine ulasacagini belirler.

TabStop: TAB ile seçilmiş olani sonlandırılır.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar

TextPosition: Gösterilen metnin pozisyonunu nesneye iliskin belirleyen deger.

TickFrequency: Tickfrequency özelligini ne kadar siklikta gerçeklestireceğini belirler.

TickStyle: Sliderde isaretin yerlestirmesi gösterir veya isaretleri kontrol eder.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.
Top: Nesnenin üstten hizasını belirler.

Value: Seçenek seçildiğinde true değeri alır.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

Slider Fonksiyonları:

Slider1.Container: Formda kontrole ait mevcut atdesign zamanlaması yapmaz.

Slider1.DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon nesnesini geri döndürür.

Slider1.GetNumTicks: Min ile slider kontrolünün max özellikleri arasında işaretlerin sayısı belirler.

Slider1.HWnd: Form veya kontrole ahandle kurar.

Slider1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

Slider1.Parent: Kontrol veya başka nesneyi kapsayan form nesneyi geri döndürür.

Slider1.Text: Nesnede kapsanılan metni belirler.

IMAGECOMBO : Kontrolün liste kisiminda her bir maddenin resmini belirler.

Name: Nesneyi tanımlamak için kullanılan isim.

BackColor: Arka plan rengini belirler.

BorderStyle: Kenar çizgilerini belirler.

CausesValidation: Onaylamada kaybeden kontrolda gerçekleşdiği setleri gösterir.

DataBindings: Gelistiriciye bindable özelliklerini kapsayarak mevcut databindings koleksiyon
Nesnesi geri döndürür.

DataField: Veritabanına bağlantı için alan seçilmesini sağlar.

DataFormat: Veritabanından alınacak bilginin formatını belirler.

DataMember: Data bağlantısı kurar ve bir değer belirtir.

DataSource: Veritabanının tablosunu belirler.

DragIcon: Sürükleme olayı başladığında farenin alacağı şekli belirler.

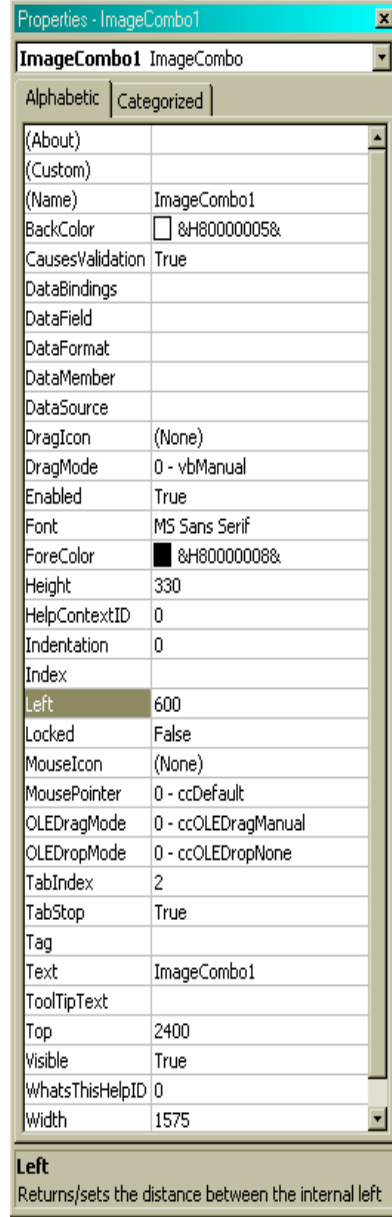
DragMode: Sürükleme olayının otomatik veya manual olacağını belirler.

Enabled: Nesneye karşılık verebildiğini belirleyen değer, Clicklenebilirliğini belirler.
True ile Click olayı çalışır

Font: RemoteData kontrolü güncel yazı tipini belirtir. Yazı tipini belirler.

ForeColor: Açıklama yazısının rengini belirler.

Height: Nesnenin yüksekliğini belirler.



HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

Indentation: Kontrolde nesnelerin girinti genişliğini belirler.

Index: Indexli kullanımda indexini belirler.

Left: Nesnenin sola hizasını belirler.

Locked:Text kutusuna girişi engeller.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Gösterilen fare işaret edicisinin tipini gösteren değeri belirler. Mouse nesne üzerindeyken hangi şekli alacağını belirler.

OLEDragMode: Bu kontrol OLE sürükleme olayında kaynağı ve bu işlem gibi davranabildiği belirtir.

OLEDropMode: Hedef parçası çalışmalarını ele aldığını belirtir.

TabIndex: TAB tusuna basıldığında kaçınca TAB' da kendine ulaşacağını belirler.

TabStop: TAB ile seçilmiş olanı sonlandırılır.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar

Text: Nesnede kapsanılan metni belirler.

ToolTipText: Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasını belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

ImageCombo Fonksiyonlari:

ImageCombo1.ComboItems: Comboitems koleksiyonu ile imagecombo kontrolündeki bütün comboitem nesnesini kapsar .

ImageCombo1.Container: Formda kontrol'e ait mevcut atdesign zamanlamasi yapmaz.

ImageCombo1.DataChanged: Thebound kontrolünde veri degistirildigi degeri gösterir

ImageCombo1.HWnd: Form veya kontrol'e ahandle kurar.

ImageCombo1.ImageList: Baska kontrol ile ilgili olan herhangi bir imagelist 'i kontrol eder.

ImageCombo1.Object: Özellik veya metodu görülebilir yolla kontrol eder otomatik olarak ismini belirler.

ImageCombo1.Parent: Kontrol veya baska nesneyi kapsayan form nesneyi geri döndürür.

ImageCombo1.Selecteditem: Seçilmiş listitem ve düğüme iliskin listeleri belirler.

ImageCombo1.SelLength: Seçilen karakterlerin sayisini belirler.

ImageCombo1.SelStart: Eger hiçbir metin seçilmezse seçilen metnin baslangiç noktası pozisyonunu belirtir .

ImageCombo1.SelText: Eger hiçbir karakterler seçilmezse su anki seçilmiş metni kapsayan string'den oluşur .

I-MICROSOFT MULTIMEDIA DTCs

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden MICROSOFT MULTIMEDIA DTCs seçeneği seçilerek istedigimiz Component' i ekleriz.

Page Transitions :

Sayfa geçişleri ile ilgili ayarlamaları yapar.

Name : Nesneni adını gösterir. Bu özellik, sadece tasarım aşamasında tanımlanabilir.

CausesValidation: Onaylamada kaybeden kontrolda gerçekleştiği setleri gösterir.

DragIcon: Sürüklemeye olayı başladığında fareni alacağı şekli belirler.

DragMode: Sürüklemeye olayının otomatik veya manual olacağını belirler.

Height : Nesnenin yüksekliğini belirler.

HelpContextID : Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlıları belirler.

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Left : Nesnenin sola hizasını belirler.

TabIndex : TAB tusuna basıldığında kaçinci TAB' da kendine ulaşacağını belirler.

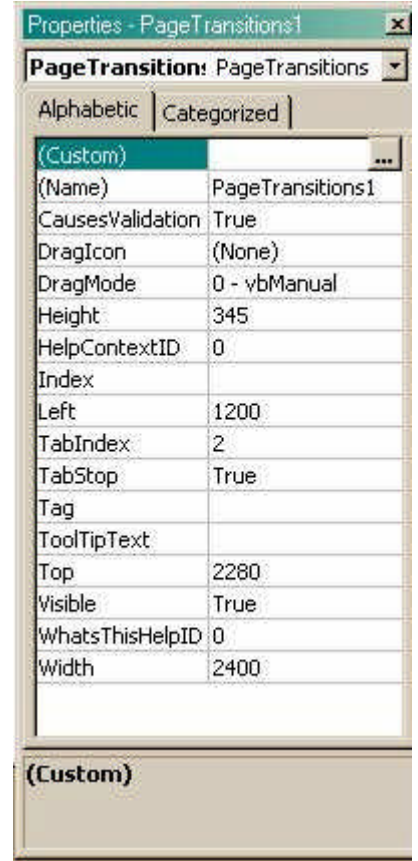
TabStop : TAB ile seçilmiş olanı sonlandırılır.

Tag : Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

ToolTipText : Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasını belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.



WhatsThisHelpID: Nesne için birlesmis içerik sayisi setler.

Width: Nesnenin genisligini belirler.

Page Transitions Fonksiyonlari

PageTransitions1.Name: Form , kontrolu tanımlamak kodta kullanılan isim veya veri, nesneye erisir.

PageTransitions1.Object: Özellik veya metod , görülebilir temel yoluyla kontrola otomatik olarak uzatılan isminin aynisindan özellik veya sahip olan kontrolun metoduna iliskin özellikler.

PageTransitions1.Parent: Kontrol veya baska nesne veya koleksiyonu kapsayan form , nesne veya koleksiyon.

PageTransitions1.Setfocus: Nesnenin etkin konum önceligine sahip olmasini saglar.

PageTransitions1.Drag: Basla , bit veya herhangibir kontrolun sürükleme çalışmasını iptal eder. Çizgi , menü , şekil , zamanlayıcı veya comondialog , kontrol eder .

PageTransitions1.Move: Midiform veya formu hareket veya kontrol eder.

PageTransitions1.Showwhatsthis: Pencere 95' nin yanında yardım saglanılan bu popup ne oldugunun kullanarak yardım dosyasında seçilmiş konuyu gösterir.

Time Lines

Name : Nesneni adini gösterir.Bu özellik, sadece tasarim asamasinda tanimlanabilir.

CausesValidation: Onaylamada kaybeden kontrolda gerçeklesdigi setleri gösterir.

DragIcon: Sürüklemeye olayi basladiginda farenin alacagi sekli belirler.

DragMode: Sürüklemeye olayinin otomatik veya manual olacagini belirler.

Height : Nesnenin yüksekligini belirler.

HelpContextID : Bir nesne için yardımcı dosyanin ve kimligin içeriğindeki yanlıslari belirler.

Index : Üzerinde çalışılan nesnenin, ayni ad altında toplanan dizi nesnelerden hangisi oldugunu belirten sayiyi içerir.

Left : Nesnenin sola hizasini belirler.

TabIndex : TAB tusuna basildiginda kaçinci TAB' da kendine ulasacagini belirler.

TabStop : TAB ile seçilmiş olani sonlandırılır.

Tag : Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

ToolTipText : Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasini belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.



Time Lines Fonksiyonlari

TimeLines1.Name: Form , kontrolu tanımlamak kodta kullanılan isim veya veri, nesneye erişir.

TimeLines1.Object: Özellik veya metod , görülebilir temel yoluyla kontrole otomatik olarak uzatılan isminin aynısından özellik veya sahip olan kontrolün metoduna ilişkin özellikler.

TimeLines1.Parent: Kontrol veya başka nesne veya koleksiyonu kapsayan form, nesne veya koleksiyon.

TimeLines1.Setfocus: Nesnenin etkin konum önceliğine sahip olmasını sağlar.

TimeLines1.Drag: Başla , bit veya herhangi bir kontrolün sürükleme çalışmasını iptal eder. Çizgi , menü , şekil , zamanlayıcı veya comondialog , kontrol eder .

TimeLines1.Move: Midiform veya formu hareket veya kontrol eder.

TimeLines1.Showwhatsthis: Pencere 95' nin yanında yardım sağlanan bu popup ne olduğunun kullanarak yardım dosyasında seçilmiş konuyu gösterir.

J-MICROSOFT REMOTEDATA CONTROL

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden MICROSOFT REMOTEDATA CONTROL seçeneği seçilerek istediğimiz Component' i ekleriz.

MSRDC :

Sinir kontrolü boyunca uzak odbc veri kaynağında depolanılan veriye erişim sağlar.

Name: Nesneni adını gösterir. Bu özellik, sadece tasarım aşamasında tanımlanabilir.

Align: Nesnenin olduğu yeri belirleyen değeri setler.

Appearance: Veri kontrolünün Appearance'ini belirtir. Nesnenin görüntüsünü belirler.

BackColor: RemoteData kontrolü BackColorunu belirtir. Nesnenin arka plan rengini belirler.

BatchSize: Yigindaki kayıtların sayısını güncelleştirir.

BOFAction: Dosyaların başlangıç noktalarına hareket kazandırmak için kullanılır. Alınan dosya hareketinin başladığını belirtir.

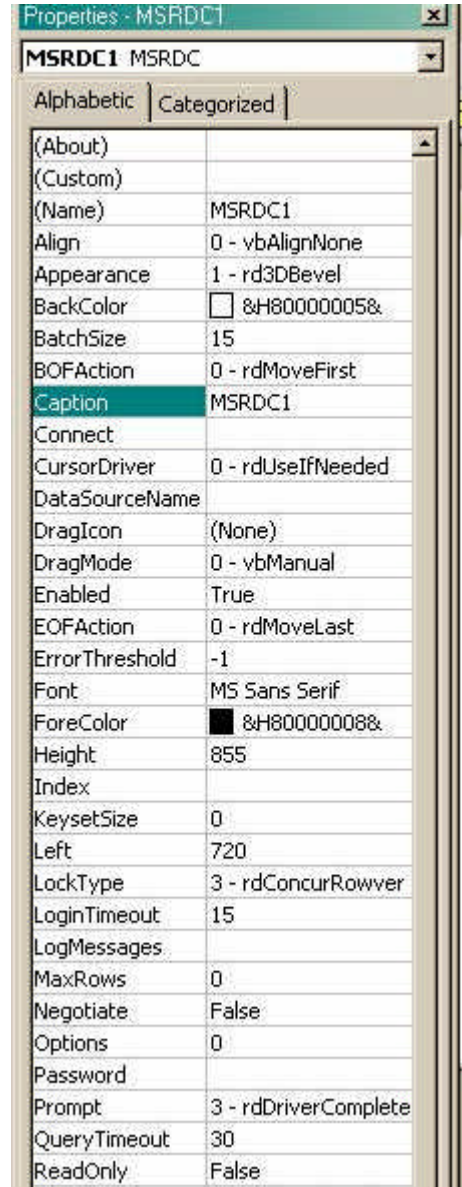
Caption: Remotedata kontrolü ünvanını belirler. Başlık bilgisi tanımlanır.

Connect: Açık bağlantının kaynağındaki liste bilgisini belirtir.

CursorDriver: Yaratılmak istenen imlecin tipini belirtir.

DataSourceName: Veritabanına bağlanacak nesnenin ismini belirtir.

DragIcon: Sürükleme olayı başladığında farene alacak şekli belirler.



DragMode: Sürükleme olayinin otomatik veya manual olacagini belirler.

Enable: Nesneye karsilik verebildigini belirleyen deger, Clicklenebilirliğini belirler. True ile Click olayi çalisir.

EOFAction: Alinilan dosya hareketi degerini sonlandigini belirler.

ErrorThreshold: Programda büyük derecedeki hatalari belirler.

Font: RemoteData kontrolu güncel yazi tipini belirtir. Yazı tipini belirler.

ForeColor: Remotedata kontrolu ForeColorunu belirtir. Açıklama yazisinin rengini belirler.

Height: Nesnenin yüksekligini belirler.

Index: Üzerinde çalisilan nesnenin, ayni ad altinda toplanan dizi nesnelerden hangisi oldugunu belirten sayiyi içerir.

KeysetSize: Keyset tamponunda siralarin sayisini belirler.

Left: Nesnenin sola hizali olacagini belirler.

LockType: Concurrency tasimasinin tipini belirtir.

LoginTimeout: Veri kaynagini baglamak için zaman asimi degerini belirler.

LogMessages: ODBC iz dosyasinin yerini belirtir.

MaxRows: Geri dönölmek için siralarin maksimum sayisi belirtir.

Negotiate: Siraya koyulabilen aktif nesneyi belirtir.

Options: Rdoresultsetin speciefies seçenegi ayarlari yapilir.

Password: Rdoenvironmentin yaratilmasi sirasinda sifre kullanimini saglar.

Prompt: Çabuk seçenegi ODBCÜ belirtir.

QueryTimeout: Kusku isletimi için zaman asimi degerini belirtir.

ResultSetType	1 - rdOpenKeyset
RowsetSize	100
SQL	
Tag	
ToolTipText	
Top	720
UserName	
Visible	True
WhatsThisHelpID	0
Width	2055

Caption
Determines RemoteData control's caption

ReadOnly: RemoteData kontrolu veri kaynagi, sadece okunuldugunu belirtir.

ResultSetType: RemoteData kontrolu yoluyla desteklenmis resultset tipini belirler.

RowsetSize: Rowsetde siralarin sayisini belirler.

SQL: SQL' de açık ve net bir sekilde sorgu veya anlatim tanimlamasi yapar.

Tag: Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

ToolTipText : Mouse nesne üzerinde iken mesaj verdirmek için kullanilir.

Top: Nesnenin üstten hizasini belirler.

UserName: Kullaniciyi belirtir.

Visible: Nesnenin form üzerinde görünüp görünmeyecegini belirler. True görünür.

WhatsThisHelpID: Nesne için birlesmis içerik sayisi setler.

Width: Nesnenin genisligini belirler.

MSRDC Fonksiyonlari

MSRDC1.BatchCollisionCount: Son batch - mode güncellestirmesini tamamlamayan siralarin sayisini belirten deger.

MSRDC1.Cancel: Eszamanli olmayan modta olusan kuskunun islemini iptal et veya herhangi biri iptal et esnasinda belirlenmis rdoya karsi sonuçlar karsi olur.

MSRDC1.Connection: RemoteData kontrolu temelde olan rdoconnection nesnesine iliskin deger.

MSRDC1.Drag: Basla, bit veya herhangi bir kontrolun sürüklenme çalismasini iptal eder. Çizgi,menü, sekil, zamanlayici veya commondialog, kontrol eder.

MSRDC1.EditMode: Güncel sıra için yayına hazırlanin durumunu belirten deger.

MSRDC1.Environment: RemoteData kontrolu temelde olan rdoenvironment nesnesisine iliskin deger.

MSRDC1.Move: Midiform veya formu hareket veya kontrol eder.

MSRDC1.Object: Özellik veya metod , görülebilir temel yoluyla kontrola otomatik olarak uzatılan isminin aynisindan özellik veya sahip olan kontrolun metoduna iliskin özellikler.

K-MICROSOFT RICH TEXTBOX CONTROLS

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden MICROSOFT RICH TEXTBOX CONTROLS seçeneği seçilerek istedigimiz Component' i ekleriz.

RichTextBox :

Klasik TextBox kontrolünden format özelliği ileri gidilen daha çoğu aynı zamanda temin eder iken richtextbox kontrolü, metine girmek ve yayına hazırlamak için kullanıcıya izin verir.

Name: Nesneni adını gösterir. Bu özellik, sadece tasarım aşamasında tanımlanabilir.

Appearance: Veri kontrolünün Appearance'ini belirtir. Nesnenin görüntüsünü belirler.

AutoVerbMenu: Seçilmiş nesnelerinin gösterileceği veya gösterme değerini belirler.

BackColor: Nesnenin arka plan rengini belirler.

BorderStyle: Nesne için sınır biçimini belirler.

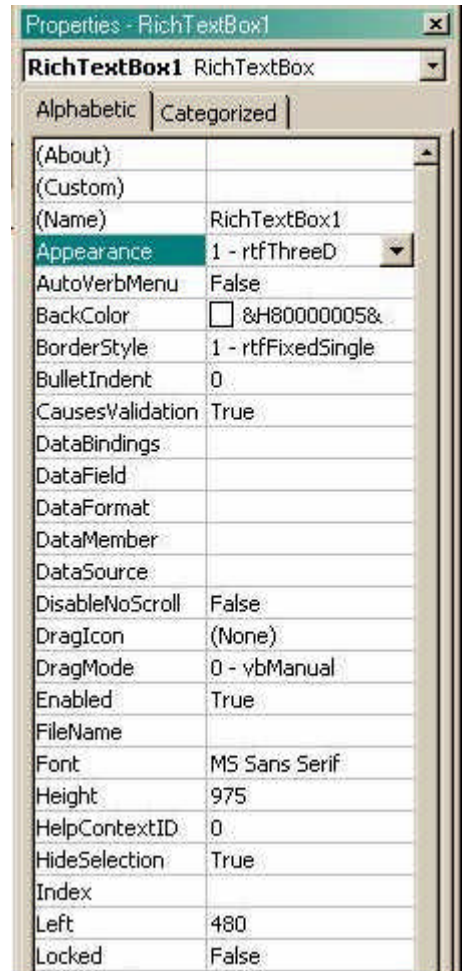
BulletIndent: RichTextBox' de kullanılan, içeriden başlamak miktarını, kontrol eder ve belirler.

CausesValidation: Onaylamada kaybeden kontrolde gerçekleşdiği setleri gösterir.

DataBindings: Mevcut olan Bindable özelliklerini biriktiren DataBindings koleksiyon nesnesini belirtir.

DataFormat: Bu parçanın Bindable özelliğine karşı kullanım için veri nesnesinin biçimini belirler.

DataMembers: Veri bağlantısını DataMemberi işaret eden değeri belirler.



DataSource: Veritabanındaki baglanilacak olan yeri belirler.

DisableNoScroll: RichTextBox kontrolunda tomar barlari oldugunu belirleyen degeri belirtir.

DragIcon: Sürükleme olayi basladiginda farenin alacagi sekli belirler.

DragMode: Sürükleme olayinin otomatik veya manual olacagini belirler.

Enable: Nesneye karsilik verebildigini belirleyen degeri belirler. Kullanilabilirliğini belirler.

FileName: Tasarım zamanında RichTextBox kontrolu içine yüklenilen dosyanın ismini, 'FileName' ini belirler.

Font: Yazı tipini belirler.

Height : Nesnenin yüksekliğini belirler.

HelpContextID : Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

HideSelection: Eger seçilmiş nesne highlighted (bir şeyin özel bir bölümüne dikkat çekmek) kalırsa degeri belirtir.

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Left: Nesnenin sola hizalı olacağını belirler.

Locked: RichTextBox kontrolunda içerik yayına hazırlanilabildiği degeri göstermesi sağlar.

MaxLength: RichTextBox karakterlerin maksimum sayısı olduğu degeri göstermesi sağlar.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Gösterilen fare işaret edicisinin tipini gösteren degeri belirler. Mouse nesne üzerindeyken hangi sekli alacağını belirler.

MaxLength	0
MouseIcon	(None)
MousePointer	0 - rtfdDefault
MultiLine	True
OLEDragMode	1 - rtfdOLEDragAuto
OLEDropMode	2 - rtfdOLEDropAuto
RightMargin	0
ScrollBars	0 - rtfdNone
TabIndex	0
TabStop	True
Tag	
Text	RichTextBox1
ToolTipText	
Top	960
Visible	True
WhatsThisHelpID	0
Width	2055

Appearance
Returns/sets the paint style of a control at run time.

MultiLine: RichTextBox kontrolu kabul edebildigi ve gösterebildigi degeri göstermesi saglar.

OLEDragMode: Bu kontrol OLE sürükleme olayinda kaynagi ve bu islem gibi davranabildigi belirtir.

OLEDropMode: Bu kontrol OLE sürükleme olayi bittiginde veya durdurulugunda çalisir.

RightMargin: Sag kenar için kullanılan Textwrap, Ortalama gibi özellikleri hazirlar.

ScrollBars: RichTextBox kontrolunun yataya sahip oldugunu gösteren degeri belirler.

TabIndex : TAB tusuna basildiginda kaçinci TAB' da kendine ulasacagini belirler.

TabStop : TAB ile seçilmiş olani sonlandırılır.

Tag : Programiniz için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

Text: Nesnede kapsanılan metini belirler.

ToolTipText : Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin üstten hizasını belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler

RichTextBox Fonksiyonlari

RichTextBox1.Container: Mevcut atdesign, zamanlama yapmaz .

RichTextBox1.Drag: Basla, bit veya herhangi bir kontrolün sürükleme çalışmasını iptal eder. Çizgi, menü, şekil, zamanlayıcı veya comondialog, kontrol eder.

RichTextBox1.Find: Verilen IP için RichTextBox kontrolunda metini arar.

RichTextBox1.hWnd: Form veya kontrole ahandle degerini belirler.

RichTextBox1.Move: Midiform veya formu hareket veya kontrol eder.

RichTextBox1.Object: Özellik veya metod , görülebilir temel yoluyla kontrole otomatik olarak uzatılan isminin aynısından özellik veya sahip olan kontrolun metoduna iliskin özellikler.

RichTextBox1.OLEDrag: OLE Drag / Drop çalışmasını başlatmak için kullanılır.

RichTextBox1.OLEObjects: OLEObjects koleksiyonu, OLEObject nesnesinin koleksiyonunu kapsar.

RichTextBox1.Parent: Kontrol veya baska nesne veya koleksiyonu kapsayan form, nesne veya koleksiyon.

RichTextBox1.Refresh: Nesnenin hem görünüş açısından hemde içerik açısından güncellenmesini sağlar.

RichTextBox1.SaveFile: Dosyaya RichTextBox kontrolunun içeriğini korur.

RichTextBox1.SelAlignment: RichTextBox kontrolunda paragrafların siraya koymasını kontrol eder.

RichTextBox1.SelBullet: Eger güncel seçim veya araya koymak noktasını kapsayarak RichTextBox kontrolunda paragraf hizli ve düzgün bir biçime sahip olmasını belirler.

RichTextBox1.SelCharOffset: RichTextBox kontrolunda metin alttaki subscript olarak veya üstte superscript olarak, ana hat (normal) de ana hat ana hat belirldigini belirler.

RichTextBox1.SelColor: RichTextBox kontrolunda metnin rengini belirler.

RichTextBox1.SelFontName: RichTextBox kontrolunda araya koymak noktasini izlemek için kullanilir.

RichTextBox1.SelPrint: Basmak için cihaza RichtextBox kontrolunda formatli metini gönderir.

RichTextBox1.SelStrikeThru: RichTextBoxde su anki seçilmis metnin dönen veya set yazi tipi biçimleri, kontrol ederler.

RichTextBox1.SelUnderline: RichTextBoxde su anki seçilmis metnin dönen veya set yazi tipi biçimleri, kontrol ederler.

RichTextBox1.SetFocus: Nesnenin etkin konum önceligine sahip olmasini saglar.

RichTextBox1.ShowWhatsThis: Pencerele 95' nin yaninda yardim saglanilan bu popup ne oldugunun kullanarak yardim dosyasinda seçilmis konuyu gösterir.

RichTextBox1.Span: RichTextBox kontrolunda dayanilan bir takim belirlenmis karakterleri metni seçer.

RichTextBox1.UpTo: RichtextBoxde belirlenmis karakter setinin üyesi olan ilk karakteri, kontrol eder.

L-MICROSOFT SYSINFO CONTROLS

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden MICROSOFT SYSINFO CONTROLS seçeneği seçilerek istedigimiz Component' i ekleriz.

SysInfo :

SYSINFO kontrolu , işletim sisteminin tarafından bütün uygulamalara gönderilen sistem mesajlarına karşılık vermek size izin verir.

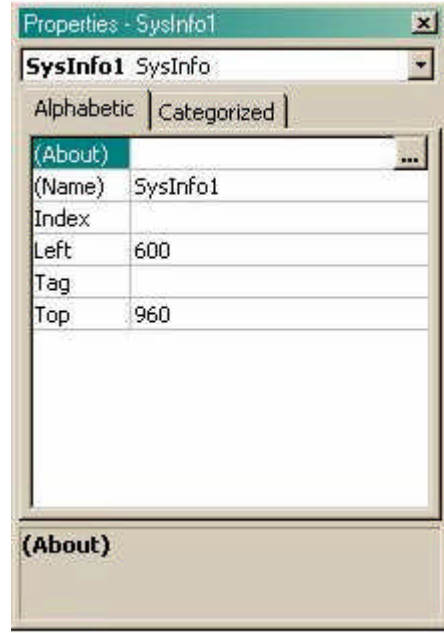
Name: Nesneni adını gösterir. Bu özellik, sadece tasarım aşamasında tanımlanabilir.

Index : Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Left: Nesnenin sola hizalı olacağını belirler.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

Top: Nesnenin üstten hizasını belirler.



SysInfo Fonksiyonlari

SysInfo1.BatteryFullTime: Bataryanın dolu yük yaşamını belirten değer.

SysInfo1.BatteryLifePercent: Dolu batarya güç kalanının yüzdesini belirler.

SysInfo1.BatteryLifeTime: Bataryanın kalan yaşamını belirten değer.

SysInfo1.BatteryStatus: Bataryanın (S) yükünün konumunu belirten değer.

SysInfo1.Object: Özellik veya metod , görülebilir temel yoluyla kontrole otomatik olarak uzatılan isminin aynısından özellik veya sahip olan kontrolün metoduna ilişkin özellikler.

SysInfo1.OSBuild: Su anki kullanılan işletim sistemi hakkında bilgi sağlayan değer.

SysInfo1.OSPlatform: Su anki kullanılan işletim sistemini tanımlayan değer.

SysInfo1.OSVersion: Su anki kullanılan işletim sisteminin versiyonunu tanımlayan değer.

SysInfo1.Parent: Kontrol veya başka nesne veya koleksiyonu kapsayan form , nesne veya koleksiyon.

SysInfo1.WorkAreaHeight: Pencere 95 için taskbar görünebilir masaüstü ayarlarının yüksekliğini ayarlar.

SysInfo1.WorkAreaTop: Pencere 95 için görünebilir masaüstü ayarlarının üst kenarı için taskbar koordinat değerini belirler.

SysInfo1.WorkAreaWidth: Pencere 95 için taskbar görünebilir masa üstü ayarlarının genişliği değerini belirler.

M-MICROSOFT TABBED DIALOG CONTROL

Project menüsünden Components (Ctrl+T) seçenegi seçilir. Gelen pencereden MICROSOFT TABBED DIALOG CONTROL seçenegi seçilerek istedigimiz Component' i ekleriz.

SSTab :

SSTAB kontrolu, diger kontrollar gibi davranan bir grup tablolarin, herbirini saglar.

Name: Nesneni adini gösterir. Bu özellik, sadece tasarim asamasinda tanimlanabilir.

BackColor: Nesnenin arka plan rengini belirler.

Caption: Kontrolu ünvanini belirler. Baslik bilgisi tanimlanir.

CausesValidation: Onaylamada kaybeden kontrolda gerçeklesdigi setleri gösterir.

DragIcon: Sürükleme olayi basladiginda farenin alacagi sekli belirler.

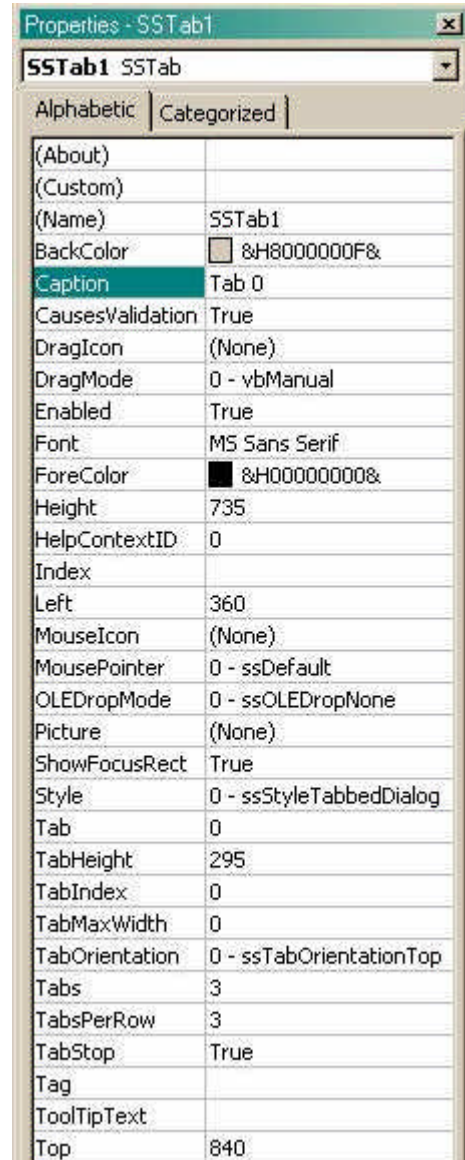
DragMode: Sürükleme olayinin otomatik veya manual olacagini belirler.

Enable: Nesneye karsilik verebildigini belirleyen deger, Clicklenebilirliğini belirler. True ile Click olayi çalisir.

Font: Tabbed Dialog kontrolu güncel yazi tipini belirtir. Yazı tipini belirler.

ForeColor: Tabbed Dialog kontrolu ForeColorunu belirtir. Açıklama yazisinin rengini belirler.

Height: Nesnenin yüksekliğini belirler.



HelpContextID: Bir nesne için yardımcı dosyanın ve kimliğin içeriğindeki yanlısları belirler.

Index: Üzerinde çalışılan nesnenin, aynı ad altında toplanan dizi nesnelerden hangisi olduğunu belirten sayıyı içerir.

Left: Nesnenin sola hizalı olacağını belirler.

MouseIcon: Davranış biçimi fare ikonu kurar. Mouse nesnenin üzerindeyken hangi resmi alacağını belirler.

MousePointer: Gösterilen fare işaret edicisinin tipini gösteren değeri belirler. Mouse nesne üzerindeyken hangi şekli alacağını belirler.

OLEDropMode: Bu kontrol OLE sürükleme olayı bittiğinde veya durdurulduğunda çalışır.

Picture: Kontrol üzerinde göstereceği resim tanımlanır.

ShowFocusRect: Odak dikdörtgeninde çizileceğini belirler.

Style: Nesnelerin biçimini belirler. Tipini ve çalışma düzeni gibi.

Tab: Aktif nesne sayısını belirler.

TabHeight: Nesnelerin yüksekliğini belirler.

TabIndex: TAB tusuna basıldığında kaçinci TAB' da kendine ulaşacağını belirler.

TabMaxWidth: Herbir nesnenin maksimum genişliği belirler.

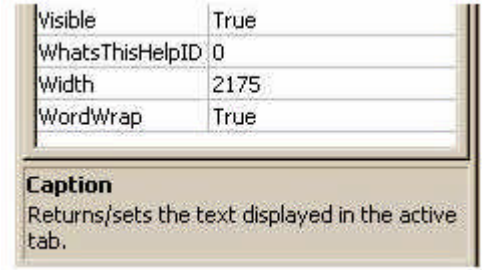
TabOrientation: Kontrol etiketlerinin hangi tarafa gideceğine karar verir. Kontrol etiketlerine yönlendirme yapar.

Tabs: Nesnelerin sayısını belirler.

TabPerRow: Herbirisi sırasında belirlemek için nesnelerin sayısı belirler.

TabStop: TAB ile seçilmiş olanı sonlandırılır.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.



ToolTipText : Mouse nesne üzerinde iken mesaj verdirmek için kullanılır.

Top: Nesnenin iç üst kenarı ile üst arasındaki mesafeyi belirler.

Visible: Nesnenin form üzerinde görünüp görünmeyeceğini belirler. True görünür.

WhatsThisHelpID: Nesne için birleşmiş içerik sayısı setler.

Width: Nesnenin genişliğini belirler.

WordWrap: Herbir nesnenin ünvanında metin sonraki çizgiye saracağını belirler.

SSTab Fonksiyonları

SSTab1.DataBindings: Gelistiriciye Bindable özelliklerini kapsayarak mevcut DataBindings koleksiyon nesnesi değerini belirler.

Sstab1.Drag: Basla, bit veya herhangi bir kontrolün sürükleme çalışmasını iptal eder. Çizgi, menü, şekil, zamanlayıcı veya comondialog, kontrol eder.

SSTab1.hWnd: Form veya kontrole ahandle değerini belirler.

Sstab1.Move: Midiform veya formu hareket veya kontrol eder.

Sstab1.Object: Özellik veya metod , görülebilir temel yoluyla kontrole otomatik olarak uzatılan isminin aynısından özellik veya sahip olan kontrolün metoduna ilişkin özellikler.

Sstab1.parent: Kontrol veya başka nesne veya koleksiyonu kapsayan form , nesne veya koleksiyon.

SSTab1.Rows: SSTab kontrolunda nesnelerin sıralarının toplam sayısı belirler.

Sstab1.SetFocus: Nesnenin etkin konum önceliğine sahip olmasını sağlar.

Sstab1.ShowWhatsThis: Pencere 95' nin yanında yardım sağlanan bu popup ne olduğunun kullanarak yardım dosyasında seçilmiş konuyu gösterir.

N-MICROSOFT WINSOCK CONTROLS

Project menüsünden Components (Ctrl+T) seçeneği seçilir. Gelen pencereden MICROSOFT WINSOCK CONTROLS seçeneği seçilerek istedigimiz Component' i ekleriz.

Winsock :

Kullaniciya winsock kontrolu, görülmez TCP' ye kolay erisim saglar. Microsoft erisimi, görülebilir temel, görülebilir c++ veya görülebilir foxpro gelistircilerinin tarafından kullanılabilir .

Name: Nesneni adini gösterir.Bu özellik, sadece tasarim asamasinda tanimlanabilir.

Index: Üzerinde çalışılan nesnenin, ayni ad altında toplanan dizi nesnelerden hangisi oldugunu belirten sayiyi içerir.

Left: Nesnenin sola hizali olacagini belirler.

LocalPort: Yerel bilgisayarda kullanılan istasyonlari belirler.

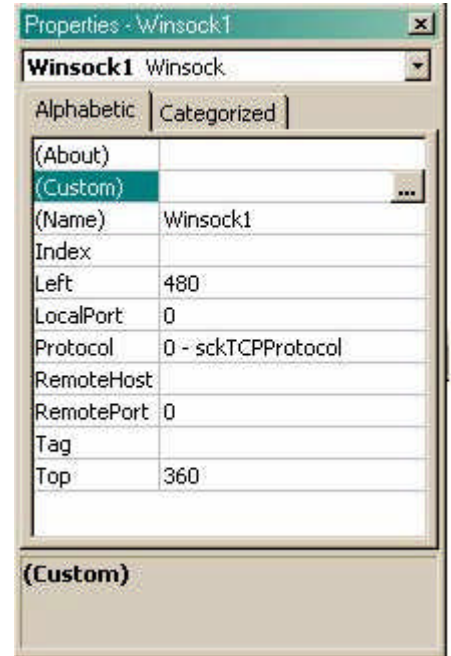
Protocol: Soket protokolunu belirler.

RemoteHost: Uzak yerdeki bir bilgisayarın ismini tanımlar.

RemotePort: Uzak yerdeki bilgisayara bağlanılmak için istasyon belirler.

Tag: Programınız için ihtiyaç duyulan herhangi bir ekstra veriyi depolar.

Top: Nesnenin iç üst kenari ile üst arasındaki mesafeyi belirler.



Winsock Fonksiyonlari

Winsock1.Bind: TCP bağlantıları için kullanılmak için LocalPort ve LocalIP belirtir. Eger çok protokol adaptörlerine sahip iseniz bu metod kullanılır.

Winsock1.BytesReceived: Veriyi geri almak için getdata metodunu kullanılır.

Winsock1.Close: Hem müşteri hem de hizmetçi uygulamaları için TCP bağlantısı veya dinleyiş socketini kapat.

Winsock1.Connect: Uzak bilgisayara bağlantıyı talep et.

Winsock1.Listen: Bu metod, TCP bağlantıları için sadece çalışır.

Winsock1.Object: Özellik veya metod, görülebilir temel yoluyla kontrole otomatik olarak uzatılan isminin aynısından özellik veya sahip olan kontrolün metoduna ilişkin özellikler.

Winsock1.Parent: Kontrol veya başka nesne veya koleksiyonu kapsayan form, nesne veya koleksiyon.

Winsock1.PeekData: PeekData hariç Getdataya benzer, girdi kuyrugundan veriyi kaldırmak için kullanılır. Bu metod, TCP bağlantıları için sadece çalışır.

Winsock1.SendData: Uzak bilgisayara veriyi gönderir.

Winsock1.State: Numaralandırılan tip olarak ifade edilen kontrolün durumunu belirler.

8-DOSYA ISLEMLERİ

Visual basic dilinde yazılan bazı programların kalıcı olarak saklanması gerekebilir. Örneğin okuldaki öğrencilerin kayıt listesi. Bu yüzden bilgilerin saklanması için visual basic dilinde dosyalama işlemleri kullanılmaktadır.

Visual basic dilinde kullanılan iki tip dosyalama türü vardır. Bunlar:

- Rasgele Erisimli Dosyalar
- Sirali Erisimli Dosyalar

Dosya Açma:

Her iki tür dosyalama türünde dosya açmak için kullanılan komut OPEN komutudur.

Kullanımı:

```
Open "Dosya Adi" [For Açma Modu] [Access Erisim Modu] [Lock] As  
[#] Dosyano [Len=kayituzunlugu]
```

Şimdi buradaki komutları tanıyalım:

- Dosya adi:Açacağımız dosyamızın adını tanımlamak için kullanılır.
- Açma modu:Açacağımız dosyamızın modunu belirtmek için kullanılır.
Bes adet dosya açma modumuz vardır:
 1. Random:Bu modda her kaydın uzunluğu sabittir ve sabit olduğu için dosyadaki herhangi bir kayda o kaydın numarasını vererek erişebiliriz.
 2. Binary:Bu moda dosya içindeki bir karaktere ulaşmak istediğimiz zaman karakter numarasını vererek ulaşabilmekteyiz.
 3. Input:Bu modda dosyamızı açtığımız zaman dosyamız okunmak üzere açılır.
 4. Output:Bu modda dosyamızı açtığımız zaman dosyamız yazmak için açılır.
 5. Append:Dosyamıza yeni kayıt eklemek için dosyamızı bu modda açarız. Eklenen yeni kayıt dosyamızın sonuna eklenir.
- Erisim Modu:Dosyamızı açarken hangi modda açacağımızı belirtmek için kullanırız. Üç tane parametresi vardır:
 1. Read:Dosyamız sadece okunmak için açılır. Yazma işlemi yapmak mümkün değildir.
 2. Write:Dosyamız sadece yazmak için açılır.
 3. Read Write:Dosyamızı hem yazma hem de okuma modunda açar.
- Lock:Anahtarlama işlemi demektir. İsteğe bağlı olarak dosya açılırken diğer programlara karşı erişim engellenebilir. Dört tane parametresi vardır:
 1. Shared:Açılmış olan bir dosyaya başka programlar tarafından okuma ve yama yapılabilir.
 2. Read:Açılmış olan dosya başka uygulamalar tarafından okunamaz,yazılabilir.

3. Write: Açılmış olan dosya yazma işlemine karşı kilitlenir.
 4. Read Write: Açılmış olan dosyamız hem yazmaya hem de okumaya karşı kilitlenir.
- Dosya No: Dosyamızı açarken vereceğimiz numarayı belirtir. 1-255 sayıları arasında numara verebiliriz. Dosyamızı çağırmak istediğimiz zaman verdiğimiz numarayı kullanırız.

Dosyamızı random, input ve binary modun da açarsak dosyamızı kapatmaya gerek kalmadan başka bir dosya numarası vererek tekrar açabiliriz. Fakat append ve output modlarında dosyamızı kapatmadan tekrar açmamız mümkün değildir.

- Kayıt Uzunluğu: Kayıt uzunluğumuz dosyamızı açma moduna göre değişmektedir. Fakat 3276'dan büyük olamaz. Binary modda kayıt uzunluğu yoktur. Dosyamız Input modda açılmissa her kaydın uzunluğu 128 karakter olarak kabul edilir. Dosyamıza yazdığımız kayıt bizim verdiğimiz numaradan büyükse hata oluşacaktır.

Dosyamız random modundan farklı bir modda açılmissa verdiğimiz kayıt uzunluğu sayısı karakter buffer modunu belirler. Verilmezse 512 kabul edilir. Buffer bir seferde okunacak ve yazılacak karakter sayısını belirtir. Dosyamızdan bir kayıt okunacağı zaman buffer boyutu kadar okuma yapılır.

Rasgele Erisimli Dosyalarda Okuma ve Yazma İşlemi:

Bu tür dosyalarda yani Random ve Binary modda açılan dosyalarda okuma işlemi Get komutu ile, yazma işlemi de Put komutuyla olmaktadır.

Kullanımı: Put[#]dosyano,[kayitno],degisken
 Get[#]dosyano,[kayitno],degisken]

Degisken: Kaydın alınacağı değişkeni belirtir.

Dosya No: Dosya içindeki kayıt numarasını belirtir. aynı zamanda Open komutuyla belirlen dosya numarasıdır.

Kayıt No: Yazılacak yada okunacak değişkenin dosya içindeki kayıt numarasını belirtir.

ÖRNEK: Dim a
 A="Günaydin"
 Put #1,2,a

Yukarıdaki örnekte "a" değişkenini bir numaralı dosyanın iki numaralı kaydına yazar. Kayıt numarası belirtmeseydik "a" değişkeni sıradaki kayda yazılırdı.

ÖRNEK: Dim a

```
Get #1,2,a
Msgbox(a)
```

Bu örnekte de bir numaralı dosyani iki numaralı kaydini okur.

Visual basic programlama dilinde properties içindeki degiskenleri direkt olarak dosyaya yazamayiz ve okuyamayiz. Bunun yerine su sekli kullaniriz:

```
a=text1.text      Bu örnekte Text1 içindeli deger "a" degiskenine
Put #1,2,a        aktarilir ve bir numaralı dosyanin iki numaralı
```

kaydindan okunur.

Sirali Erisimli Dosyalarda Okuma ve Yazma Islemi:

Bu tür dosyalarda yani Output ve Append modunda açılan dosyalarda yazma islemi Write ve Print komutlariyla olur.

Kullanimi:Write #dosyano,[degiskenler]

Dosya No:Yazilacak kaydın numarasini belirtir.

Degiskenler:Write komutunda dosyaya yazilacak degiskenler tirnak içinde ve araya virgöl konularak yazilir.

```
ÖRNEK:  Dim isim,yas
         Open "/deneme" for output as#1
         isim="Ferda Uca"
         yas="18"
         write #1,isim,yas
         isim="Ayse Ünal"
         yas="20"
         write #1,isim,yas
         close#1
```

```
Çıktisi: "Ferda Uca","18"
         "Ayse Ünal","20"
```

Print Komutu:

Kullanimi:Print #dosyano,[degisken formati]

Dosya No:Yazilacak kaydın dosya numarasini belirtir.

Degisken formati:Degisken formati ekrana çıktı yapıyormus gibi ekrana yazar.

```
ÖRNEK:  Dim isim,yas
         Open "/deneme" for output as #1
         isim="Ferda Uca"
         yas="18"
         write #1,isim,yas
         isim="Ayse Ünal"
         yas="20"
         write #1,isim,yas
```

close#

Çıktısı: Ferda Uca 18 Ayse Ünal 20

Input moddaki dosyalarda okuma işlemi Input ve Line input komutlarıyla olmaktadır.

Input:

Kullanımı: Input #dosyano,degisken listesi

Dosya No: Okuma yapılacak dosyanın numarasını belirtir.

Degisken Listesi: Dosyadan okuma yapıldığı zaman kayıtlar bu degiskene aktarılır.

Line Input:

Kullanımı: Line Input #dosyano,degisken

Dosya No: Okuma yapılacak dosyanın numarasını belirtir.

Degisken : Dosyadan okuma yapıldığı zaman kayıtlar bu degiskene aktarılır.

Input ve line input arasındaki farkı kısaca şöyle açıklayalım:

Input komutunda tirnak içinde yazılmış olan karakterler bir kayıt olarak kabul edilir. Tirnak içinde bir karakter bulunmuyorsa satır sonuna kadar olan karakterler bir kayıt olarak kabul edilir. Line input komutunda ise tirnak işaretleri dikkate alınmamakta ve her satır bir kayıt kabul edilmektedir.

Bir örnekle bunu daha iyi açıklayalım:

ÖRNEK: Deneme isimdeki dosyada şu kayıtlar olsun:

Input:

Line input:

Ferda Uca 18
"Ayse Ünal" 20

Ferda Uca 18
Ayse Ünal 20

Dim isim,yas
Open "/deneme" for input as #1
Input #1,isim,yas
Print "isim=",isim
Print "yas=",yas
Close #1

Dim isim,yas
Open "/deneme" for input as #1
Input #1,isim,yas
Print "isim=isim",isim
Print "yas=",yas
Close #1

Çıktısı: isim=Ferde Uca 18
yas= Ayse Ünal

isim=Ferde Uca 18
yas=Ayse Ünal 20

Örneklerden de anlaşıldığı gibi Line Input komutunda tirnak işaretlerinin bir anlamı yoktur. Input komutunda ise tirnaga kadar olan kısım bir kayıt kabul edilmektedir.

Input komutunun birde fonksiyon formatı vardır. Bu formatı kullandığımız zaman dosyadan istediğimiz sayıda karakter okuyabiliriz.

Kullanimi: Input(karakter sayısı,dosya no)

```
ÖRNEK: Dim d
        Open "/deneme.dat" for input as #1
        Do while NOT EOF(1)
        Print input(2,#1)
        Loop
        Close #1
```

Bu örnekte bir numaralı kayıttan iki karakter okur.

Dosyaları Kapatmak: her iki modda da açtığımız dosyaları kapatmak için CLOSE komutunu kullanırız.

Kullanimi: Close[#dosya no]

Dosya No: kapatılacak olan dosyamızın numarasının belirtir.

Dosya Kontrol Fonksiyonları:

EOF: Numarasi verilmiş olan dosyanın sonuna kadar kontrolünü yapar.

Kontrolü tamamlanan dosya TRUE değerini alır.

Kullanimi: EOF(dosyano)

LOF: Dosyanın byte olarak uzunluğunu bulmak için bu fonksiyondan yararlanılır.

Kullanimi: LOF(dosyano)

SEEK: Aktif olan kaydı değiştirmek için bu fonksiyondan yararlanılır. Yani numarası verilmiş olan dosyayı yeni konumuna tasir.

Kullanimi: SEEK #dosyano,yenikonum

Aktif gösterici konumunu görmek için bu komutun fonksiyon formatı kullanılır: SEEK(dosyano)

FreeFile: Açtığımız dosyalara kayıt numarası verirken, okuma veya düzeltme içinde aynı numarayla dosyamızı çağırıyorduk. Her verdiğimiz dosya numarasının sayısı farklı olacağı için dosya numarasını bulmak zor olabilir. Bu zorluğu engellemek yani dosya numarasını bulmak için FreeFile fonksiyonunu kullanırız.

```
Dim FreeFile
Dosya no=FreeFile
Open "dosyaadi" For "dosyamodu" As #dosyano
```

UYGULAMA: Öğrencilerin bilgilerini tutan bir menü oluşturalım: Bu menüde

- Kayıt no
- Adı Soyadı
- Numara
- Cinsiyet
- Okul

Microsoft Visual Basic 6.0

- Bölüm bilgileri bulunsun.

Bu menünün "kayıt", "arama", "düzeltme", "silme", "listeleme" modüllerini oluşturalım.

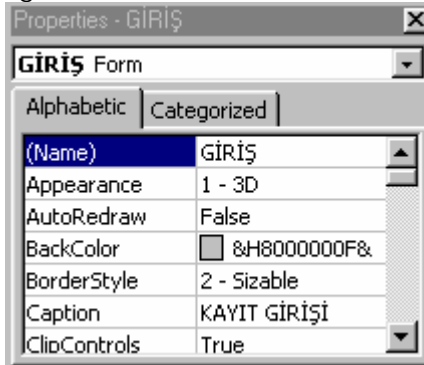
Örneğimizdeki bilgileri değişken olarak kullanmak amacıyla programımıza Project Add Modul seçeneği ile modül ekleyerek tanımlama kısmını string olarak hazırlarız.

```
Type kayıtbilgisi
adsoy As String * 30
adres As String * 30
cinsiyet As String * 5
okul As String * 10
bölüm As String * 10
End Type
Global tanımlar As kayıtbilgisi
```

Kayıt bilgilerinin her biri Tanım olarak geçmekte ve tüm form veya modüllerde geçerli olabilmesi için Global kısmında dosyalama adı verdim. Bu kısmı tanımladıktan sonra ilk modülümüz olan Kayıt modülümüzün görüntüsü aşağıdaki gibidir:

Textbox butonlarına kayıt no, adı soyadı, numara isimlerini verdim. Cinsiyet için Option, okul için Combobox ve bölüm için Listbox butonlarını seçtim. Formun başlığını Properties penceresinde Caption kısmında "KAYIT GİRİŞİ"

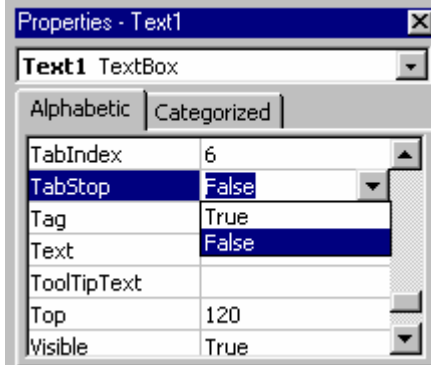
olarak;projenin adini yine ayni pencerede Name kismina "GIRIS" olarak degistirdim.



Simdi kodlama kisimimize geçebiliriz:

```
Private Sub Form_Load()  
Open "bilgi.dat" For Random As #1 Len = Len(tanımlar)  
kayıtno = LOF(1) / Len(tanımlar)  
kayıtno = kayıtno + 1  
Text1.Text = kayıtno  
End Sub
```

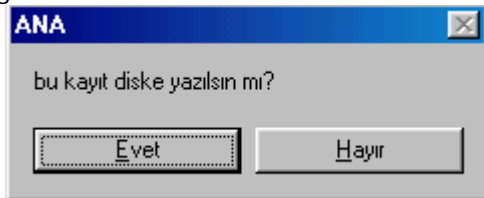
Dosyamizi Open komutu ile açariz adini ve modunu bu kisimda belirtiriz. Kayit numarasinin biz girmeden otomatik olarak artmasi için ve dosyadaki kayit sayisinin bulunmasi için yukaridaki komutlari kullaniriz. Kayit numarasini biz girmeden kendisi artacagi için Properties penceresinde Text1(kayit no) butonunun Tabstop özelligi False hale getirilir.



Kayit Butonu:

```
Private Sub Kayıt_Click()  
tanımlar.adsoyad = Text2.Text  
tanımlar.numara = Text3.Text  
If Option1.Value = True Then  
tanımlar.cinsiyet = Option1.Caption  
Else  
tanımlar.cinsiyet = Option2.Caption  
End If  
tanımlar.okul = Combo1.Text  
tanımlar.bölümü = List1.Text  
cvp = MsgBox("bu kayıt diske yazılsın mı?", 4)  
If cvp = 6 Then  
Put #1, kayıtno, tanımlar  
kayıtno = kayıtno + 1  
Text2.Text = ""  
Text3.Text = ""  
Combo1.Text = ""  
List1.Clear  
Text2.SetFocus  
Text1.Text = kayıtno  
End If  
End Sub
```

Textbox içindeki bilgiler tanımlar kısmına aktarılır ve Put deyimi ile dosyaya yazılır. Click olayında yazmamızın sebebi kayıt butonuna bastığımız anda kayıt işlemi yapılır. MsgBox komutu ile parantez içinde görmüş olduğumuz diyalog kutusu ekrana gelir ve 4 rakamı ile bize Evet/Hayır diyalog kutusunu gönderir.



Setfocus komutu ile kayıtno(text1box) kutusuna giriş yapmayacağımız için imlecimizin adı soyadı(text2box) kutusunda durmasını sağlar.

Temizle Butonu(Command2):

```
Private Sub Temizle_Click()  
Text2.Text = ""  
Text3.Text = ""  
Combo1.Text = ""  
List1.Clear  
Text2.SetFocus  
Text1.Text = kayıtno  
End Sub  
Textbox kutularının içeriklerini temizler.
```

Çıkis Butonu(Command3):

```
Private Sub Çıkış_Click()
Close #1
Unload Me
End Sub
```

Dosyamızı Close komutu ile kapatırız.

Birde diğer formlardan farklı olarak Form Active ve Combo1.Click kullanılmıstır.

Listeleme için Listbox kutusunun görüntüsünü görmek için:

```
Private Sub Combo1_Click()
If Combo1.Text = "İstanbul Üniversitesi" Then
List1.AddItem "İktisat"
List1.AddItem "Hukuk"
List1.AddItem "Matematik Öğretmenliği"
List1.AddItem "Türkçe Öğretmenliği"
End If
If Combo1.Text = "Sakarya Üniversitesi" Then
List1.AddItem "İnşaat"
List1.AddItem "İşletme"
List1.AddItem "Bilgisayar Programcılığı"
List1.AddItem "Bilgiyarlı Muhasebe"
End If
End Sub

Private Sub Form_Activate()
Text2.SetFocus
Combo1.AddItem "İstanbul Üniversitesi"
Combo1.AddItem "Sakarya Üniversitesi"
End Sub
```

Her formda tanımlayacağımız gibi Object liste kutusunda (general);Prog liste kutusunda (declarations) seçeneği seçiliyken kayıtno kısmını tanımlarız.

```
Dim kayıtno As Integer
```

ARAMA MODÜLÜ:

The image shows a Visual Basic form titled "Arama". It has a standard Windows-style title bar with minimize, maximize, and close buttons. The form contains the following controls:

- A label "Arama" at the top left.
- A text box labeled "Kayıt No".
- A text box labeled "Adı Soyadı".
- A text box labeled "Adres".
- A group box labeled "Cinsiyet" containing two radio buttons: "Kız" and "Erkek".
- A dropdown menu labeled "Okul".
- A list box labeled "List1" below the "Okul" dropdown.
- Two buttons at the bottom: "Temizle" and "Çıkış".

Formumuzun görüntüsünü nasıl hazırlayacağımızı daha önce anlatmistik. Formumuzun adini "ARAMA" olarak degistirdim. Sadece ilk Textbox(kayıt no)'ya giriş yapacağımız için diğer Textbox'ların özelliklerini Properties penceresinde Tabstop özelliğinden False durumuna getirdim.

```
Private Sub Form_Load()  
Open "bilgi.dat" For Random As #1 Len = Len(tanımlar)  
kayıtno = LOF(1) / Len(tanımlar)  
End Sub
```

Dosyamız vermiş olduğumuz isimle açılmakta ve dosyanın byte olarak uzunluğu kayıt uzunluğuna bölünerek kayıt numarası bulunmaktadır.

```

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
Get #1, Text1.Text, tanimlar
List1.Clear
Text2.Text = tanimlar.adsoyad
Text3.Text = tanimlar.numara
If tanimlar.cinsiyet = "kız" Then
Option2.Value = True
Else
Option1.Value = True
End If
Combo1.Text = tanimlar.okul
List1.AddItem tanimlar.bölümü
End If
End Sub

```

Kayıt numarasının içine yazarız. Çünkü kayıt numarası girildikten sonra aradığımız kayıt Enter tusuna bastıktan hemen sonra karşımıza çıkar. Keyascii=13 then satiri bunun için kullanılmaktadır. Get komutu ile kayıt numarası girildikten sonra okuma işlemi yapılır.

```

Private Sub Çıkış_Click()
Close #1
Unload Me
End Sub

Private Sub Temizle_Click()
Text2.Text = ""
Text3.Text = ""
Combo1.Text = ""
List1.Clear
Text2.SetFocus
Text1.Text = kayıtno
End Sub

```

DÜZELTME MODÜLÜ:

Diger formlarda yaptigimiz gibi formumuzun adini Düzeltme olarak degistiririz. Ekran görüntümüz ise aynı,sadece Command butonlarımız farklıdır.

```
Private Sub Form_Load()  
Open "bilgi.dat" For Random As 1 Len = Len(tanımlar)  
kayıtno = LOF(1) / Len(tanımlar)  
End Sub
```

Arama modülünde olduğu gibi Lof fonksiyonu ile dosyanın byte olarak uzunluğu bulunur ve kayıt uzunluğuna bölünerek(len) kayıt numarası bulunur.

Text1(Kayıt No):


```

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
Get #1, Text1.Text, tanımlar
List1.Clear
Text2.Text = tanımlar.adsoyad
Text3.Text = tanımlar.numara
If tanımlar.cinsiyet = "kız" Then
Option2.Value = True
Else
Option1.Value = True
End If
Combo1.Text = tanımlar.okul
List1.AddItem tanımlar.bölümü
End If
End Sub

```

Kayıt numarası girildikten sonra Enter tusuna basıldığında düzeltilecek kayıt ekrana gelir ve yeni kayıt yazıldıktan sonra kayıt butonuna basılır.

Kayıt Butonu(Command1):

```

Private Sub Command1_Click()
seçim = MsgBox("kaydın yeni şekli yazılsın mı?", 4)
If seçim = 6 Then
tanımlar.adsoyad = Text2.Text
tanımlar.numara = Text3.Text
If Option1.Value = True Then
tanımlar.cinsiyet = Option1.Caption
Else
tanımlar.cinsiyet = Option2.Caption
End If
tanımlar.okul = Combo1.Text
tanımlar.bölümü = List1.Text
kayıtno = Text1.Text
Put #1, kayıtno, tanımlar
Text2.Text = ""
Text3.Text = ""
Combo1.Text = ""
List1.Clear
Text1.Text = ""
End If
End Sub

```

Kaydın yeni şekli için Textbox içeriklerini tanımlar kısmına yollar ve kaydın yeni halini dosyaya yazılır.

Vazgeç Butonu(Command2):

Microsoft Visual Basic 6.0

```
Private Sub Command2_Click()  
cvp = MsgBox("düzeltilecek başka kayıt var mı?", 4)  
If cvp = 6 Then  
Text1.Text = ""  
Text2.Text = ""  
Text3.Text = ""  
Combo1.Text = ""  
List1.Clear  
Text1.SetFocus  
Else  
End If  
End Sub
```

Düzeltilmek istedigimiz baska bir kayıt varsa msgbox komutu ile ekrana gelen diyolog kutusundan Evet dersek Textbox'larin içerikleri silinir. Hayir dersek End deyimi ile çalışma sonra erer.

Çıkis Butonu(Command3):

```
Private Sub Çıkış_Click()  
Close #1  
Unload Me  
End Sub
```

SİLME MODÜLÜ:

The screenshot shows a Windows-style window titled "Silme". Inside the window is a frame labeled "Frame1". Within this frame, there are several controls: a text box for "Kayıt No", a text box for "Adı Soyadı", a text box for "Adres", a "Cinsiyet" section with two radio buttons labeled "Kız" and "Erkek", a dropdown menu for "Okul", and a list box for "Bölüm" which currently displays "List1". At the bottom of the frame, there are three buttons: "Sil", "Temizle", and "Çıkış".

Formumuzun basligini Silme olarak degistiririz.

```
Private Sub Form_Load()
Open "bilgi.dat" For Random As 1 Len = Len(tanımlar)
kayitno = LOF(1) / Len(tanımlar)
End Sub
```

Silinecek dosya açılır ve kayıt numarası bulunur.

Silme Butonu(Command1):

```
Private Sub SİLME_Click()
c = MsgBox("silme istediğinizden eminmisiniz?", 4)
If c = 6 Then
tanımlar.adsoyad = ""
tanımlar.numara = ""
tanımlar.cinsiyet = ""
tanımlar.okul = ""
tanımlar.bölümü = ""
Put #1, Text1.Text, tanımlar
Text2.Text = ""
Text3.Text = ""
Combo1.Text = ""
List1.Clear
Text1.Text = ""
Text1.SetFocus
End If
End Sub
```

Msgbox ile gelen mesaj kutusuna evet dersek Textbox içerikleri temizlenir ve dosyaya yazılır. Setfocus ile imlecimiz tekrar kayitno(text1box) kutusuna konumlandırılır.

Temizle ve çıkis butonlarının kodlari diger modüllerin kodlari ile aynidir.

Kayıt No(Tetxt1box):

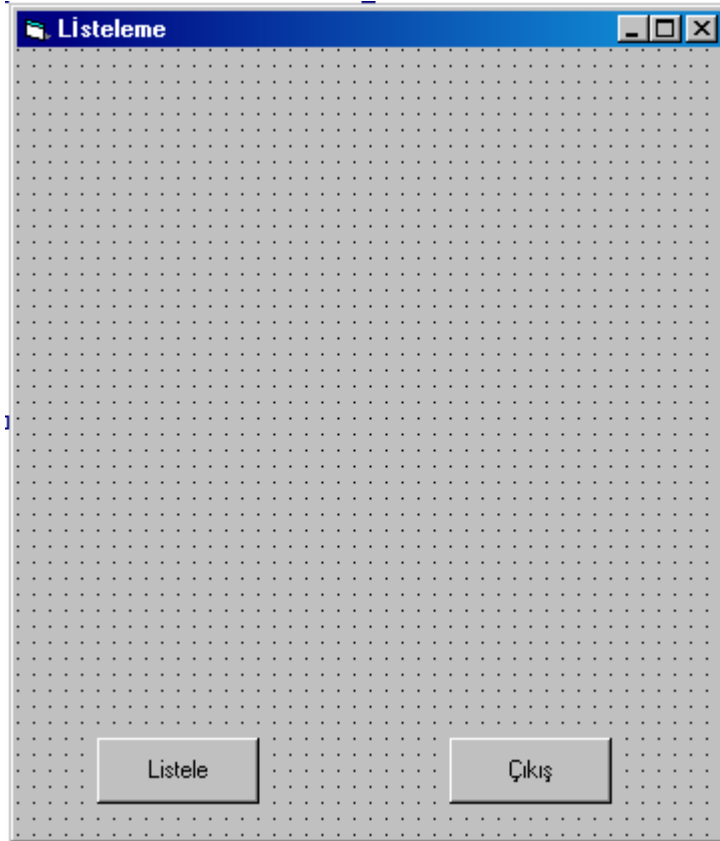
```
Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
Get #1, Text1.Text, tanımlar
List1.Clear
Text2.Text = tanımlar.adsoyad
Text3.Text = tanımlar.numara
If tanımlar.cinsiyet = "kız" Then
Option2.Value = True
Else
Option1.Value = True
End If
Combo1.Text = tanımlar.okul
List1.AddItem tanımlar.bölümü
End If
End Sub
```

Silinecek kayıt numarası girildikten sonra Enter tusuna basılınca

Microsoft Visual Basic 6.0

silinecek kayıt dosyadan okunur.

LISTELEME MODÜLÜ:



Listeleme modülünde sadece Command butonlari kullanılmaktadır. Command1 butonuna tıkladığımız zaman o ana kadar kayıt edilmiş veriler listeli bir biçimde ekrana gelir.

Form Load :

```
Private Sub Form_Load()  
Open "bilgi.dat" For Random As 1 Len = Len(tanımlar)  
kayıtno = LOF(1) / Len(tanımlar)  
kayıtno = kayıtno + 1  
End Sub
```

Listelenecek dosya açılır. Kayıt numarası bulunur ve her seferinde bir artar.

Listele Butonu(Command1):

```

Private Sub Command1_Click()
Print: Print
Print Tab(2); "adsoyad"; Tab(20); "numara"; Tab(35); "cinsiyet"; T
Print String(90, "***")
For i = 1 To kayıtno
a = a + 1
Get #1, a, tanımlar
If EOF(1) Then GoTo a1
Print Tab(2); tanımlar.adsoyad; Tab(20); tanımlar.numara; Tab(35);
Next i
a1:
Print String(90, "***")
End Sub

```

1'den başlayarak kayıt numarasına kadar bir döngü kurulur ve her seferinde bir artarak devam eder. Dosya sonuna(EOF) kadar listelenecek kayıt var mı diye kontrol edilir. Listelenecek kayıt varsa döngü durur ve ekrana listeli bir şekilde kaydı yazar.

Print string ifadesi kayıtlar arasına ile 90 tane "*" isareti konulmak istenmiştir.

Çıkış Butonu(Command1):

```

Private Sub Command2_Click()
Close #1
Unload Me
End Sub

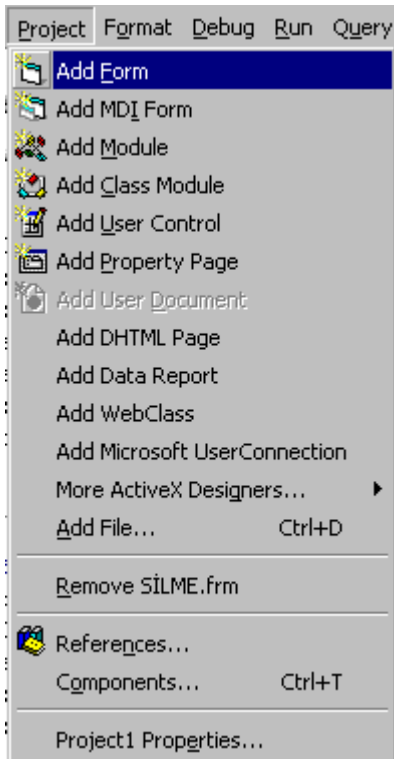
```

Burada unutulmaması gereken bir nokta da formun içinde (general) (declarations) yani kayıt numarasının tanımlandığı kısımda listeleme_click içinde kullanılan değişkenleri tanımlamaktır.

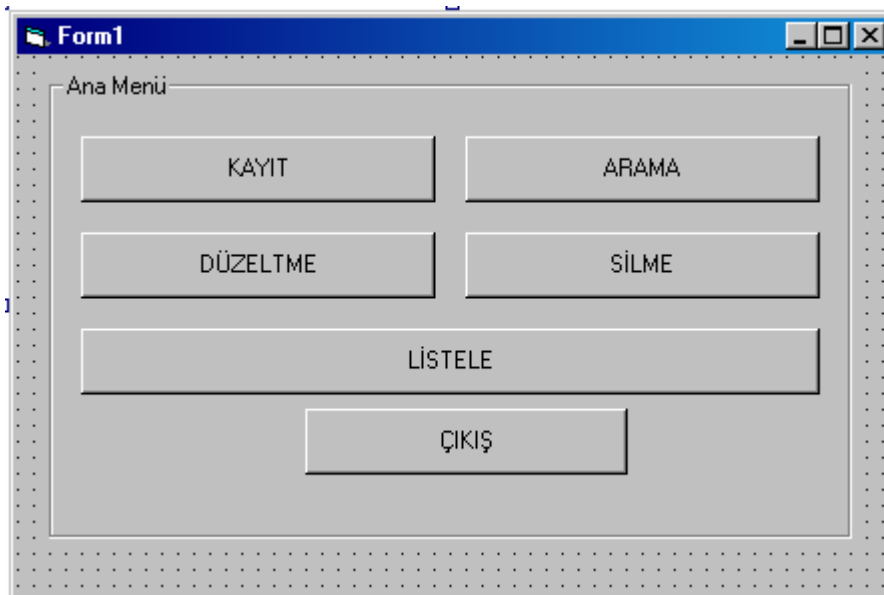
(General)	(Declarations)
<pre> Dim a, i As Integer Dim kayıtno As Integer </pre>	

Buraya kadar yaptıklarımızı tek bir menü altında toplamak istersek: Project penceresinden Add Form seçeneği ile yeni bir form ekleriz

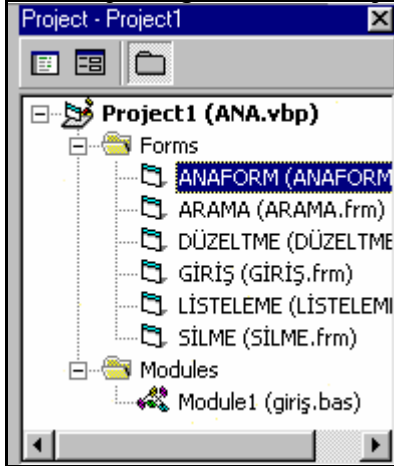
Microsoft Visual Basic 6.0



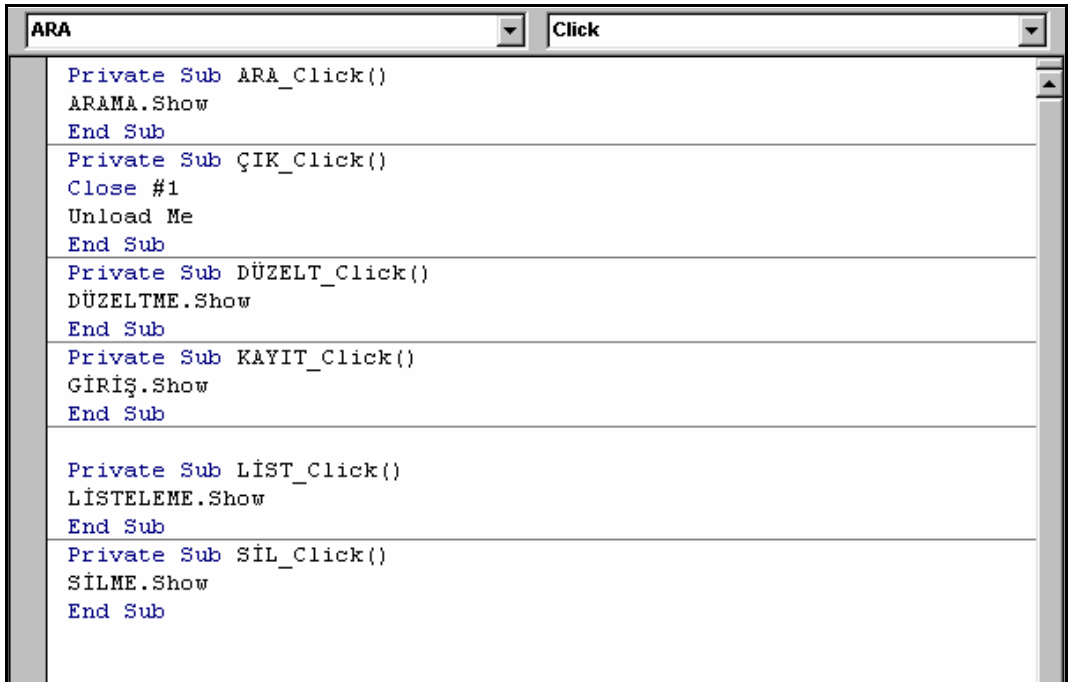
Yeni formumuzun görüntüsünü aşağıdaki gibi sadece Command butonları kullanarak hazırlarız:



Formumuzun adini Ana Menü olarak degistirir ve Project Menüsünden Add File komutuyla diger modul dosyalarini projeye dahil ederiz.



Kayıt, arama, düzeltme, silme ve listeleme butonlarının Properties penceresinde Caption kısmında isimlerini verdikten sonra her butonun Click olayına isimleriyle birlikte Show kısımlarını yazarız.



Böylece tek bir menü altında tüm işlemlerimizi görüntülemiş oluruz.

The image shows a Microsoft Visual Basic 6.0 application window titled 'Form1'. Inside the window, there is a menu bar labeled 'Ana Menü' (Main Menu). Below the menu bar, there are three buttons: 'KAYIT' (Registration), 'DÜZELTME' (Correction), and an empty button. A dialog box titled 'Kayit' (Registration) is open in front of the main window. The dialog box contains a frame labeled 'Frame1' with the following fields and controls:

- 'Kayıt No' (Registration No): A text box containing the number '8'.
- 'Adı Soyadı' (Name Surname): A text box.
- 'Adres' (Address): A text box.
- 'Cinsiyet' (Gender): Two radio buttons labeled 'Kız' (Girl) and 'Erkek' (Boy).
- 'Okul' (School): A dropdown menu.
- 'Bölüm' (Department): A text box.

At the bottom of the dialog box, there are three buttons: 'Kayıt' (Registration), 'Temizle' (Clear), and 'Çıkış' (Exit).

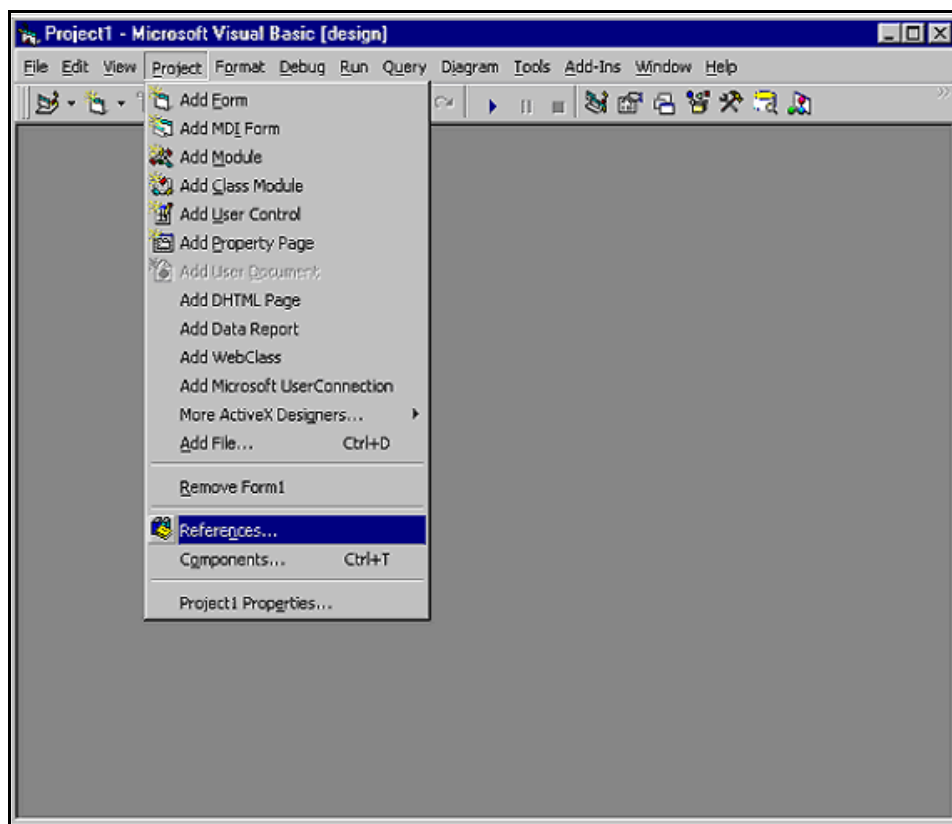
9-PROGRAM KODU YAZARAK VERİ TABANI DOSYASI HAZIRLAMAK

Kitabımızın daha önceki bolumlerinde anlatıldığı gibi Visual basic ile birlikte verilen Visual Data Manager veya baska bir veri tabani programi ile hazirlanan veri tabani dosyalari üzerinde data kontrolü yardimiyla her türlü islemi yapmak mümkündür.Bu bölümde öncelikle visual basic projesi dahilinde çalışma aninda nasıl MDB uzantili veri tabani dosyasinin hazirlandigi konusunda bilgi verilecektir.Daha sonra Data kontrolunden yararlanmak yerine program kodu yazarak veri tabani dosyasi üzerinde nasıl işlem yapıldigi konusunda bilgi verilecektir.Hemen eklemek gerekirse bu bölümde anlatılan özelliklerin çoğunlugu visual basic'in profesyonel sürümü tarafından desteklenmektedir.

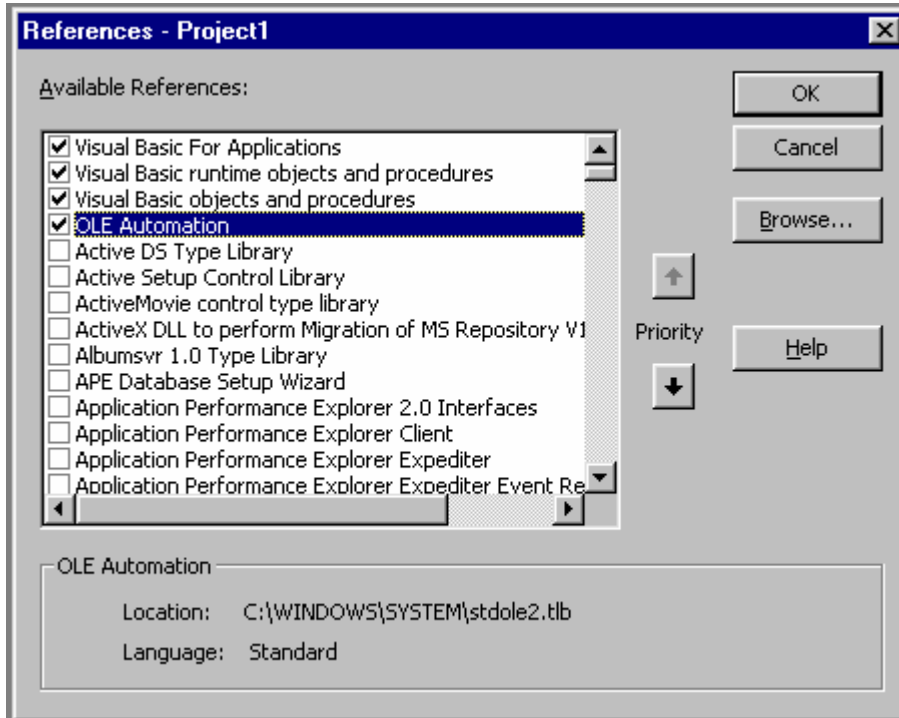
Önce dos uyumlu bir veri tabani programlama dili olan Clipper'da veri tabani dosyalarinin nasıl açıldığı konusunda bilgi verecegim.Clipper'da genel alışkanlıkla kullanılmak istenen DBF uzantili veri tabani dosyasi önce harddiskte araştırılır.Ardından dosya varsa açılır.Eger söz konusu program ilk kez çalıştırılıyor ve dolayısıyla söz konusu dosya harddiskte henüz yoksa oluşturulur.Dosya oluşturma işlemi temelde iki aşamada yapılır.Önce henüz içeriği bos olan bir dosya hazirlanır.Ardından dosyada yer alacak bütün alanlar özellikleri ile birlikte sıra ile dosyaya ilgili deyimler yardimiyla yazılır.

Benzer durum Visual Basic te de söz konusudur.Bu nedenle önce program kodu ile çalışma lanında Visual Basic uygulaması dahilinde nasıl MDB uzantili veri tabani dosyasi oluşturulduğu konusu üzerinde duracağız.Hemen hatırlatmak istiyorum: MDB uzantili veri tabani dosyalari DBF uzantili dosyalardan oldukça farklıdır.DBF uzantili dosya kavramı, Access formatındaki mdb dosyalarında Table ile karşılanmaktadır.mdb dosyasında istenen sayıda tablo yer alabilmektedir.

Visual Basic uygulaması dahilinde çalışma anında yeni bir Database nesnesi yani MDB uzantili veri tabani dosyasi oluşturmak için visual basic'in Create Database() fonksiyonundan yararlanılır.Visual Basic'in 3.0 ve 4.0 versiyonlarında çalışma alanında Create database() fonksiyonu ile veri tabani dosyasi oluşturmak için önceden herhangi bir hazırlık yapmaya gerek yoktu.Ancak 5.0 versiyonda Create Database() gibi fonksiyonları kullanabilmek için daha önceden Project menüsünden References komutunu verip ekrana References komutunu verip ekrana References diyalog kutusunu getirmek gerekir.



Project menüsünden References komutu verildiği zaman ekrana gelen diyalog kutusunda, üzerinde çalıştığınız Visual Basic proje dahilinde kullanılacak kitaplıklar veya imkanlar seçilmektedir. Yeni bir Visual Basic projesi hazırladığınızda references diyalog kutusunda listelenen seçeneklerin ilk 4'ü seçili durumdadır.



Micrasoft DAO 3.5 Object Library üzerinde çalışan projeye References diyalog kutusu aracılığı ile dahil edilmediği sürece Create Database() gibi fonksiyonları kullanmak mümkün değildir. Eğer üzerinde çalıştığınız projede Visual Basic'in nesne bağlama ve katma özelliklerinden yararlanamıyorsanız References diyalog OLE Automation seçeneğini iptal edebilirsiniz.

CreateDatabase() fonksiyonu dışarıdan ikisi seçimli olan toplam 3 parametre almaktadır. Olusturulacak veri tabanı dosyasının adını içeren ilk parametrenin kullanılması zorunludur..

```
Dosya=Create database("TEST.MDB")
```

Bu program satırı ile hard diskteki adı TEST:MDB ve program içindeki adı Dosya olan bir veri tabanı nesnesi oluşturulur. Yeni oluşturulan dosya adındaki database nesnesinin ayrıca SET deyimi ile set edilmesi gerekir. Bu nedenle yukarıda verilen program satırı;

```
SET Dosya=CreateDatabase("TEST:MDB")
```

Seklinde değiştirilmelidir. Bütün fonksiyonlar gibi Create Database() fonksiyonu da geriye bir değer döndürür. Burada geriye döndürülen değeryeni bir veri

Microsoft Visual Basic 6.0

tabanı nesnesidir. Bu nesnenin bir kopyası program çalıştığı sürece Database özellikli bir değişkende saklanacağı için bu değişkenin daha önceden tanımlanması gerekir. Database nesnesinin içeriğini program çalıştığı sürece bellekte saklayacak değişken Database bildirisiyle tanımlanır.

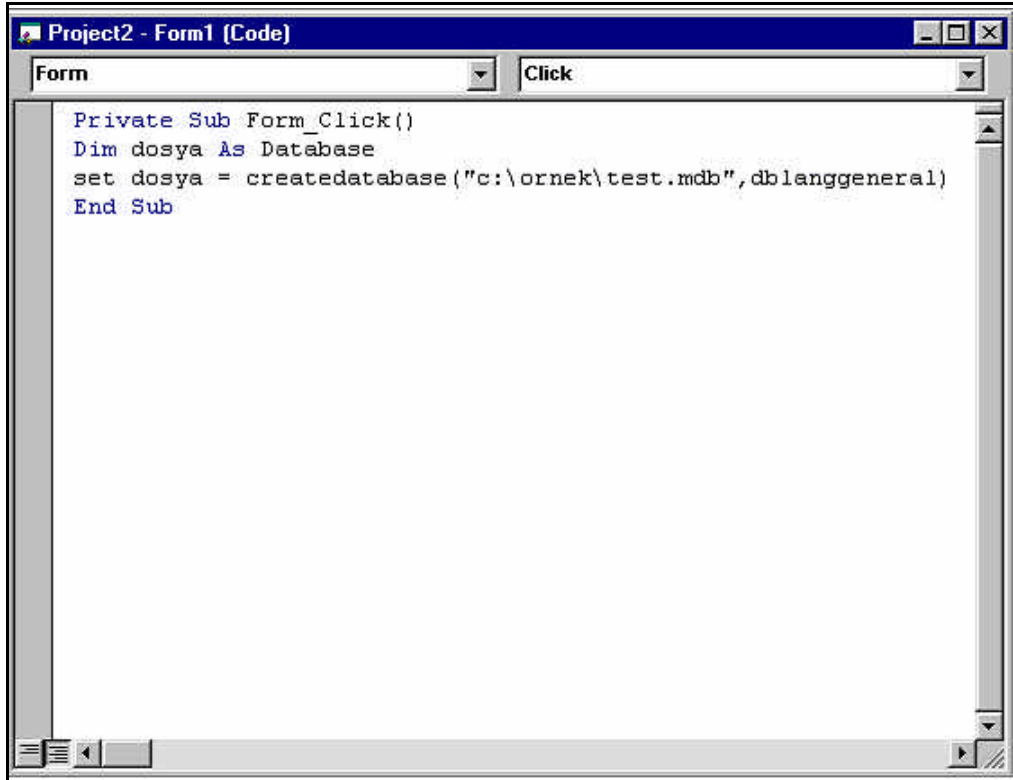
```
Dim Dosya As Database  
SET Dosya=Create Database("test.mdb")
```

Oluşturulacak veri tabanı dosyasının adından önce istenirse dosyanın yazılacağı sürücü ve dizin adı belirtilebilir. CreateDatabase() fonksiyonuna verilen 2. parametrede dil veya ülke kodu belirtilir.

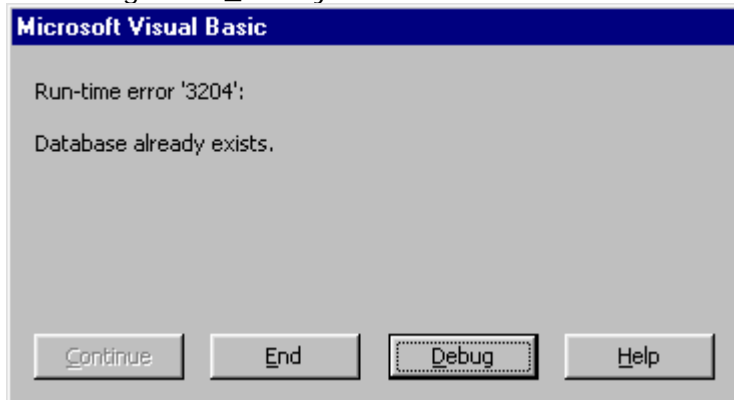
```
Dim Dosya As Database  
Set Dosya=Create Database("c:\ORNEK\TEST.MDB",dbLangGeneral)
```

Create Database() fonksiyonu ile Access formatında MDB uzantılı veri tabanı dosyası oluşturulurken Visual Basic çok sayıda sabit bilgiden yararlanır. Bu sabit bilgilerin isiginda dosya hazırlanır. Bu sabit bilgiler Visual Basic'in 3.0 versiyonunda Visual Basic ile birlikte verilen DATACONS.TXT adlı dosyada bulunuyordu. Bu nedenle Visual Basic'in 3.0 versiyonu dahilinde üzerinde çalışılan projeye dahil edilmesi gerekiyordu. Ancak Visual Basic'in 4.0 ve 5.0 versiyonunda buna gerek yoktur. Create Database() fonksiyonu ile MDB uzantılı veri tabanı dosyası hazırlanırken gerek duyulan sabitler projeye otomatik olarak dahil edilmektedir.

Yukarıda verilen iki satırlık program kodu işletilecek olunursa, belirtilen sürücü ve klasörde TEST.MDB adında Access formatında bir veri tabanı dosyası oluşturulur. Yeni bir veri tabanı dosyası oluşturmak üzere hazırladığım 2 satırlık program kodlarını üzerlerinde çalıştığım örnek projedeki formun Form_Click yordamına dahil ettim.



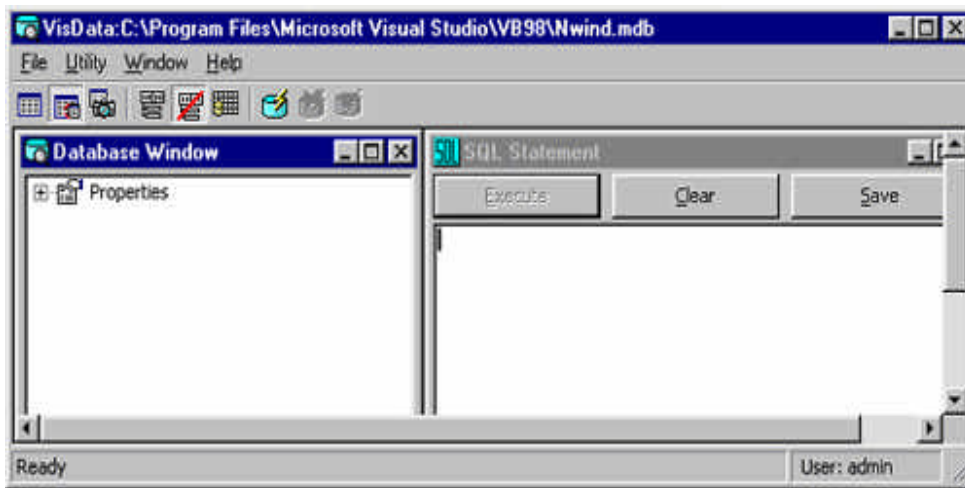
Form_Click yordami bu sekilde hazirlanan proje baslatilip çalıřma anında formun üzerinde tıklama yapılırsa söz konusu veri tabanı dosyası oluşturulur. Harddiskte bulunan bir veri tabanı dosyası Create Database() fonksiyonu ile tekrar oluşturulmak istenirse hata meydana gelir. Ařagıda verilen ekran görüntüsünü TEST.MDB dosyasını oluşturduktan sonra çalıřma anında formun üzerinde ikinci kez tıklama yaparak Create Database() fonksiyonunun kullanıldığı Form_Click yordamını ikinci kez işletildikten sonra aldım.



Microsoft Visual Basic 6.0

Bu şekilde oluşturulan TEST.MDB dosyası program çalıştığı sürece program dahilinde artık bir nesnedir. Bu Database tipindeki nesnenin de kendine özgü Properties veya özellikleri bulunur. Nasıl ki daha önce hakkında bilgi verilen değişik nesneler üzerinde işlem yapmak için değişik amaç ve işleve sahip methodlar kullanılıyorsa Database nesnesi üzerinde etkili olan methodlar vardır.

Bu şekilde oluşturulan bu TEST.MDB adlı veri tabanı dosyasının içeriği henüz bostur. Bu dosyada bilgi kaydedebilecek tablo henüz yoktur. Bunu göstermek için Visual Basic ile birlikte verilen Visual Data Manager uygulamasını çalıştırıp hazırladığım TEST.MDB dosyasını açınca aşağıda verilen ekran görüntüsünü elde ettim.



Olusturdugum TEST.MDB adındaki veri tabanı dosyasına şimdi bir Tablo ekleyeceğim. Önce bu amaçla hazırlanan yordamı verelim. Daha sonra yordamda kullanılan deyim ve fonksiyonlar hakkında bilgi verelim. Yukarıdaki sayfada verilen 2 satırlık program kodu ile olusturdugum Access uyumlu veri tabanı dosyasına tablo eklemek için önce yeni bir proje olusturalım. Veri tabanı dosyasına tablo eklemeye kullanılan program satırlarını projenin formuna ait Form_Click yordamına dahil edelim.

```
Sub Form_Click()  
  
Dim Dosya As Database  
Dim Tablo As New Table Del  
Dim Idx As New Index  
Dim Alan1 As New Field  
Dim Alan2 As New Field  
Dim Alan3 As New Field  
Dim Alan4 As New Field
```

```

Dim Alan5 As New Field
Dim Alan6 As New Field
Set Dosya=OpenDatabase("C:\ORNEK\TEST.MDB")
Tablo.Name="ADRES"
Alan1.Name="Ad"
Alan1.Type=10
Alan1.Size=15
Tablo.Fields.Append Alan1
Alan2.Name="Soyad"
Alan2.Type=10
Alan2.Size=15
Tablo.Fields.Append Alan2
Alan3.Name="Adres"
Alan3.Type=10
Alan3.Size=30
Tablo.Fields.Append Alan3
Alan4.Name="Tel"
Alan4.Type=10
Alan4.Size=10
Tablo.Fields.Append Alan4
Alan5.Name="Sehir"
Alan5.Type=10
Alan5.Size=10
Tablo.Fields.Append Alan5
Idx.Name="primaryKey"
Idx.Unique=True
Idx.Primary=True
Idx.Fields="Soyad"
Tablo.Indexes.Append Idx
Dosya.TableDefs.Append Tablo
End Sub

```

Şimdi mevcut bir veri tabanı dosyasına tablo eklemek amacıyla kullanılan bu yordamı kısım kısım açıklayalım. Yukarıdaki sayfalarda belirtildiği gibi veri tabanı dosyaları ile ilgili olarak kullanılan her nesne için önceden değişken tanımlama işlemi yapılmalıdır. Bunun için yordamın en başında veri tabanı dosyası üzerinde işlem yapmada kullanılacak bütün nesneler için önceden değişken tanımlaması yapıldı. Aşağıda tekrar verilen satırlarda Database, Table, Index ve Field nesneleri için değişken tanımlama işlemleri yapılmaktadır. Veri tabanı dosyası nesnesi için değişken tanımlanırken Dim bildirisi deyimi ile birlikte Database bildirisi deyimi kullanılmaktadır.

```

Dim Dosya As Database
Dim Tablo As New TableDef
Dim Idx As New Index
Dim Alan1 As New Field

```

Microsoft Visual Basic 6.0

```
Dim Alan2 As New Field  
Dim Alan3 As New Field  
Dim Alan4 As New Field  
Dim Alan5 As New Field
```

Benzer şekilde Table nesnesi için değişken tanımlamada TableDef bildiri deyimi kullanılmaktadır. Bizim örnekte veri tabanı dosyasına Tablo eklemek amacıyla kullanacağımız degiskene Tablo adını verelim. Eğer veri tabanı dosyasına birden fazla Table nesnesi içeren veri tabanı dosyası üzerinde işlem yapmak istediyseniz TableDef bildiri deyimi ile birden fazla Table tipli değişken tanımlamamız gerekirdi. Aslında Dim Tablo As New TableDef program satırı ile yapılan değişken tanımlama, bildiğimiz değişken tanımlama işleminden oldukça farklı işleme sahiptir.

TableDef bildiri deyimi ile değişken tanımlamadan öte başlangıçta herhangi bir özelliği olmayan bir Table nesnesi tanımlanmaktadır. Tanımlanan nesnenin yeni olduğunu belirtmek için TableDef bildiri deyiminden önce New bildiri deyimi kullanılmaktadır. Bu şekilde yapılan değişken tanımlama işlemine Nesne Tanımlama denilebilir.

TabloDef bildiri deyimi ile Table nesnesi için değişken tanımlanması yapılan program satirından sonraki satırda Index bildiri deyimi ile Index nesnesi için değişken tanımlaması yapılmaktadır. Veri tabanı programlarının en önemli özelliklerinin başında Index kullanımı konusunda sağladıkları kolaylıklardır. Bunun için veri tabanı dosyası ve Table nesnesi söz konusu olunca, Index dosyasından yararlanmamak düşünülemez. Veri tabanı dosyasında bulunan her Table için ayrı indexleme yapılır. Hazırladığım örnek veri tabanı dosyasındaki tabloda bir tek index kullanacağım için bir tek Index tipli değişken tanımladım.

Veri tabanı dosyası hazırlanırken en basta dosyada saklanılacak bilgiler veya alanların adları, tipleri ve uzunlukları belirlenir. Hard diskteki adı TEST:MDB olan veri tabanı dosyasına ekleyeceğimiz tabloda 5 alanın olmasını istediğim için 5 adet Field(alan) nesnesi hazırlamamız gerekiyor. Bunun için yordamın başında her Field nesnesi için Field tipli bir değişken tanımlaması yapılmalıdır. Field tipli değişken tanımlamak için Field Bildiri deyiminden yararlanılır.

Yordamda kullanılacak nesneler için değişken tanımlama işlemi yapıldıktan sonra ilk olarak kendisine Table nesnesi eklenecek veri tabanı dosyasının açılması gerekir. Hard diskteki adı TEST:MDB olan veri tabanı dosyası üzerinde işlem yapabilmek için dosyanın OpenDatabase() fonksiyonu ile açılması gerekir. OpenDatabase() fonksiyonu dışarıdan parametre olarak üç tanesi seçilebilir olan toplam 4 bilgi almaktadır. İlk parametrede açılacak dosyanın adı belirtilir.


```
Set Dosya=OpenDatabase("C:\ORNEK\TEST.MDB")
```

Bu program satiri ile hard diskteki adi TEST:MDB olan Access formatindaki veri tabani dosyasi açilip Database tipli"Dosya"degiskenine SET edilir.OpenDatabase() fonksiyonu verilmeyen parametreler için varsayılan degerleri kullanilir.Eger açılmak istenen dosya çok kullanicili bir ortamda kullaniliyorve baska kullanicilarin dosyaya erisimi engellemek isteniyorsa OpenDatabase() fonksiyonuna 2.parametre olarak "True"degeri verilmelidir.

```
Set Dosya=OpenDatabase("C:\ORNEK\TEST.MDB",true)
```

Açmak istediginiz dosya üzerinde islem yapılmasını istemiyorsanız dosyayı ReadOnly modunda açmalısınız.Dosyayı ReadOnly modunda açabilmek için 3.parametreden yararlanılır.OpenDatabase() fonksiyonuna 3.parametre olarak"true"mantıksal degeri verilirse dosyada degisiklik yapılamaz.Daha önceki konularda belirtildiği gibi Visual Basic dahilinde baska veri tabani programlari ile hazırlanan veri tabani dosyalarını açip üzerinde islem yapma imkanı vardır.Eger açılmak istenen dosya MDB uzantılı Access formatlı dosya değilse,söz konusu dosyanın tipi 4.parametrede belirtilmelidir.

```
Set Dosya=OpenDatabase("C:\ORNEK\TEST.MDB",True,False,Paradox)
```

Örnek yordamının değısken tanımlanması yapılan satırlarında Dim Tablo As New TableDef ile yeni bir tablo nesnesi hazırlamistik.Daha önceki konularda belirtildiği gibi yeni bir nesne ilk kez hazırlanırken varsayım olarak bir takım özelliklere sahip olur.Daha sonradan söz konusu nesnenin özelliklerinde degisiklik yapılabiliniyordu.Bunun için önce nesnenin adı,ardından nesnenin değıstirilmek istenen özelliğinin adı belirtiliyordu.Veritabanı dosyası için hazırladığımız Table tipli nesnenin henüz adı belli değildir.Aşağıda verilen program satiri ile Table nesnesinin adı belirlenmektedir.

```
Tablo.Name="ADRES"
```

Table nesnesine ADRES adını verelim.Table nesnesinin adı belirlendikten sonra sıra,bu nesneye Field veya alan eklemeye gelir.Daha önceden ADRES adlı Table nesnesinde 5 adet alanın bulunmasına karar vermiş ve her an için Field tipi değısken tanımlama işlemi yapmış veya 5 adet yeni Field nesnesi hazırlamistik."Tablo" adlı Table nesnesinin adını Table nesnesine ait Name değıskenine nasıl aktardıysak,benzer işlemi bütün Field nesneleri için yapmak gerekir.Ancak bir Field veya alanın adından baska,alanın tipinin ve uzunluğunun mutlaka belirtilmesi gerekmektedir.Aşağıda verilen 3 program satiri ile bir alanın adı,tipi ve uzunluğu belirtilmektedir.

```
Alan1.Name="ad"  
Alan1.Size=15  
Alan1.type=10
```

İlk bakışta bu 3 program satırından ilk ikisi sanırım hemen anlaşılmıştır. Ancak alanın veya Field nesnesinin tipini belirtmeye yarayan Type değişkenine aktarılan sayısal değer hakkında bilgi vermek gerekiyor. Bu bölümün başında Visual Basic dahilinde Data kontrolünden yararlanmadan veri tabanı dosyaları üzerinde işlem yaparken bazı sabitlerden yararlandığını belirttim. Bu sabitlerin bir kısmını aşağıda verilen ekran görüntüsünde görebilirsiniz.

Bu ekran görüntüsüne dikkatlice bakılacak olunursa, yukarıdaki sayfada Field nesnesine ait Type değişkenine aktarılan 10 sayısal değerinin gerçekte Text tipindeki verileri temsil ettiği tespit edilir. Field nesnesine ait Type değişkenine aktarılan 10 sayısal değerinin Text tipindeki verileri temsil ettiğini ezberle bilmiyorsanız, 10 yerine Type değişkenine dbText değerini aktarabilirsiniz.

Bu program satırları ile alanın veya Field nesnesini, Table nesnesine dahil etmeye gelir. Çünkü şimdiye kadar verilen program satırları ile özellikleri belirlenen Field nesnesinin Table nesnesi ile bir ilişkisi henüz yoktur. Önce New Field bildirisiyle oluşturulup daha sonra kendine ait Name, Type ve Size değişkenleri ile özellikleri belirlenen bir Field nesnesini Table nesnesine dahil etmek veya Field nesnesinden Table nesnesi içinde yararlanmak için Append özelliği söz konusu nesneye ait Properties penceresinde görülebiliyordu.

New Field bildirisiyle oluşturulan Field nesnesinin özelliklerini saklamada kullanılan toplam 8 değişken otomatik olarak tanımlanır. Bunlar Attributes, CollatingOrder, Name, OrdinalPosition, Size, SourceField, SourceTable ve Type. Field nesnesine ait bu değişkenlerin bazılarının Data kontrolüne yönelik olduğunu hemen hatırlatmak istiyorum.

Hazırlanan Field nesnesinin Append bildirisiyle Table nesnesine nasıl eklendiği konusunda bilgi vermeden önce Table nesnesinin özellikleri veya Table nesnesine ait değişkenler hakkında kısa bilgi verelim. Nasilki yeni oluşturulan bir Field nesnesinin gerek duyduğu değişkenler Visual Basic tarafından otomatik olarak tanımlanıyorsa, aynı işlemler Table nesnesine ait Field adlı değişkende saklanır. Lütfen dikkat edelim: Field kendi başına bir nesne iken, Fields Table nesnesine ait bir değişkendir. BUNdan çıkarılması gereken sonuç şu olmalıdır: Hazırlanan Field nesnesi daha önceden hazırlanan Table nesnesinin Fields değişkenine aktarılmalıdır. Bir Table nesnesinde birden fazla Field(ala) bulunacağından Table nesnesine ait Fields değişkeni Dizi değişken özelliğine sahip olmak durumundadır.

```
Tablo.Fields.Append Alan1
```

Bu program satırı ile örneğimize göre hard diskteki adlı ADRES ve program içindeki adı "Tablo" olan Table nesnesine Append methodu ile "Alan1" adlı Field

nesnesi eklenmektedir. Table nesnesine eklenen bu Field nesnesi Table nesnesine ait Fields degiskeninde saklanmaktadır. Hazirladigimiz örnek Table nesnesinde 5 alan olacagi için ayni islemler diger bütün alanlar için tekrarlanır.

Simdi sıra table nesnesine,yapilacak kayitlara erisimde kullanılacak olan Index'in belirlenmesine geldi. Table nesnesine eklenen alanlardan istenen alan Index anahtari olarak kullanılabilir. Örnek olması için "soyad" adli alani Index anahtari olarak kullanacagiz. Örnek programin basinda New Index bildiri deyimi ile program içindeki adi "idx" olan bir index nesnesi tanımlamistik. Bu Index nesnesinden Table nesnesi içinde yararlanabilmek veya Index nesnesini Table nesnesine dahil edebilmek için daha önce Index nesnesine ait Name, Unique, Primary ve Fields degiskenlerine bilgi aktarilmasi gerekiyor.

```
Idx.Name="PrimaryKey"  
Idx.Unique=True  
Idx.Primary=True  
Idx.Fields="soyad"
```

Index'in adi Name degiskenine, Index anahtari olarak kullanılacak alanin adi ise Index nesnesine ait Fields degiskenine aktarilir. Table nesnesine ait Fields degiskeni ile Index nesnesine ait olani birbirleriyle karistirmamak gerekir. Bir Table nesnesinde birden fazla Index kullanılabilir. Ancak bunlardan yalnızca biri Primary Index olarak seçilebilir. Hazirlanan Index nesnesi birincil index olarak kullanılacaksa, index nesnesine ait primary degiskenine mantiksal True degeri aktarilamalidir. Bunun disinda ayni anahtar degerine sahip birden fazla kayda izin verilip verilmeyecegi Index nesnesine ait Unique özelligi ile belirlenir.

Index nesnesinin gerekli olan bütün özellikleri bu sekilde belirlendikten sonra yine Append methodu ile hazirlanan Index nesnesi Table nesnesine dahil edilmelidir. Table nesnesine dahil edilen Field nesneleri Table nesnesine ait Fields nesnesine ait Indexes degiskeninde saklanır. Bu nedenle Append methodundan önce Table nesnesinin Index aktarilan degiskenin adi verilmelidir.

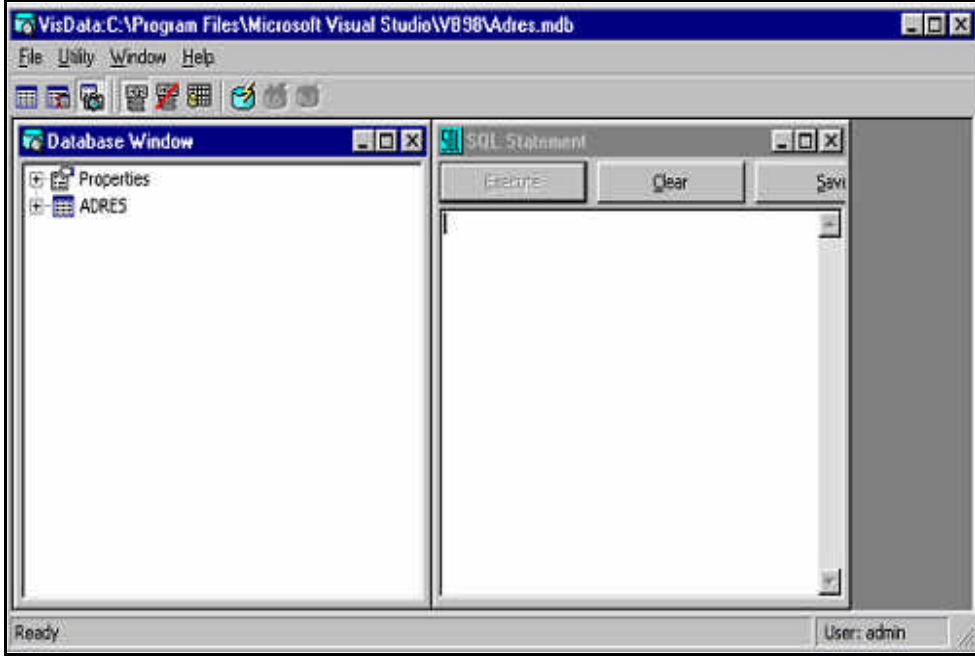
```
Tablo.Indexes.Append Idx
```

Hazirladigimiz Index nesnesine Idx adini verdigimizi hatirlatalim. Bu islemlerden sonra Table nesnesi veri tabani nesnesine dahil edilebilir duruma geldi. Table nesnesini Database nesnesine dahil edebilmek için yine Append methodundan yararlanilir. Database nesnesinde bulunan Table'lar Database nesnesine ait TableDefs adli degiskende saklandigi için database nesnesine Table nesnesi eklemede kullanılan program satiri asagidaki gibi olmalidir. Hazirladigimiz örnekte Database nesnesine "Dosya" adini verdigimiz biliyorsunuz.

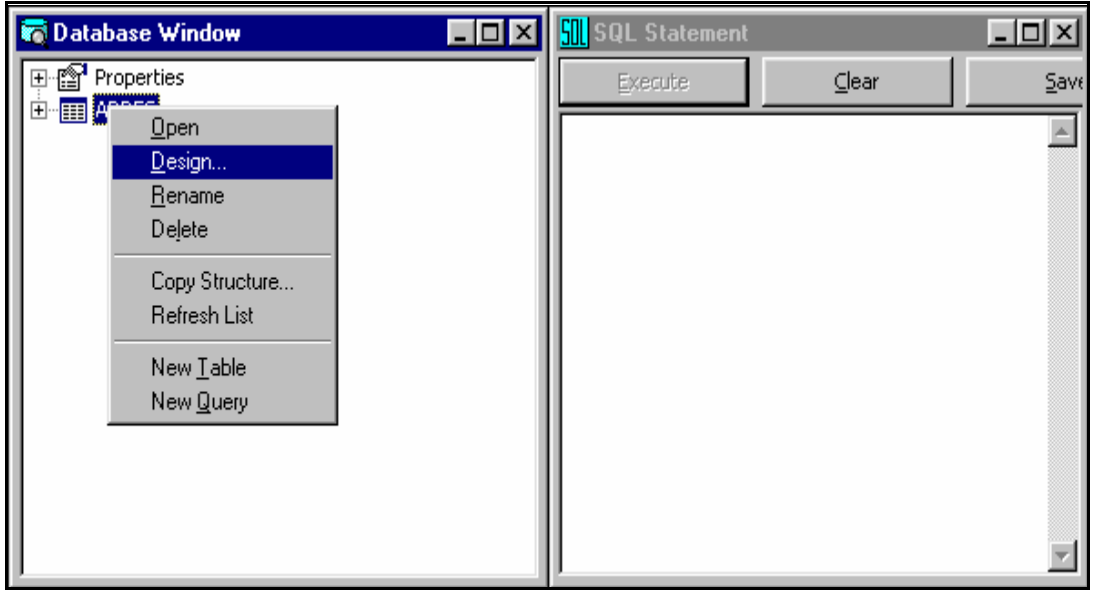
Microsoft Visual Basic 6.0

Dosya.TableDefs.Append Tablo

Bu örnek program çalıştırıldıktan sonra harddiskteki adı TEST:MDB olan Access formatındaki veri tabanı dosyasına ADRES adındaki bir tablo eklenir.bunu göstermek için TEST.MDB dosyasını Visual Data Manager ile açıp ekrana getirdim.



Program kodu yazarak veri tabanı dosyasına dahil ettiğim tablonun kayıt yapısını size göstermek için ADRES adını verdiğimiz tablonun üzerinde farenin sağ tuşu ile tıklama yaparak tablo üzerinde yapılabilecek işlemleri temsil eden kısayol menüsünü açalım.



Tabloya ait kısayol menüsünden Design komutu verilirse ADRES adlı tablonun kayıt yapısı ekrana getirilir. Bunu aşağıda verilen ekran görüntüsünde tespit edebilirsiniz. Table Structure diyalog kutusunda program kodu yazarak tablo için tanımlanan indexi görebilirsiniz.

Table Structure

Table Name:

Field List:

- ad
- soyad
- adres
- tel
- sehir

Name:

Type: ☐ FixedLength

Size: ☒ VariableLength

CollatingOrder: ☐ AutoIncrement

OrdinalPosition: ☒ AllowZeroLength

ValidationText:

ValidationRule:

Default Value:

Index List:

- primarykey

Name:

☒ Primary ☒ Unique ☐ Foreign

☒ Required ☐ IgnoreNull

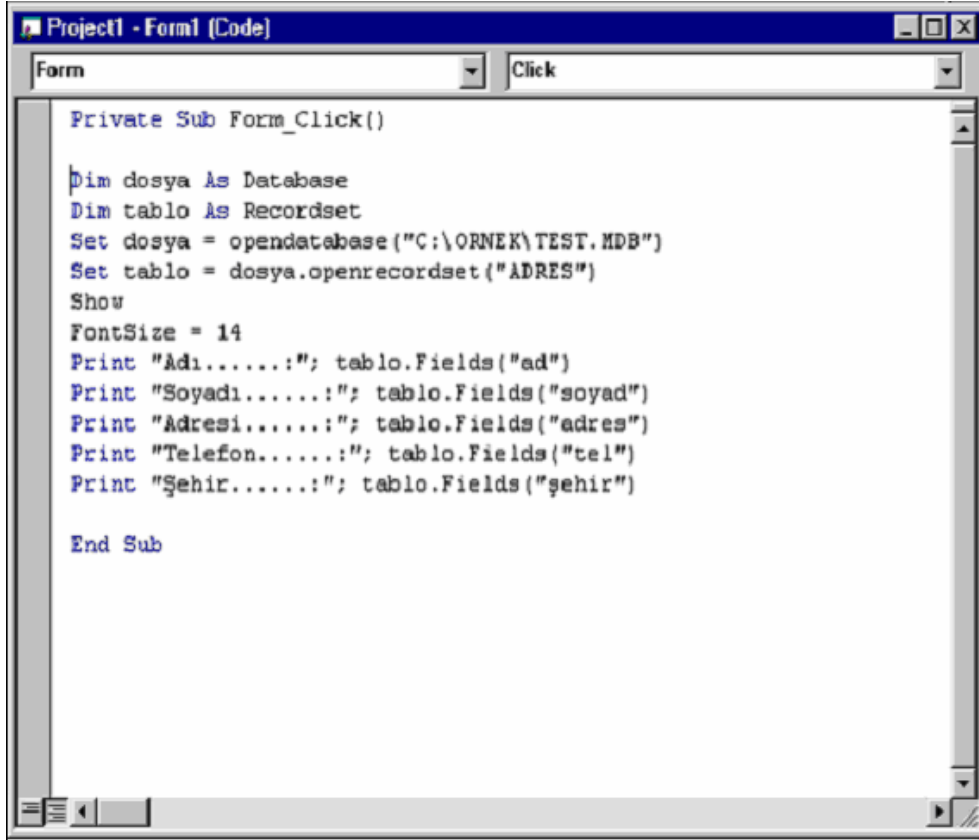
Fields:

Create Database() fonksiyonu ile bu sekilde hazirlanan veri tabani dosyasi üzerinde Data kontrolü aracılığıyla her türlü işlem yapılabilir. Ancak daha önce belirtildiği gibi bu bölümden itibaren Data kontrolünden yararlanmadan veri tabani dosyaları üzerinde nasıl işlem yapıldığı konusunda bilgi verilecektir.

MDB uzantili Access formatli veri tabani dosyalarında Tablolar, Sorgular, Raporlar ve Formlar bulunmaktadır. Ancak asıl Tablolar üzerinde işlen yapmaktayız. Bu nedenle Table nesnesi hakkında biraz daha bilgi sahibi olmak gerekir. xBASE veri tabani programında DBF uzantili bir veri tabani dosyası USE deyiimi ile açıldığı zaman kayıt göstergesi dosyadaki ilk kayda konumlanır. Benzer durumla Visual Basic dahilinde MDB uzantili veri tabani dosyaları üzerinde işlem yaparken karşılaşılar. Üzerinde bulunan kaydın içeriği Table nesnesi alanlarına otomatik bir şekilde aktarılır.

Bu anlattıklarımızı örneklemek için aşağıda verilen örnek program kodunu hazırlayalım. Bu program kodu dahilinde bu bölümün daha önceki sayfalarında hazırlanan TEST.MDB adlı veri tabani dosyasındaki ADRES tablosundaki ilk kaydın içeriği ekrana getirilmektedir. Aşağıda ekran görüntüsü verilen program

satirlerini çalistirmadan önce Visual Data Manager'dan yararlanarak ADRES tablosuna bir kaydın girisini yapalım. Henüz kayıt içermeyen bir tabloyu açıp tablonun alanları üzerine işlem yaparsanız hata meydana gelir.



Verilen örnek programın kodlarından görüleceği gibi en basta Database bildiri deyimi ile bir Database nesnesi ve Table bildiri deyimi Table nesnesi için değişken tanımlanması yapılmaktadır. Daha sonra Open Database() fonksiyonu ile önce TEST.MDB veri tabanı dosyası ardından OpenRecordSet() fonksiyonu ile TEST.MDB dosyasındaki ADRES tablosu açılmaktadır. Tablo açılır açılmaz kayıt okuma kafası birincil Index'in işaret ettiği ilk kayda konumlanır.

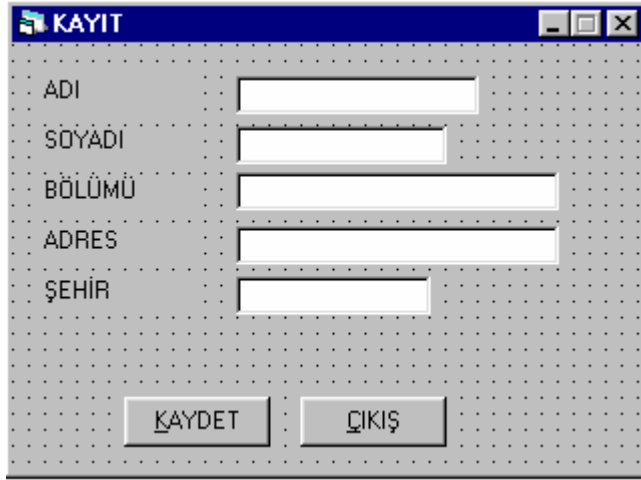
Bu sırada kaydın içeriği Print deyimi ile ekrana yazılmaktadır. From_Click yordamı bu şekilde hazırlanan Proje çalıştırılıp formun üzerinde tıklama yapılırsa aşağıda verilen ekran görüntüsü elde edilir.

Programa eklenecek bu program kodu ile tabloda bulunan diğer kayıtların içerikleri ekrana getirilebilir. Kaydın içeriğini Print deyimi ile ekrana vermek yerine formun üzerine Textbox nesneleri yerleştirilip kayıtların içerikleri görüntülerken Textbox kullanılabilir.

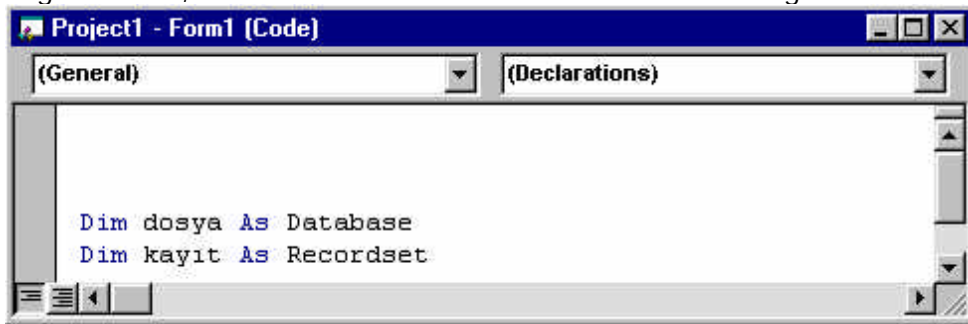
Microsoft Visual Basic 6.0

MDB Dosyasina Yeni Kayit Yazmak

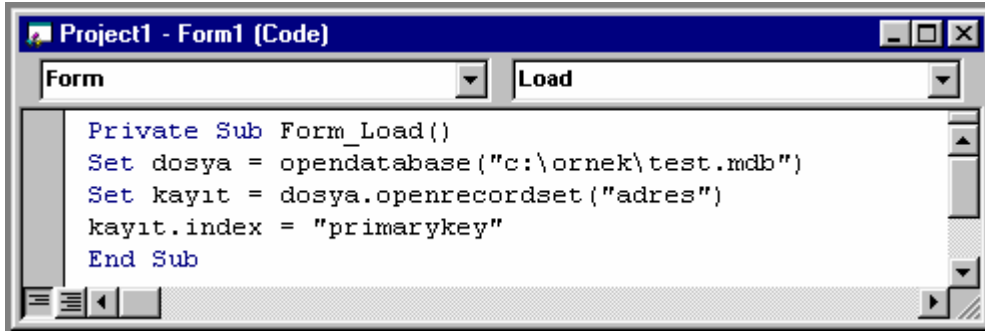
Yukaridaki sayfada verilen program kodu ile daha önce olusturulan MDB uzantili veri tabani dosyasinin ilk kaydinin Print deyimi ile ekrana yazilmasi saglandi.Simdi ise söz konusu veri tabani dosyasina yeni bir kaydın nasıl yazıldığı konusunda bilgi verilecektir.Bunun için asagida ekran görüntüsü verilen formu hazirliyalim.



Çalışma anında kullanıcı formdaki Textbox'lara kaydın içeriğini girip Kaydet düğmesinde tıklama yapılırca Textbox'ların içerikleri MDB dosyasının ADRES adlı tablosuna yazılacaktır.Bu nedenle bir önceki bölümde verilen örnekte olduğu gibi TEST.MDB veri tabanı dosyası ile dosyanın ADRES adlı tablonun açılması sağlanmalıdır.Veri tabanı dosyasını ve dosyadaki tabloyu açmak için daha önceden Database ve RecordSet tipli iki değişkenin tanımlanması gerekir.Veri tabanı dosyasına kayıt girmek amacıyla kullanılacak bu örnek projede birden fazla yordam olacağı için Database ve RecordSet tipli değişkenlerin,forma aitDeclaration kısmında tanımlanması gerekir.



Database tipli degiskene "Dosya", RecordSet tipindeki degiskene ise "Kayit" adini verelim. Degisken tanımlama işleminden sonra dosyanın ve tablonun açılması sağlanmalı veya tanımlanan degiskenlere Set edilmelidir. Bu işlemleri yapacak program satırlarını From_Load yordamina dahil edelim.



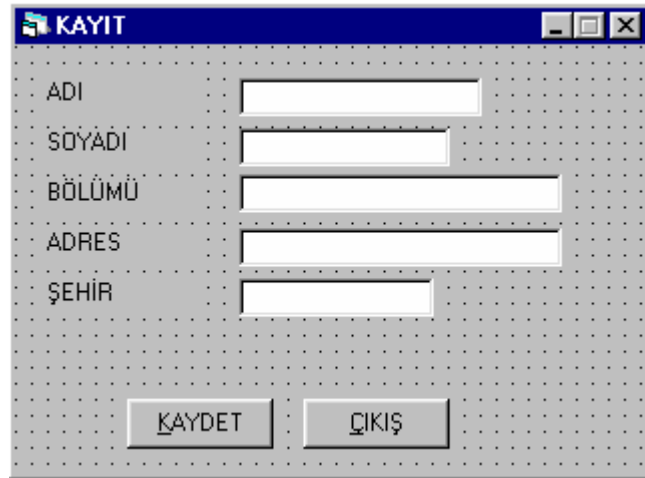
```

Project1 - Form1 (Code)
Form Load
Private Sub Form_Load()
Set dosya = opendatabase("c:\ornek\test.mdb")
Set kayit = dosya.openrecordset("adres")
kayit.index = "primarykey"
End Sub

```

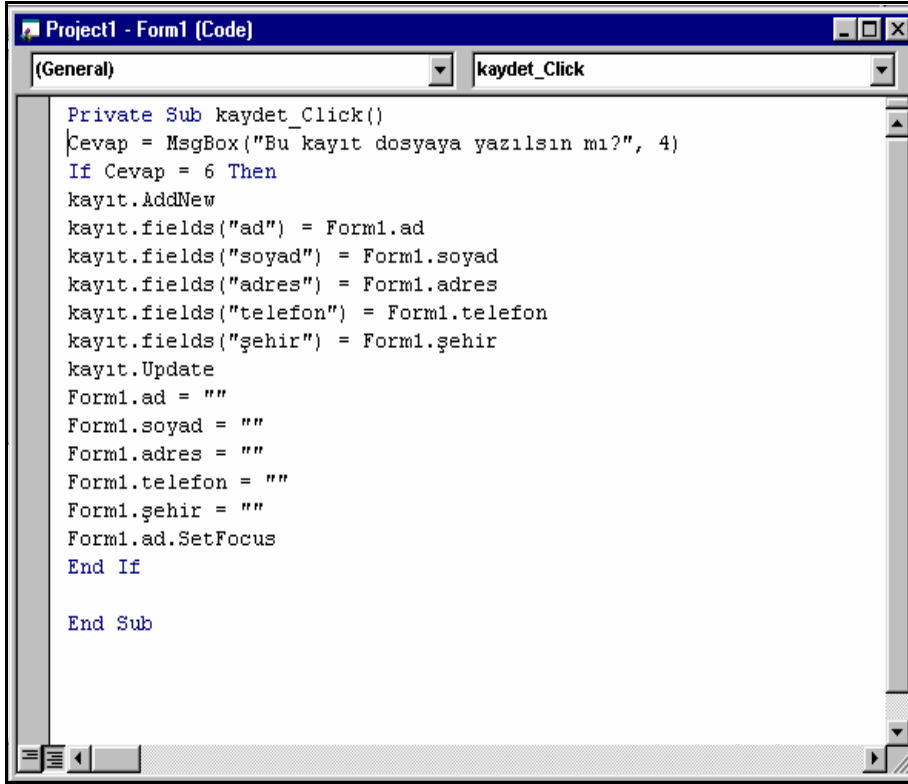
From_Load yordamındaki Kayit.Index="PrimaryKey"satiri ile kullanılacak index dosyasi seçilmektedir. Kayit girisinde olmasa bile kayit okumada mutlaka index dosyasi açılmalıdır. Seçilen index dosyasi o sırada kullanılan Table nesnesine ait "Index"adli degiskende saklanır.

Kayit girilecek tablodaki alanlarla aynı sayıda TextBox bulunan bir forma sahip bu proje çalıştırılınca imleç formdaki ilk TextBox'a konumlanır ve aşağıdaki gibi bir görüntü elde edilir.



The screenshot shows a Visual Basic form titled "KAYIT". It contains five text boxes for data entry, labeled "ADI", "SOYADI", "BÖLÜMÜ", "ADRES", and "ŞEHİR". Below the text boxes are two buttons: "KAYDET" (Save) and "ÇIKIŞ" (Exit). The form has a dotted grid background.

Veri tabanı dosyasına yazılmak istenen kaydın ayrıntılarının ilgili TextBox'lara yazılması tanımlanınca formdaki Kaydet başlıklı düğmede tıklamayapılmalıdır. Bu nedenle TextBox'lara girilen bilgileri MDB dosyasına yazacak program kodlarının Kaydet başlıklı düğmeye ait Click yordamina yazılması gerekir.



Çalışma anında TextBox'lara bilgi girilip "Kaydet" düğmesinde tıklama yapıldığı zaman işletilen Kaydet_Click yordamında en başta kullanıcıdan MsgBox() fonksiyonu ile kaydın tablo yazılması onay alınmaktadır. Aşağıda verilen ekran görüntüsünü çalışma anındaki formdaki metin kutularına bilgi girip kaydet düğmesine tıklama yaptıktan sonra aldık.

The screenshot shows a form titled 'Veri Tabanı İşlemler'. It contains several text boxes for user input and three buttons at the bottom.

Label	Value
ADI	Memik
SOYADI	Yanık
TELEFON	2249210
ADRES	Köy Hizmetleri İl Müd
ŞEHİR	Adana

Buttons: ÖNCEKİ KAYIT, SONRAKİ KAYIT, ÇIKIŞ

Yöneltilen soruya evet cevabının verilmesi halinde ilk olarak Addnew methodu ile tablonun sonuna bos bir kayıt eklenir.Ardından TextBox'ların içerikleri tek tek kaydın alanlarına aktarılır.En son olarak Update methodu ile RecordSet nesnesinin içeriklerinin tabloya yazilmasi saglanmaktadır.Tabloya ayni sekilde yeni bir kayıt yazmak için ayni işlemlerin tekrar edilmesi gerekir.

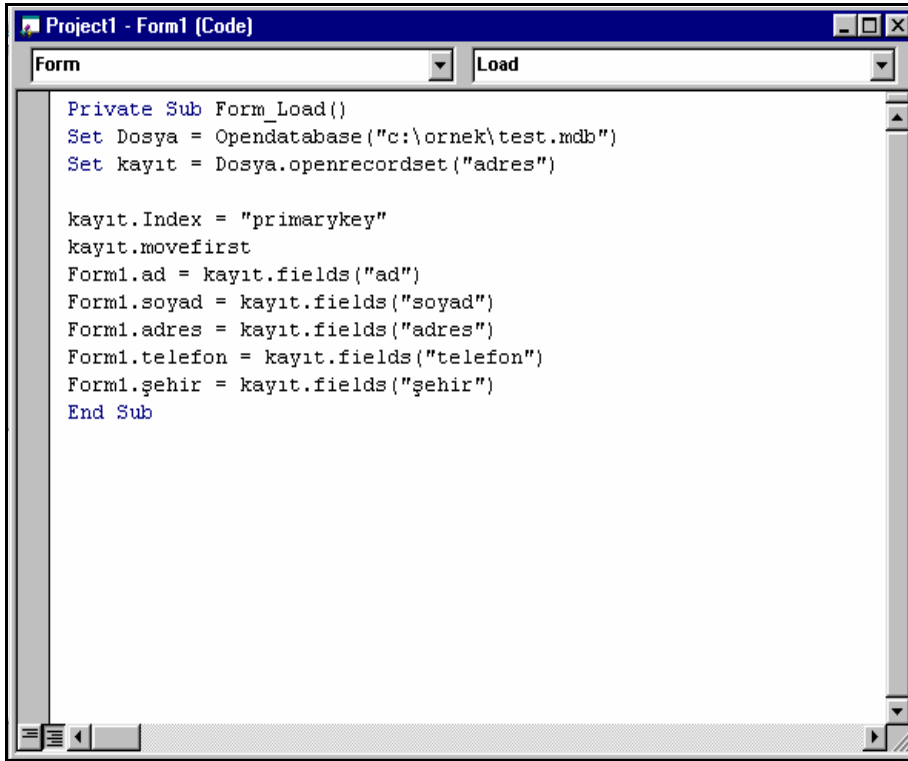
Veri Tabanı Dosyasındaki Kayıtların Arasında Dolasmak

Data kontrolünün anlatıldığı bölümde veri tabanı dosyasının kayıtları arasında nasıl dolastığı ve bu amaçla hangi methodların kullanıldığı konusunda bilgi verilmisti.Bu işlemler Data kontrolü yardımıyla çok kolay bir şekilde yapılabilirdi.Simdi size Data kontrolünü kullanmadan veri tabanı dosyasının tablolarında bulunan bulunan kayıtlar arasında nasıl dolastığı konusunda bilgi vereceğim.

Bu amaçla hazırladığımız örnek projenin Declaration kısmında yine Database ve RecordSet tipinde iki değişken tanımlayalım.Bu projenin formuna yukarıda verilen örnek projenin formundan farklı olarak Sonraki Kayıt ve Önceki Kayıt başlıklı iki düğme olacak.Yukarıda hazırlayıp program kodlarını verdiğimiz basit projeyi tablonun sonuna yeni kayıt yazmayı düşünerek hazırladığımız için sizinde aşağıdaki sayfalarda anlatacağım işlemleri yeni bir proje üzerinde gerçekleştirmenizi istiyorum.

The screenshot shows a Visual Basic form titled "Veri Tabanı İşlemler". The form has a grid background. It contains five text boxes for data entry, each with a label to its left: "ADI", "SOYADI", "TELEFON", "ADRES", and "ŞEHİR". Below these text boxes are three buttons: "ÖNCEKİ KAYIT", "SONRAKİ KAYIT", and "ÇIKIŞ". The "SONRAKİ KAYIT" button is highlighted with a blue border.

Proje başlatılır başlatılmaz hem veri tabanı dosyası açılacak hemde dosyanın seçilen tablosundaki ilk kaydı ekrana getirilecek.Bu amaçla projenin formunun From_Load yordamını aşağıdaki gibi düzenliyeğim.



Örnek projenin From_Load yordamini bu şekilde düzenledikten sonra hemen biruyarida bulunabilirim.Program kodu yazarak çalışma anında hazırladığımız ve TEST adini verdigimiz veri tabani dosyasina ADRES adında bir tablo eklemis ve bu tablo için "SOYAD alanini temel alan bir Index tanımlamistik.Ayrıca Visual Data Manager'dan yararlanarak tabloya kaydini yaptigimiz kisinin soyadi"Y" harfi ile basliyordu.Ardından program kodu yazarak ADRES tablosuna yazdigim 2.kaydin "soyad" alanındaki bilgi "I" harfi basliyordu.Türkçenin alfabetik sirasina göre "I"harfi"Y" harfinden önce geldigi zaman yukarida ekran görüntüsü verilen yordam isletildigi zaman tabloya eklenen 2.kaydin ekrana getirilmesi gerekir.Bu yordam sayesinde Proje çalıştırılır çalıştırılmaz geçerli Indexin isaret ettigi ilk kayıt ekrana getirilir.

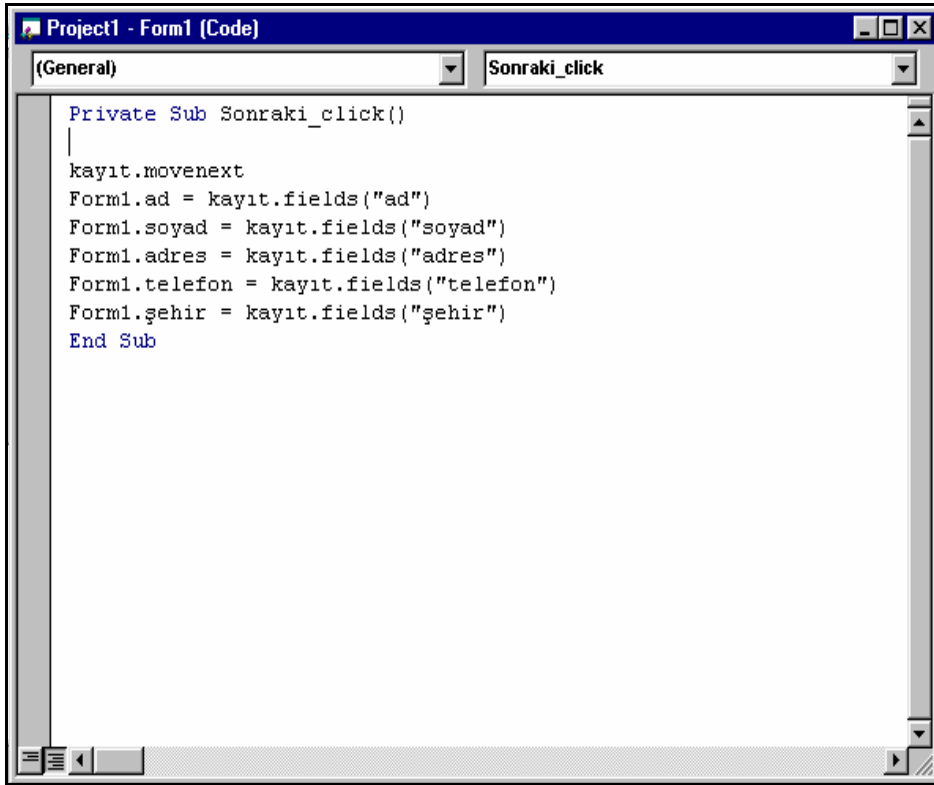
ADI	Memik
SOYADI	Yanık
TELEFON	2249210
ADRES	Köy Hizmetleri İl Müd
ŞEHİR	Adana

ÖNCEKİ KAYIT SONRAKİ KAYIT ÇIKIŞ

Ancak TEST adini verdigimiz veri tabani dosyasini CreateDatabase() fonksiyonu ile hazirlarken fonksiyonu ikinci parametre olarak "dbLangGeneral" yerine dbLangTurkish bilgisini vermis olsaydik tanimlanan Indexler Türkçe'nin alfabetik sirasini gösterirlerdi.

```
Set Dosya = Create Database("C:/ORNEK/TEST.MDB",dbLangTurkish)
```

Bu sirada Sonraki Kayit baslikli düğmede tiklama yapilince bu düğme ile ayni ada sahip yordam isletilir bir sonraki kaydi ekrana getirmek için MoveNext methodu ile kayit okuma kafasi bir sonra kayda konumlandirilir. Bu konumlandirma islemi ile birlikte tabloda bulunan alanlarin içerikleri güncellesir. Eson olarak tabloda bulunan alanlarin içerikleri formda bulunan TextBox nesnelere akarilir.



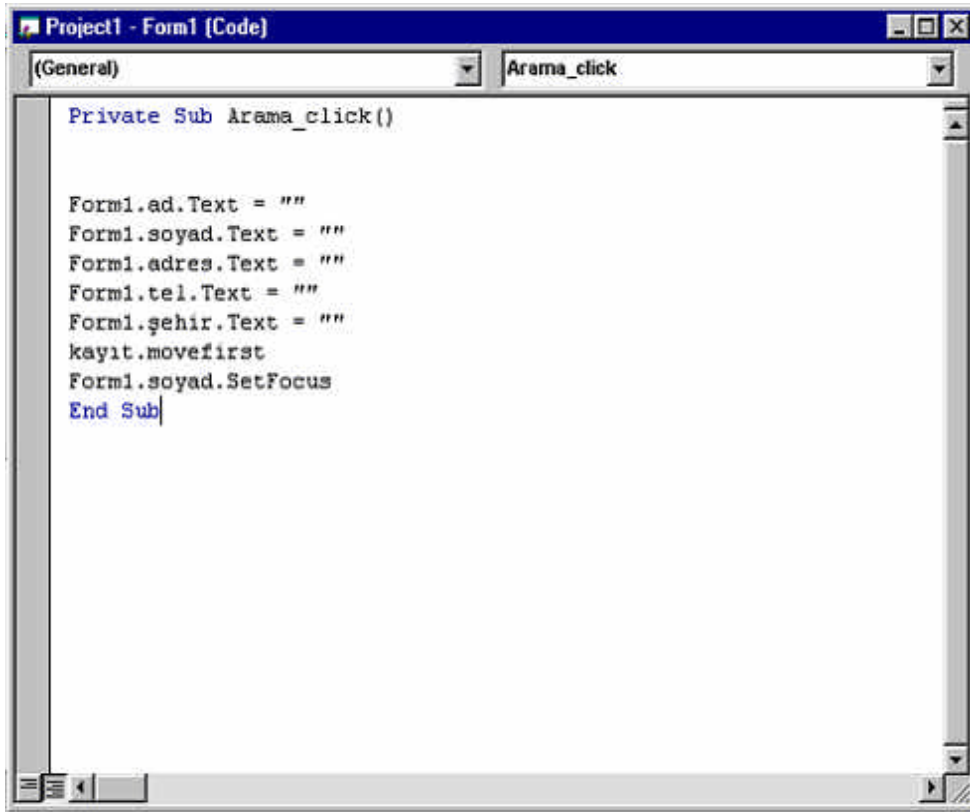
Bir önceki kayda gitmeyi sağlayan yordamin bu yordamdan tek farki MoveNext yerine MovePrevious methodunu kullanmasidir.Data kontrolü hakkında bilgi verilen bölümde belirtildiği gibi kayıt okuma kafasi tablodaki en son kayitta iken MoveNext ,kayıt okuma kafasi ilk kayitta iken MovePrevious methodu kullanildiginda hata meydana gelmekte ve programin çalışması kirlenmektedir.Bu nedenle MoveNext ve MovePrevious methodlarını kullanmadan önce dosya sonu ve basi kontrolü yapılmalıdır.Bu amaçla EOF() ve BOF() fonksiyonlarından yararlanılır.

Veri Tabanı Dosyasında Kayıt Aramak

Çok sayıda kayıt istemeyen tablolarda istenen kaydı bulup ekrana getirmek için yukarıda anlatıldığı gibi MoveNext ve MovePrevious methodlarından yararlanılabilir.Ancak tablolarda çok sayıda kayıt varsa bu şekilde kayıt aramak bazen pratik olmaz.Bu amaçla Indexlerden yararlanarak hızlı bir şekilde istenen kayıt tablodaki bulunabilir.Veritabanı dosyasının aktif tablosundan arama yapmak için Seek methodu kullanılır.Açık olan herhangi bir index yokken Seek methodu kullanılmaz.Seek methodunun çalışma şeklini gösterebilmek için yukarıda verilen örnek projenin formuna "Kayıt Arama"başlıklı bir düğme ekleyelim.

The screenshot shows a Visual Basic form titled "Veri Tabanı İşlemler". The form has a dotted background. It contains five text boxes for data entry, each with a label to its left: "ADI", "SOYADI", "TELEFON", "ADRES", and "ŞEHİR". Below the text boxes are four buttons: "ÖNCEKİ KAYIT", "SONRAKİ KAYIT", "ÇIKIŞ", and "KAYIT ARAMA".

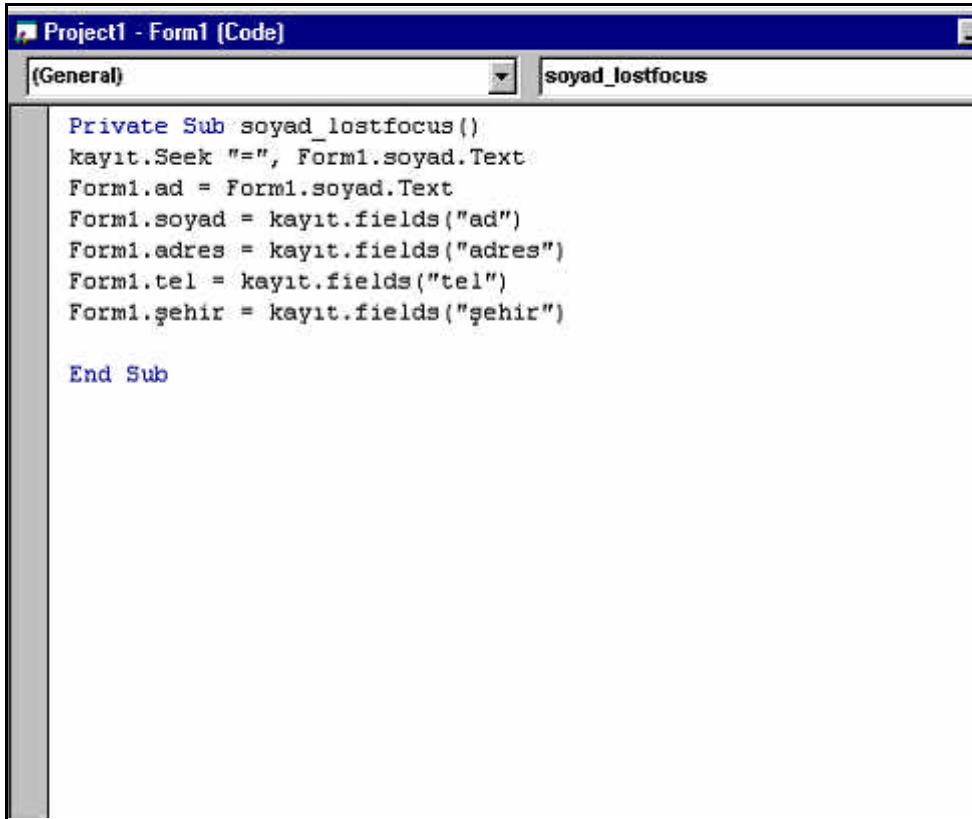
Kullanıcı çalışma anında tablodan kayıt araması yapmak istediği zaman Kayıt Arama başlıklı düğmede tıklama yapacak. Bu düğmede tıklama yapılırken kullanıcının aranacak kaydın arama anahtarı istenir. Üzerinde çalıştığımız TEST.MDB dosyasındaki tek table nesnesi olan ADRES tablosunda bir tek index hazırlanmış ve index için soyad alanı index anahtarı olarak kullanılmıştır. Buna göre bir Seek methodu ile ancak soyad alanına göre kayıt araması yapılabilir. Bu amaçla Kayıt Arama başlıklı düğmede tıklama yapılırken SetFocus methodu ile imlecin "Soyad" TextBox'ine gitmesi sağlandı.



Verilen program kodlarından tespit edilebileceği gibi Kayıt Arama başlıklı ve "Arama"adlı düğmede tıklama yapılırken, formdaki bütün TextBox'ların içerikleri silinmektedir. Aşağıda verilen ekran görüntüsünü çalışma anında Kayıt Arama başlıklı düğmede tıklama yaptıktan sonra çalışma anında alalım.

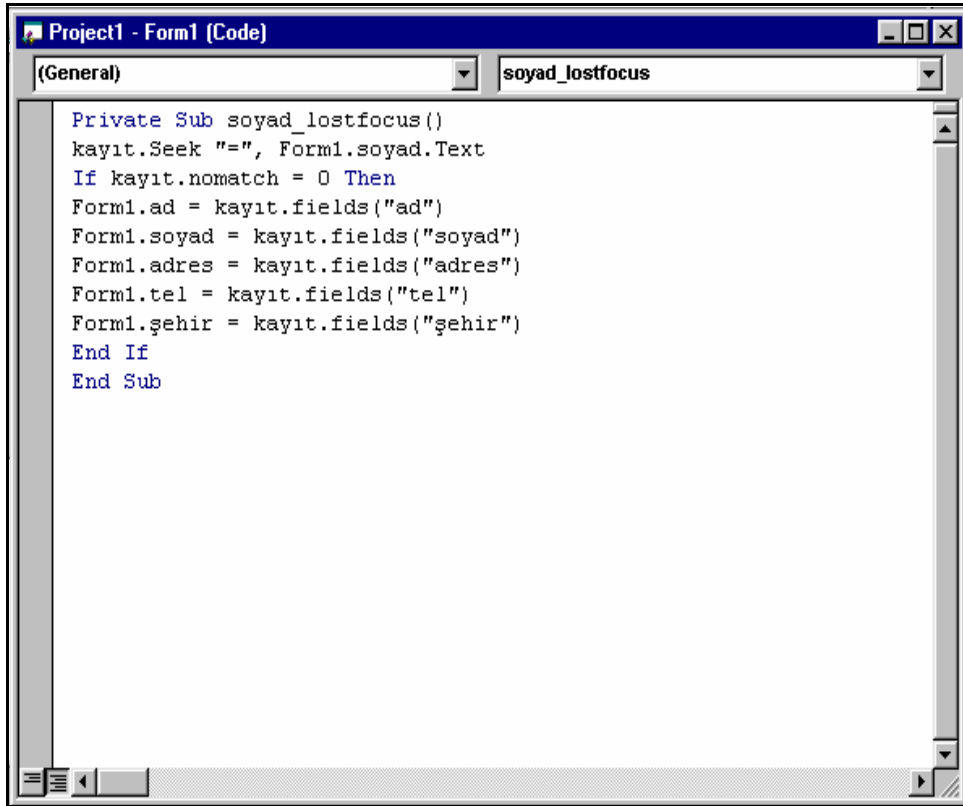
The screenshot shows a Windows-style application window titled "Veri Tabanı İşlemler". The window has a standard title bar with minimize, maximize, and close buttons. The main area of the window has a light gray background with a fine grid pattern. It contains five text boxes arranged vertically, each preceded by a label: "ADI", "SOYADI", "TELEFON", "ADRES", and "ŞEHİR". Below these text boxes, there are four buttons arranged horizontally. The buttons are labeled "ÖNCEKİ KAYIT", "SONRAKİ KAYIT", "ÇIKIŞ", and "KAYIT ARAMA".

Bu sirada Tab tusu veya fare ile ekleme noktasini bir sonraki metin kutusuna aktaracak olursaniz" Soyad" adli bu metin kutusu ile ilgili olarak LostFocus olayi meydana gelir. Bu nedenle kayıt arama islemini yapacak program kodlarini "Soyad" adina sahip metin kutusunun LostFocus yordamina dahil edelim. Yapılan tanımlamaya göre kullanıcı aramak istedigii kaydın anahtarini girdikten sonra aramayı başlatabilmesi için Soyad adındaki TextBox'ta iken tab tusuna basması gerrekiyor.



```
Private Sub soyad_lostfocus()  
    kayıt.Seek "=", Form1.soyad.Text  
    Form1.ad = Form1.soyad.Text  
    Form1.soyad = kayıt.fields("ad")  
    Form1.adres = kayıt.fields("adres")  
    Form1.tel = kayıt.fields("tel")  
    Form1.şehir = kayıt.fields("şehir")  
  
End Sub
```

Soyad adlı TextBox ta iken Tab usuna basılması halinde işletilecek bu yordam ile soyad TextBox'in içerisine göre Seek methodu ile arama yapılır. Bu yordamda aranan kayıt bulunduktan sonra alanların içerikleri ilgili TextBox lara aktarılıyor. Aranan kayıdın dosyadan bulunup bulunmadığını öğrenmek için Table nesnesine ait NoMatch değişkeninin içerisine bakılır. Seek methodu ile yapılan her arama işleminden sonra tabloya ait NoMatch değişkeninin içeriği güncellenir. Aranan kayıt tabloda varsa bu değişkenin içerisine mantıksal False(0), aranan kayıt bulunmazsa bu kez NoMatch değişkenine mantıksal True(1) değeri aktarılır. Bu nedenle yukarıda verilen program kodunu biraz değiştirelim.



Örnek yordami dikkatlice inceliyecek olursanız Seek methodunun disaridan iki parametre aldığını görebilirsiniz. Birincisinde "=" "<" gibi oparetörler, ikincisinde ise arama anahtari verilir. Eger arama anahtarimiz Maas gibi sayisal bilgi içeren bir alan ve tablodan maasi 15.000.000'dan fazla olan herhangi bir kisinin kaydi araniyor olsaydi Seek methodu TabloNesnesi.Seek>15000000 seklinde kullanilirdi. Seek methodundan sonra tabloya ait NoMatch degiskeninin içeriği güncellesir. Aranan kayit bulunmus ise bu degiskenin içeriği mantiksal yanlis veya 0 olur.

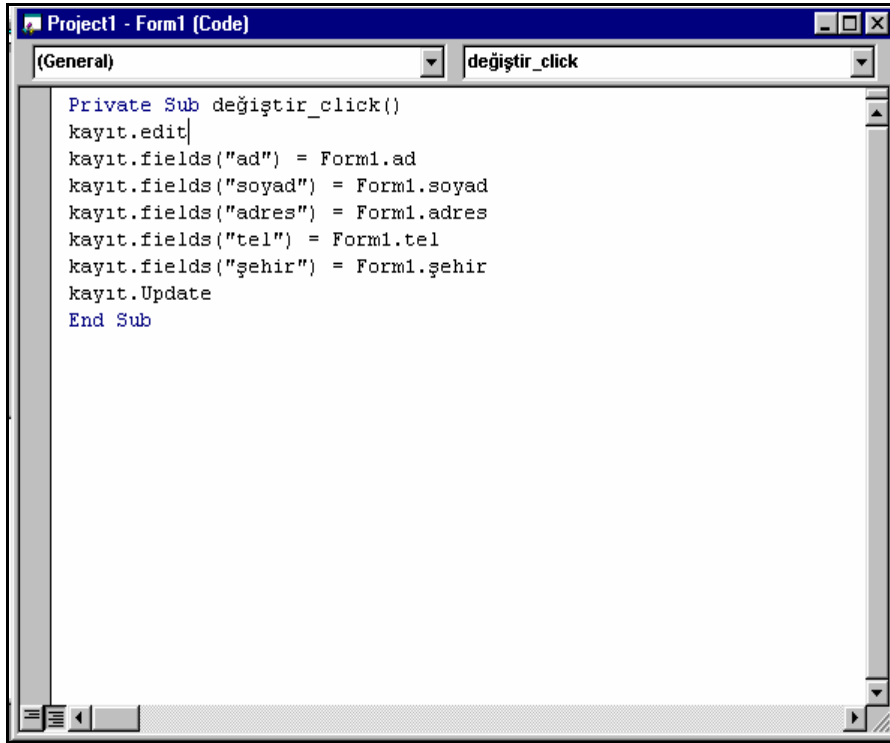
Kayitlarda Degisiklik Yapmak

Veri tabani dosyasinin table nesnesine yazilmis bir kaydi degistirmek için önce söz konusu kayit tablodan aranilip bulunur. Arama MoveNext veya Seek gibi methodlarla yapilabilir. Simdiye kadar veri tabani dosyasi tablosunda bulunan kayitlarin içeriklerini ekrana getirmek veya kayit girisi yapmak için her seferinde her alan için forma eklemis oldugumuz TextBox nesnelerinden yararlandik. Bu islemler için TextBox'lardan yararlanmak bir zorunluluk olmamakla birlikte TextBox kullanmak programciya kolayliklar saglamaktadır. Buna göre içeriği TextBox'lar aracılığı ile ekrana getirilmis olan

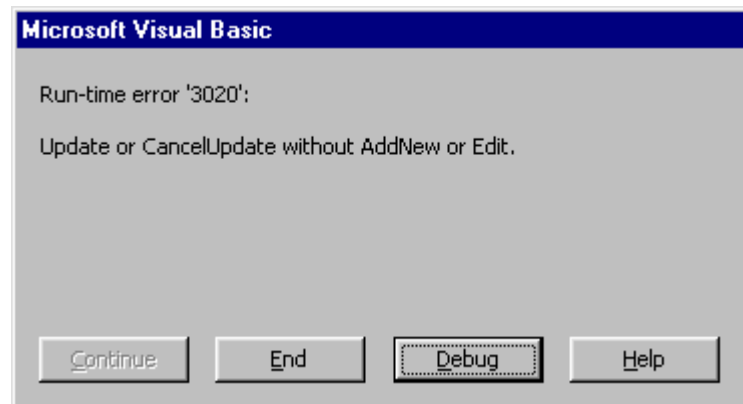
bir kayıta değişiklik yapmak için önce söz konusu TextBox'ların içeriklerinde istenen değişiklik yapılmalıdır.Ardından bu değişiklikler Table nesnesinin üzerinde bulunan aktif kaydına yansıtılmalıdır.Bu işlemleri gösterebilmek için yukarıda verilen örnek projenin formuna Degistir baslikli bir düğme ekleyelim.

The image shows a Visual Basic form titled "KAYIT". The form has a grid background. It contains five text boxes for data entry, labeled "ADI", "SOYADI", "TELEFON", "ADRES", and "ŞEHİR". To the right of these text boxes is a button labeled "DEĞİŞTİR". At the bottom of the form, there are four buttons: "ÖNCEKİ KAYIT", "SONRAKİ KAYIT", "ÇIKIŞ", and "KAYIT ARAMA".

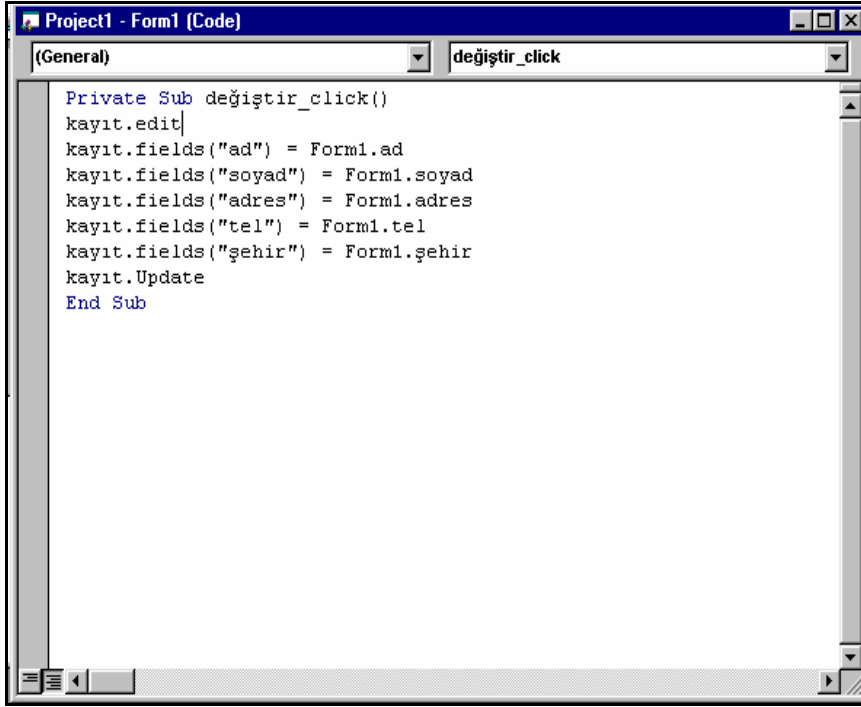
Çalışma alanında kullanıcı "Degistir" baslikli düğmede tıklama yapınca TextBox'ların en son içeriği Update methodu ile ayni kaydın üzerine tekrar yazılacaktır.Bunun içinde TextBox'ların içeriklerinin tekrar alanlara aktarılması gerekir.



Ancak okumak amacıyla ekrana getirilen bir kaydın alanlarında değişiklik yapılmaz. Yukarıda verilen program kodunda bu hata var. Dolayısıyla programın çalışması kırıldı.



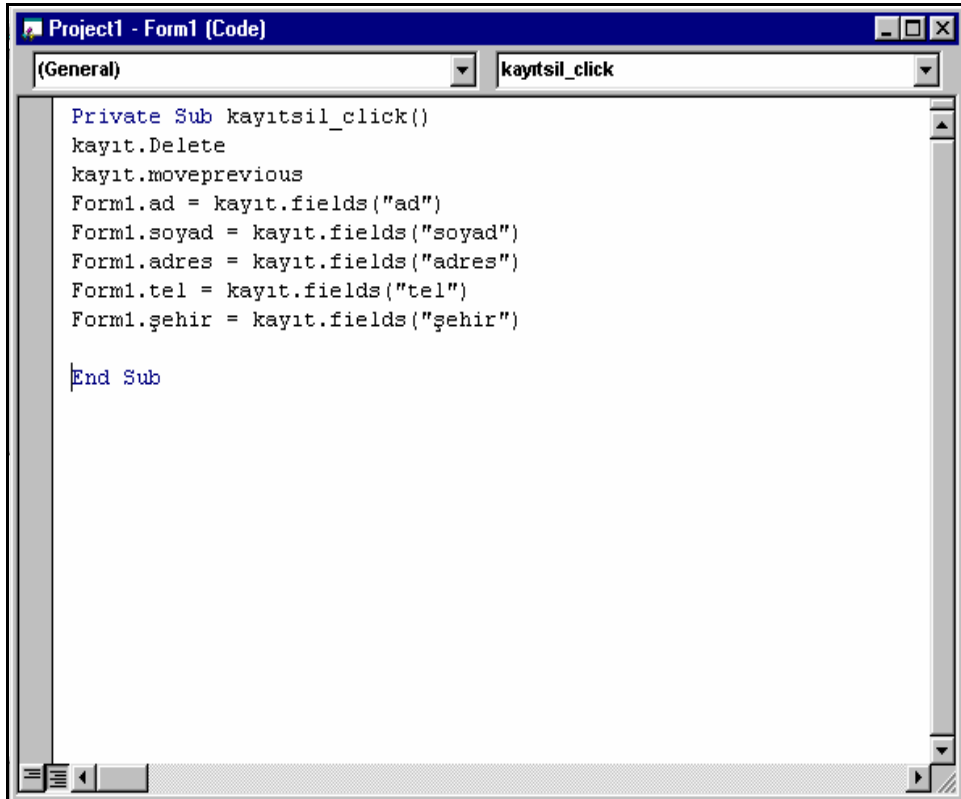
Okunan bir kaydın içeriğinde değişiklik yapabilmek için daha önceden edit modunda geçmek gerekiyor. Bunun için Edit methodundan yararlanılır. Yukarıda verilen Degistir_Click yordamina alanların içerikleri degistirilmeden önce Edit moduna geçilirse hata meydana gelmez



Edit moduna geçip aktif kayda ait alanların içerikleri değiştirildikten sonra yapılan değişikliklerin diske yani tabloya yansitilmesi için ayrıca Update methodunun kullanılması gerekir.

Veri Tabanı Dosyasından Kayıt Silmek

Program kodu yazarak MDB uzantılı veri tabanı dosyasının tablolarından kayıt silmek, Data kontrolü yardımı ile yapılan kayıt silme işleminden farklı değildir. Önce silmek istenen kayıt bulunur. Ardından Delete methodu ile söz konusu kayıt dosyadan silinir. Silmek istenen kayıt MoveNext gibi methodlarla veya Seek methodu ile bulunabilir. Hazırladığımız bu örnek projenin formuna tablodan kayıt silme özelliği kazandırabilmek için Kayıt Sil başlıklı bir düğme ekleyelim. Aranılan kayıt bulunup Kayıt Sil başlıklı düğmede tıklama yapılıncaya aşağıda verilen program kodları ile üzerinde bulunan kayıt dosyadan silinir.



Bu yordam Delete methodu ile yapılan kayıt silme işleminden sonra MovePrevious methodu ile tablodaki bir önceki kayda gidilmekte ve bu kaydın içeriği TextBox'lar aracılığı ile ekrana getirilmektedir.

The screenshot shows a form titled 'KAYIT'. It contains several input fields for data entry: 'ADI SOYADI' (Name Surname), 'NUMARASI' (Number), 'BÖLÜMÜ' (Department) which is a dropdown menu, 'ÖDEVİ' (Homework), 'VİZE' (Midterm), and 'FİNAL' (Final). At the bottom of the form, there are three buttons: 'KAYDET' (Save), 'TEMİZLE' (Clear), and 'ÇIKIŞ' (Exit).

Microsoft Visual Basic 6.0

```
Dim a, i As Integer
Private Sub Command1_Click()
Data1.Recordset.AddNew
Data1.Recordset("ad_soyad") = Text1.Text
Data1.Recordset("numarasi") = Text2.Text
Data1.Recordset("bolumu") = Combo1.Text
Data1.Recordset("odev") = Text3.Text
Data1.Recordset("vize") = Text4.Text
Data1.Recordset("final") = Text5.Text
Data1.Recordset.Update
Text1.Text = " "
Text2.Text = " "
Combo1.Text = " "
Text3.Text = " "
Text4.Text = " "
Text5.Text = " "
Text1.SetFocus
End Sub

Private Sub Command2_Click()
Text1.Text = " "
Text2.Text = " "
Combo1.Text = " "
Text3.Text = " "
Text4.Text = " "
Text5.Text = " "
Text1.SetFocus
End Sub

Private Sub Command3_Click()
Unload Me
End Sub

Private Sub Form_Activate()
Text1.SetFocus
On Error GoTo k

Data2.RecordSource = "tanım"
Data2.Refresh
Data2.Recordset.MoveLast 'en son kayda gidilir
'Recordset.RecordCount kayıt sayısını öğrenmek için
a = Data2.Recordset.RecordCount
Data2.Recordset.MoveFirst

For i = 1 To a
Combo1.AddItem Data2.Recordset("bolum1")
```



```
Data2.Recordset.MoveNext
Next

Combo1.Text = ""
Text1.SetFocus
Exit Sub

k:
    Select Case Err
    Case Is = 3021
        MsgBox ("Kayit Yok")
        Text1.SetFocus
    End Select

End Sub

Private Sub Form_Load()
Form1.Height = 3600
Form1.Width = 7000
End Sub
```

The screenshot shows a Windows-style application window with a title bar that reads "ARAMA / SİLME / DÜZELTME". The window has a standard Windows 95/NT look with a blue title bar and a gray background. Inside the window, there are six input fields arranged vertically on the left side, each with a label to its left: "ADI SOYADI", "NUMARASI", "BÖLÜMÜ", "ÖDEVİ", "VİZE", and "FİNAL". The "BÖLÜMÜ" field is a dropdown menu. At the bottom of the window, there are four buttons arranged horizontally: "SİL", "DÜZELT", "İEMİZLE", and "ÇIKIŞ". The buttons have a 3D effect and are separated by vertical lines.

```
Private Sub Command1_Click()
Data1.Recordset.Delete
Data1.Refresh
Text1.Text = ""
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
```

Microsoft Visual Basic 6.0

```
Text4.Text = ""
Text5.Text = ""
Text1.SetFocus
End Sub
```

```
Private Sub Command2_Click()
Text1.Text = ""
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text1.SetFocus

End Sub
```

```
Private Sub Command3_Click()
Unload Me
End Sub
```

```
Private Sub Command4_Click()
Data1.Recordset.Edit
Data1.Recordset("ad_soyad") = Text1.Text
Data1.Recordset("numarasi") = Text2.Text
Data1.Recordset("bolumu") = Combo1.Text
Data1.Recordset("odev") = Text3.Text
Data1.Recordset("vize") = Text4.Text
Data1.Recordset("final") = Text5.Text
Data1.Recordset.Update
Data1.Refresh
Text1.Text = ""
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text1.SetFocus
End Sub
```

```
Private Sub Form_Activate()
Text1.SetFocus
End Sub
```

```
Private Sub Form_Load()

Form2.Height = 3600
Form2.Width = 7000
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
On Error GoTo k
If KeyAscii = 13 Then
SQL = "select ham.* from ham where(ham.ad_soyad like'" &
Text1.Text & "')"
Data1.RecordSource = SQL
Data1.Refresh
Text1.Text = Data1.Recordset("ad_soyad")
Text2.Text = Data1.Recordset("numarasi")
Combo1.Text = Data1.Recordset("bolumu")
Text3.Text = Data1.Recordset("odev")
Text4.Text = Data1.Recordset("vize")
Text5.Text = Data1.Recordset("final")
End If
Exit Sub
k:
Select Case Err
Case Is = 3021
MsgBox ("Kayit Yok")
End Select
Text1.Text = ""
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text1.SetFocus
End Sub
```

ad	soyad	numarası	bölümü	ödev	vize	final
ahmet		g0125.01003	Bilgisayar Programcılığı	90	85	90
mehmet		g0125.00001	Okul Öncesi Öğretimi	80	80	80
mustafa		g0125.01055	Bilgisayar Programcılığı	95	80	70

```
Private Sub Command3_Click()  
Unload Me  
End Sub
```

```
Private Sub Form_Load()  
Form3.Height = 6300  
Form3.Width = 9930  
End Sub
```

numara	yıl aci	yıl sonu/ harf	rakam
0125.01003	88	89 BA	35
0125.00001	80	80 BA	35
0125.01055	88	77 BB	3

```

Private Sub Command2_Click()
Data1.Recordset.AddNew
Data1.Recordset("bolum1") = Text1.Text
Data1.Recordset.Update
Data1.Refresh
Text1.Text = ""
Text1.SetFocus
End Sub

Private Sub Command3_Click()
Unload Me
End Sub

Private Sub DBGrid1_Click()

End Sub

Private Sub Form_Load()
Form5.Height = 6300
Form5.Width = 9930
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
Data1.Recordset.AddNew
Data1.Recordset("bolum1") = Text1.Text

```

Microsoft Visual Basic 6.0

```
Data1.Recordset.Update  
Data1.Refresh  
Text1.Text = ""  
Text1.SetFocus  
End If  
End Sub
```

```
Private Sub Command1_Click()  
Frame2.Visible = True  
  
Text6.Text = Text2.Text  
Text7.Text = (Text3.Text * 0.5) + (Text4.Text * 0.5)  
Text8.Text = (Text7.Text * 0.4) + (Text5.Text * 0.6)  
  
If Text8.Text < 26 Then  
Text9.Text = "FF"  
Text10.Text = "0"  
End If  
If Text8.Text > 25 And Text8.Text < 46 Then  
Text9.Text = "DD"  
Text10.Text = "1"  
End If  
If Text8.Text > 45 And Text8.Text < 56 Then  
Text9.Text = "DC"  
Text10.Text = "1.5"  
End If  
If Text8.Text > 55 And Text8.Text < 66 Then  
Text9.Text = "CC"  
Text10.Text = "2"  
End If  
If Text8.Text > 65 And Text8.Text < 71 Then
```

```
Text9.Text = "CB"
Text10.Text = "2.5"
End If
If Text8.Text > 70 And Text8.Text < 81 Then
Text9.Text = "BB"
Text10.Text = "3"
End If
If Text8.Text > 79 And Text8.Text < 96 Then
Text9.Text = "BA"
Text10.Text = "3.5"
End If
If Text8.Text > 94 And Text8.Text < 101 Then
Text9.Text = "AA"
Text10.Text = "4"
End If
```

```
End Sub
```

```
Private Sub Command2_Click()
Data3.Recordset.AddNew
Data3.Recordset("numara") = Text6.Text
Data3.Recordset("yil_ici") = Text7.Text
Data3.Recordset("yil_sonu") = Text8.Text
Data3.Recordset("harf") = Text9.Text
Data3.Recordset("rakam") = Text10.Text
Data3.Recordset.Update
Data3.Refresh
Text1.Text = ""
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
Text6.Text = ""
Text7.Text = ""
Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Frame2.Visible = False
Text1.SetFocus
End Sub
```

```
Private Sub Command3_Click()
Unload Me
End Sub
```

```
Private Sub Command4_Click()
Text1.Text = ""
```

Microsoft Visual Basic 6.0

```
Text2.Text = ""
Combo1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
Text6.Text = ""
Text7.Text = ""
Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Text11.SetFocus

End Sub

Private Sub Form_Activate()
Text11.SetFocus
End Sub

Private Sub Form_Load()
Form6.Height = 3720
Form6.Width = 8190
Frame2.Visible = False
End Sub

Private Sub Text11_KeyPress(KeyAscii As Integer)
On Error GoTo k
If KeyAscii = 13 Then
SQL = "select ham.* from ham where(ham.ad_soyad like'" &
Text11.Text & "')"
Data1.RecordSource = SQL
Data1.Refresh
Text11.Text = Data1.Recordset("ad_soyad")
Text2.Text = Data1.Recordset("numarasi")
Combo1.Text = Data1.Recordset("bolumu")
Text3.Text = Data1.Recordset("odev")
Text4.Text = Data1.Recordset("vize")
Text5.Text = Data1.Recordset("final")
End If
Exit Sub
k:
Select Case Err
Case Is = 3021
MsgBox ("Kayit Yok")
End Select
Text11.Text = ""
Text2.Text = ""
```



```

Combol.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text1.SetFocus
End Sub

```

```

Private Sub Command3_Click()
Unload Me
End Sub

```

```

Private Sub Form_Load()
Form7.Height = 6300
Form7.Width = 9930
End Sub

```

Microsoft Visual Basic 6.0

Form6

HAM VERİ

ADI SOYADI

NUMARASI

BÖLÜMÜ

ÖDEVİ

VİZE

FINAL

HESAPLA

TEMİZLE

KAYDET

ÇIKIŞ

```
Private Sub Command3_Click()  
Unload Me  
End Sub
```

```
Private Sub Command4_Click()  
Text1.Text = ""  
Text2.Text = ""  
Combol.Text = ""  
Text3.Text = ""  
Text4.Text = ""  
Text5.Text = ""  
Text6.Text = ""  
Text6.Text = ""  
Text7.Text = ""  
Text8.Text = ""  
Text9.Text = ""  
Text10.Text = ""  
Text1.SetFocus  
End Sub
```

```
Private Sub Form_Activate()  
Text1.SetFocus  
End Sub
```

```
Private Sub Form_Load()  
Form8.Height = 3720  
Form8.Width = 8190  
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)  
On Error GoTo k
```

```
If KeyAscii = 13 Then
SQL = "select ham.* from ham where(ham.ad_soyad like'" &
Text1.Text & "')"
Data1.RecordSource = SQL
Data1.Refresh
Text1.Text = Data1.Recordset("ad_soyad")
Text2.Text = Data1.Recordset("numarasi")
Combo1.Text = Data1.Recordset("bolumu")
Text3.Text = Data1.Recordset("odev")
Text4.Text = Data1.Recordset("vize")
Text5.Text = Data1.Recordset("final")

SQL = "select islenmis.* from islenmis where(islenmis.numara
like'" & Text2.Text & "')"
Data2.RecordSource = SQL
Data2.Refresh

Text6.Text = Text2.Text
Text7.Text = Data2.Recordset("yil_ici")
Text8.Text = Data2.Recordset("yil_sonu")
Text9.Text = Data2.Recordset("harf")
Text10.Text = Data2.Recordset("rakam")
End If
Exit Sub
k:
Select Case Err
Case Is = 3021
MsgBox ("Kayit Yok")
End Select
End Sub
```

10-VERİ ERİSİM YÖNTEMLERİ

Veri erişim yöntemleri (data access objects) programlama aracılığıyla veritabanı dosyalarına erişmek için kullanılan tekniklerdir. Veri erişim nesneleri (data access objects) ise veritabanlarına erişimi sağlayan elemanlardır.

Bugün uygulamalarda kullandığımız verilerin bir kısmı yerel birçoğu da uzak veritabanlarında gelirler yada orada saklanırlar. Bu nedenle diğer veritabanlarındaki verilere erişim yada client/server uygulamalar geliştirmek oldukça yaygın bir programlama işlemidir. Yapılan araştırmalarda kullanılan uygulamaların çoğunun dağıtılmış yapıda ve birçok kullanıcı tarafından kullanıldığını göstermektedir. Microsoft Visual Basic, veri erişimi için çok sayıda arabirim(yöntem,teknoloji) geliştirilmiştir.

Bunlar; VB-SQL, Jet/DAO, RDO ve ADO'dur. Geliştirilen veri erişim teknikleri daha az katmanla ve daha etkin bir biçimde verilere ulaşmayı sağlar.

- DAO(Data Access Objects)
- RDO(Remote Data Objects)
- ADO(ActiveX Data Objects)

VBSQL Microsoft SQL Server için geliştirilmiş bir API-arabirimdir. Microsoft SQL Server ve Sybase SQL Server'a erişim sağlar.

DAO Microsoft Jet veritabanlarına erişmek için geliştirilmiş ilk nesne temelli arabirimdir. DAO ayrıca ISAM ve ODBC veritabanlarına erişim için de kullanılır. Visual Basic ortamında yer alan veri kontrollerine(Data Control) bağlanarak kullanılan DAO veri erişim tekniği ile çok kolay bir şekilde veritabanı yaratılır.

RDO Remote Data Objects,ODBC verilerine ulaşmak için kullanacağınız diğer bir arabirimdir. RDO, Jet ve ISAM veritabanlarına erişim için kullanılmaz. RDO'nun işlevi ODBC üzerinden ilişkisel veritabanlarına bağlanmaktır.

ADO ActiveX Data Objects veri erişim yöntemi OLE DB'ye arabirim olan bir veri erişim yöntemidir. OLE DB ve ODBC sürücüleri kullanır. ADO; RDO ve DAO'nun gelişmiş biçimidir.

ODBC ODBC sürücüsü ile çok sayıda ilişkisel veritabanına erişim için geliştirilmiş bir arabirimdir. Özellikle DAO, RDO ve ADO gibi nesne-temelli erişimlerde yaygın olarak kullanılır.

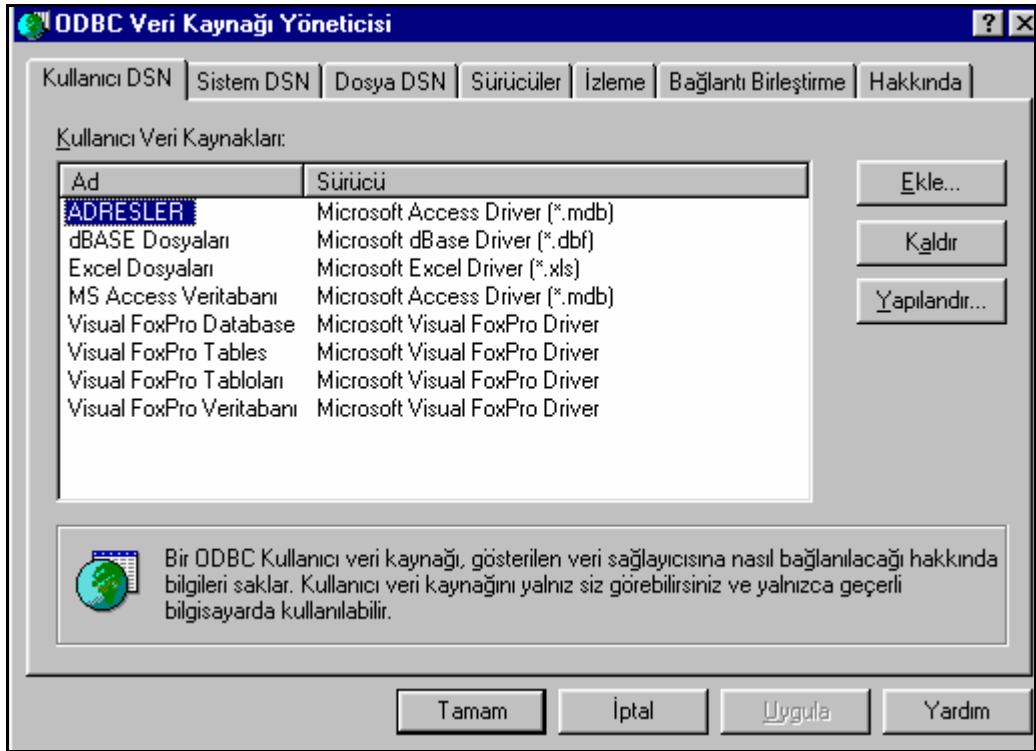
OLE DB Alt düzey bir veri erişim yöntemidir. Herhangi bir veritabanı tipi ile kısıtlı değildir. Biçim yada saklama yöntemine bakmaksızın

tüm veritabanlarına erişim için kullanılır. ODBC'nin gelişmiş biçimidir.

RDS Remote Data Service ise ADO ile bütünleşik Web verilerine erişim için kullanılır.

Visual Basic ile birlikte kullanılan veri erişimi(data access) yöntemleri, programcileri API programlamasından uzak tutmak için geliştirilmiştir. Programcilerin sadece bağlantı kurmak konusunda bilgi sahibi olmaları yeterlidir. Bu bağlantı genellikle yerel yada uzak veritabanına bağlantı şeklindedir. Örneğin ODBC aracılığıyla bir DSN tanımlamak ve onu kod içinde doğrudan kullanmak.

Örneğin bir ADO bağlantı deyimi: `cnn.Open"ADRESLER"` Tabii bu bağlantı daha önceden yapılmış bir DSN tanımı ile olmaktadır.:



Sekil-1 DSN Tanımı

Veri erişim yöntemlerinden bir tanesi olan RDO,ODBC API üzerinde, nesne temelli bir arabirim olarak geliştirilmiştir. RDO,ODBC API üzerinde bulunan bir nesne katmanidir. RDO sayesinde API düzeyinde programlamaya gerek duymadan ODBC fonksiyonlarına kolayca ulaşılır.

RDO kullanımı DAO kullanımına benzer. Ancak RDO'nun en önemli özelliği OBBC API'nin gelişmiş özellikleridir. RDO'nun standart VB Data Control gibi veri kendi kontrolü vardır. 32-bitlik bu yeni programlama modeli olan RDO sayesinde Visual Basic programcileri ODBC üzerinden AS/400 yada diğer bilgisayar sistemlerine bağlanabilirler.

Visual Basic'in standart veritabanı motoru "Jet"dir. Veritabanı motoru Microsoft Access veritabanlarına erişim için geliştirilmiştir. Jet'in verilere erişim tekniği DAO'dur. DAO ile veri erişimi Jet üzerinden olurken, RDO ile ODBC verilere doğrudan erişim sağlanır.

Yerel mi? Uzak mi? (Local mi? Remote mi?)

Eğer verilere dosya sistemi aracılığıyla erişiyorsanız "yerel verilerden" söz ediyorsunuz demektir. Diğer bir deyişle uygulamanın doğrudan ulaştığı veriler yerel verilerdir. Uzak veriler ise uygulamanın yani sıra diğer bir işlem ile ulaşılan verilerdir. Bu işlem genellikle DBMS olarak adlandırılır. Bir diğer kavram "client/server" dir. Client/server uygulamalarının amacı da uzak verilere erişmektir.

Client/Server sistemde veritabanı motoru (veritabanı işlemlerini düzenleyen kütüphane) merkezi bir server üzerindedir. Bir veritabanı motoru çok sayıda client'a aynı anda hizmet verir. Örneğin SQL-Server gibi. Uzak veritabanı ise veritabanı motorunun aynı bilgisayarda, sadece farklı bir bilgisayarda, sadece verinin farklı bir bilgisayarda olması anlamına gelir.

Cursor Tipleri

Bir veri erişim nesnesinin cursor tipi (gösterici tipi) verilere erişimin şeklini belirlemek ve erişimin performansını artırmak için kullanılan bir öğedir. Bir veri erişim nesnesi yaratılacağı zaman dört değişik gösterici (cursor) tipinden birisi kullanılabilir.

- Dynamic Cursor
- Keyset Cursor
- Static Cursor
- Forward-Only Cursor

Dinamik Gösterici (dynamic cursor), diğer kullanıcılar tarafından yapılan eklemelerin, değişikliklerin ve silmelerin görülmesini sağlar ve veri seti üzerinde yapılacak tüm hareketlere izin verir.

Anahtar Gösterici(keyset cursor), diğer kullanıcıların eklediği kayıtların görülmesini engeller. Recordset içindeki her türlü harekete izin verir. Diğer kullanıcıların yaptıkları veri değişiklikleri görülebilir.

Sabit Gösterici(static cursor), ise belli bir verinin bulunması yada raporların üretilmesi için kullanılacak veri setinin değişmeyen bir kopyasını yaratır. Veri nesnesi üzerinde her türlü harekete izin verir. Diğer kullanıcılar tarafından yapılan eklemeler, değişiklikler ve silmeler görünmezler.

Sadece-İleri Gösterici(forward-only cursor), ise sabit göstericiye benzer bir şekilde çalışır. Fakat kayıtlar arasında sadece ileriye doğru hareket etmeye izin verir.

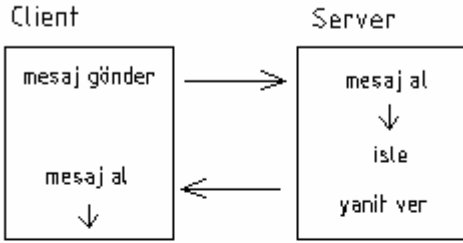
Veri nesnesi(recordset) açılmadan önce onun CursorType özelliği ayarlanır. Eğer hiçbir gösterici tipi belirtilmezse varsayılan gösterici kullanılır. Örneğin ADO veri erişim yöntemlerinde forward-only gösterici tipi varsayım olarak kullanılır.

Veritabanı işlemlerinde zaman uyumluluk özelliği vardır. Özellikle client/server uygulamalarda bu işlem türleri çok önemlidir. İki zaman uyumluluk işlemi yapılır.

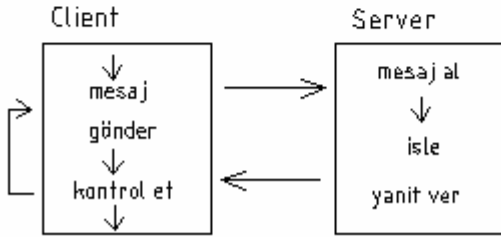
- Synchronous(zaman uyumlu)
- Asynchronous(zaman uyumsuz)

Zaman uyumlu işlem yerine getirilmesi ve bu arada başka işlemler yapılmaması anlamına gelir. Zaman uyumsuz işlem ise birçok değişiklik işleminin bir kuyruk içinde sırayla yapılmasıdır.

Synchronous(zaman uyumlu)islem



Asynchronous(zaman uyumsuz)islem



Zaman uyumlu ve zaman uyumsuz işlemlerin işleyisi

Zaman uyumsuz olan bir döngü:

```
Set r_set=vt.OpenRecordSet("Select * from ogr")
While Not r_set.EOF
    r_set.Edit
    r_set.Update
    r_set.MoveNext
Wend
```

Zaman uyumlu işlemlerde veri düzenlenirken (update) döngü devam etmez. Fakat yukarıdaki kodda kayıt seti üzerindeki tüm kayıtlar değiştirilmektedir.

Visual Basic'te Veritabanları Seçenekleri

Visual Basic ile veritabanlarına erişmek için değişik-belli seçenekler vardır. Bunlar verilerin yapısı ve kullanımı bakımından farklılık gösterir.

Mesela Data Control kolay bir veritabanı uygulamasıdır ayrıca Microsoft Access'e erişmek için Data Control'ü kolaylıkla kullanabilirsiniz.

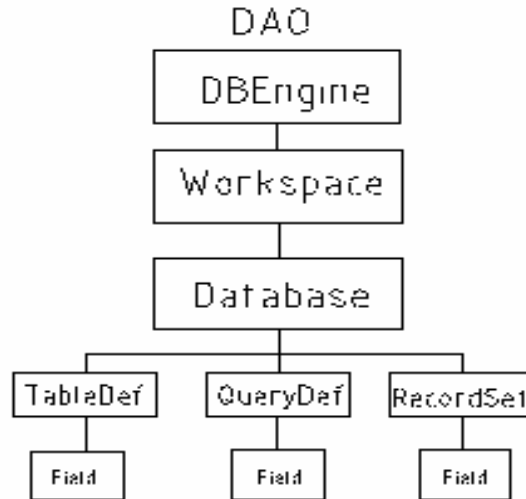
Remote Data Control(RDO) ise özellikle uzak veritabanlarına erişim için kullanılan bir veri kontrolüdür. Ayrıca 32-bit uygulamalar sunan RDC, yüksek performans sağlar.

DAO

Veritabanı programlamanın temeli DAO oluşturmak ve kullanmaktır.

DAO **Microsoft Jet database engine**'i kullanarak **workspace**, **Database**, **TableDef**, **Field**, **Recordset** ve **QueryDef** gibi veri erişim bileşenleriyle bağlantı sağlar.

DAO bir hiyerarşi yapısına sahiptir. Bu yapının en üst kısmında **Microsoft Jet database engine** yer alır.



- Bir Database uygulamasında Microsoft Jet database engine bir yada daha fazla nesne içerebilir.
- Her bir Workspace bir yada daha fazla Database koleksiyonu içerebilir.
- Her bir Database bir yada daha fazla TableDef nesnesi içerebilir.

- Her bir TableDef bir yada daha fazla Field(alan) içerebilir.
- DAO Microsoft Access tarafından oluşturulan .MDB formatındaki dosyaları kullanır.

DAO ve Jet

Data Access Objects, ODBC API üzerinde bir üst katmandır. DAO'nun, Jet'in SQL özellikleri vardır. DAO'nun temel özelliği Data Control üzerinde çok sayıda işlem yapılabilmesidir.

DAO ve Jet aynı şeyler değildir. Ancak birbirlerine bağlıdır. DAO bir veri erişim tekniği olduğu için doğrudan kullanılır. Ancak Jet doğrudan kullanılmaz. Ayrıca DAO Jet'e ulaşmak için de kullanılan bir veri erişim yöntemidir. DAO ve Jet ODBC'ye bağlanırken performans açısından farklılıklar gösterirler.

Jet sayesinde Access veritabanlarına erişilir, ISAM ve ODBC veri kaynaklarına erişim sağlanır.

DAO nesne Modeli

DAO veri erişim yönteminin iki nesne modeli vardır. Birincisi Jet sayesinde olmaktadır. Diğer yöntem ise ODBC ile verilere erişimdir. Bu yöntem ODBC Direct de denir.

DAO Programlama

Bir veritabanı uygulamasını geliştirmek için belli aşamalarındaki işlemleri yapmak gerekir.

- Kullanıcı arabirimi
- Veritabanı motoru
- Verileri saklamak ve işlemek

Veritabanı programlamanın ilk aşaması kullanıcı arabirimi ve aradaki ilişkinin düzenlenmesidir. Veriler bir Access veritabanı olarak .mdb olarak saklanırlar. Diğer veritabanlarında uzantisi farklı olabilir.

Client / Server ve Uzak Veritabanlarına Erişim

Client/Server sistemlerde veritabanı motoru merkezi bir server üzerindedir. Bu veritabanı motoru çok sayıda client'a aynı anda istedikleri hizmeti verebilir. Uzak veritabanı ise veritabanı motorunun aynı

bilgisayarda, sadece verinin farklı bir bilgisayarda olduğu anlamına gelir. Yani Microsoft Jet Database Engine, client/server bir veritabanı yöneticisi değildir.

Nesne Hiyerarşisi

Nesne hiyerarşisi bir nesnenin diğerini içermesi demektir. Nesneleri içeren nesneye de "collection" denir. DAO hiyerarşik bir nesne modelidir.

Bir nesne modeli vermek gerekirse:

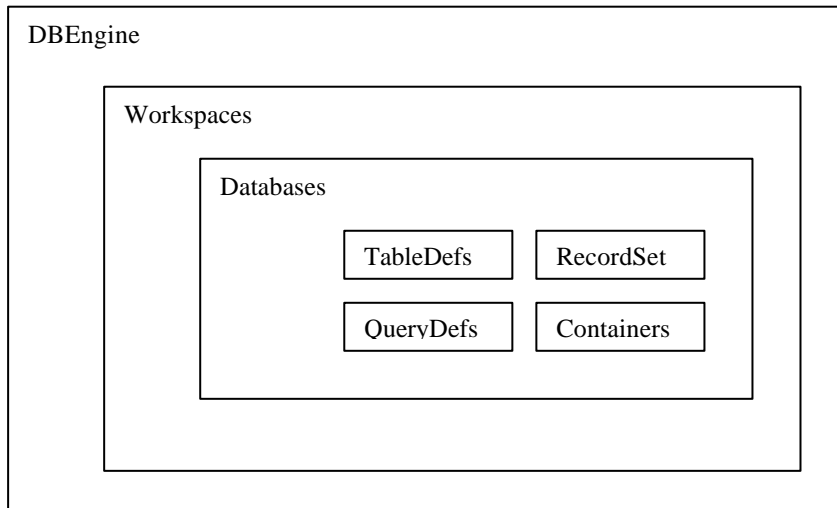
```
Dim Not As Workspace
```

Bu açıklamada Not değişkeni Workspace class'ın bir nesnesidir. Nesnenin hiyerarşisi nokta ile ayrılır.

```
DBEngine.Workspaces(0).Database(0).TableDefs(0).Fields("öğrenci")
```

Yukarıdaki ifadede TableDefs koleksiyonunun birinci TableDef'ini, Databases koleksiyonunun birinci Database'ini ve Workspace koleksiyonunun birinci Workspace'ini belirtir.

Dao nesnelerinin sematik gösterimi:



Aktif olan veritabanını göstermek için aşağıdaki komutlar kullanılır.

```
Dim w_space As Workspace
Dim data As Database
Set w_space=DBEngine
Set data =w_space.Databases(0)
```

Bir tablo tanımlanması:

```
Dim data As Database
Dim tablo As TableDef
Dim alan As Field
Dim sıra As Index
Set data=DBEngine.Workspaces(0).Databases(0)
Set Tablo=data.openRecordSet("ogr")
```

DBEngine

DBEngine DAO'nun nesne modellemesinde kullandığı bir bileşendir.

DBEngine Dao hiyerarsisinde diğer tüm nesneleri içerir ve aynı zamanda nesneleri kontrol eder.

Properties

DefaultType: bir sonraki workspace nesnesi oluşturulduğunda onun tipini ayarlar.

```
DBEngine.DefaultType=dbUseODBC
```

Aşağıdaki değerlerden birini alır:

dbUseJet

Microsoft Jet database engine'e bağlantı sağlayan bir workspace nesnesi oluşturur.

dbUseODBC

ODBC veri kaynağına bağlantı sağlayan bir workspace oluşturur.

DefaultUser

Oluşturulacak workspace nesnesinin varsayılan kullanıcı adını belirler.

DefaultPassword

Workspace nesnesi oluşturulduğunda varsayılan şifreyi belirler.

```
DBEngine.defaultuser="AKTAN"
DBEngine.Default.Password="SIFRE"
```

IniPath

Jet database engine için Windoes registreyinde kullanılan yol bilgilerini belirler.

LoginTimeout

ODBC veri kaynagina baglanti saglamaya çalisildiginda bir hata olusmadan önceki süreyi belirler.

Metodlari

BeginTrans.CommitTrans.Rollback

Veri tabani üzerinde degisiklik yaparken çikabilecek aksilikler düşünülerek önlem alınmalıdır. Mesela bir grup veri degistirilmeye baslandıktan sonra tüm veriler basarili bir sekilde degistirilmeyebilir. Bu tür durumlarla karsilasmamak için veri tabanında degisiklige baslamadan önce **BeginTrans** metodu kullanilir. Böylece yapılan degisiklikler **CommitTrans** metodu veriverilmeden etkili olmayacaktır. Eger işlem basarili bir sekilde tamamlanırsa **CommitTrans** kullanılarak degisiklikler aktif hale getirilir. Eger işlem tamamlanmazsa, hatalar çikarsa veya iptal edilirse **Rollback** metodu çağırılarak degisiklikler iptal edilir ve BeginTrans metodundan önceki duruma dönülür.

Ancak bazı veritabanlari(paradox) bu yöntemleri desteklememektedir. Bu gibi durumlarda komutlarda bir hata olusmaz ancak metodlar da görevini yerine getirmez. Eger kullandiginiz veritabani dosyasinin bir metodu destekleyip desteklemedigini bilmiyorsanız bunu **Transactions** özelligi ile öğrenebilirsiniz. Eger **True** ise meto destekleniyordur.

CreateDatabase

Yeni bir veritabani dosyasi olusturmak için kullanilir.

```
Dim Yeni As DATABASE
Set Yeni=wrkDefault.CreateDatabase("dosya.mdb",
dbLangGeneral,dbEncrypt)
```

CreateWorkspace

Yeni bir Worksapce nesnesi olusturur.

```
Dim yeni_wrkspce As Workspace
Set deni_wrkspce=CreateWorkspace("ODBCWorkspace","admin","",
dbUseODBC ) Workspaces.Append wrkODBC
```

Microsoft Visual Basic 6.0

OpenDatabase

Bir veritabanı açar.

```
Dim v_taban As workspace  
Dim data As Database  
Set data=v_taban.OpenDatabase("dosya.mdb".True)
```

RegisterDatabase

Bir ODBC veri kaynağının bilgilerini Windows registreyine aktarır.

RepairDatabase

Microsoft Jet Database'i tamir etmeye,hataları onarmaya çalışır.

```
DbEngine.RepairDatabase "dosya.mdb"
```

Workspace

Workspace nesnesi Visual Basic uygulamalarının veritabanı ile nasıl haberleşeceğini belirler.

Bir Workspace nesnesi DBEngine.Workspaces(0) biçiminde tamamlanır.

Metodları

Close

Workspace nesnesini kapatır.

```
Dim dosya As Recordset  
Dosya.close
```

CreateDatabase

Yeni bir Database nesnesi oluşturur.

```
Dim yeni_wrkspc As workspace  
Dim yeni_dbase As DATABASE  
Set wrkDefault=DBEngine.workspaces(0)  
Set yeni_dbase =yeni_wrkspc.createDatabase("dosya.mdb",  
dbLangGeneral,dbEncrypt)
```

CreateGroup

Yeni bir user nesnesi olusturur.

```
Dim var_wrkspc As Workspace
Dim yeni_grup As DATABASE
Set var_wrkspc =DBEngine.workspaces(0)
Set yeni_grup=var_wrkspc.CreateGroup("dosya","0418")
```

CreateUser

Yeni bir User nesnesi olusturur.

```
Dim yeni_user As User
Dim var_wrkspc As Workspace
Set var_wrkspc =DBEngine.Workspaces(0)
Set yeni_user =var_wrkspc.CreateUser("Zühre_
Akman","0418","sifre1")
```

OpenConnection

ODBC veri kaynagi için bir baglanti açar.

OpenDatabase

Aktif workspace ile veritabani açar.

```
Dim wrkspc As Workspace
Dim d_base As Database
Set wrkspc =CreateWorkspace("", "admin ", "", dbUsejet)
Set d_base=wrkspc.OpenDatabase("dosya.mdb", True)
```

Database

Database nesnesi açık olan veritabani dosyalarına erişim sağlar.

Metodlari

Close

Aktif Database nesnesini kapatır.

CreateProperty

Microsoft Visual Basic 6.0

Yeni kullanıcı tanımlı olan bir property nesnesi oluşturur.

CreateTableDef

Yeni bir **TableDef** nesnesi oluşturur.

Execute

Belirlenen SQL ifadesini çalıştırır.

NewPassword

Database nesnesine yeni bir şifre atanır.

OpenRecordset

Yeni bir Recordset nesnesini oluşturarak onu Recordset koleksiyonuna ekler.

TableDef

TableDef nesnesi veri tabanında bulunan tablolara erişmek ve onları kullanmak için kullanılır.

Properties

RecordCount

Tablodaki kayıt sayısını belirler.

Replicable

Nesnenin kopyasının alınıp alınmayacağını belirler.

ReplicaFilter

Kopyalama işleminden hangi kayıtların alınacağını belirler.

SourceTableName

Baglantisi saglanmis tablonun adini belirler.

Updatable

DAO nesnesinin güncellenebilir durumunu belirler.

ValidationRule

Bir tablodaki alana girilecek veri için gerekli kurallari belirler.

ValidationText

Kuralin,sartin gerçekleşmemesi durumunda görüntülenecek mesajı belirler.

Metodlari

CreateField

TableDef için yeni bir Field nesnesi oluşturarak onu field koleksiyonuna ekler.

CreateIndex

TableDef nesnesi için yeni bir index nesnesi oluşturarak onu index koleksiyonuna ekler.

CreateProperty

TableDef için yeni bir property nesnesi oluşturur.

OpenRecordset

TableDef için yeni bir Recordset nesnesi oluşturarak onu Recordset koleksiyonuna ekler.

RecordSet

Recordset nesnesi ana tabloda yer alan kayıtları temsil eder. Recordset nesnesi üç ayrı tipe sahiptir:

Table

Dynaset

Snapshot

Bu üç tip birbirinden farklı özelliklere sahiptir.

- Table

Bu tip yerel veritabanındaki aktif tabloya erişilebilir. Burada tablo değişkeni başka bir veritabanındaki tabloya erişemez.

Veri sıralamak, index oluşturmak ve **seek** metodunu kullanmak gibi işlemler ancak **table** tipli bir **recordset** ile gerçekleşir.

- Dynaset

Dynaset tipine sahip bir recordset nesnesi hem yerel, hem sorgu sonucu ve hemde başka bir tabloya erişebilir. Birden fazla tablo aynı anda güncellenebilir.

- Snapshot

Snapshot tipine sahip bir recordset nesnesi dynaset'e benzer fakat sabit veri içerir. Bu tipe sahip bir Recordset nesnesi güncellenemez. Bir sorgu sonucunda elde edilen tüm verilerin kopyasını içerir.

Eğer verilere sıralamak ve index oluşturmak istiyorsanız **table** tipini kullanmamız gerekir.

Eğer sorgu sonucunda elde ettiğiniz verileri güncellemeyecekseniz **dynaset** tipini kullanırız.

Eğer güncelleme yapmaksızın en yüksek hızı elde etmek istiyorsak o zaman **snapshot** tipini kullanmamız gerekir.

Recordset Değişkeni Oluşturmak

OpenRecordset metodu recordset değişkeni oluşturmak için kullanılan en temel metodlardır. **OpenRecordset** metodu **Database**, **TableDef** ve **QueryDef** için vardır.

Set

degisken=veritabani_adi.OpenRecordSet(Kaynak,Tip,Seçenekler)

Seklinde tanımlanır.

Diğer nesneler için ise kullanım şekli şöyledir:

Set degisken=nesne_adi.OpenRecordSet(tip[,seçenekler])

Burada **nesne_adi** olarak verilen ifade **DataBase,TableDef,QueryDef** ve **RecordSet** nesnelerine karşılık gelmektedir.

Kaynak parametresi TableDef veya QueryDef nesnelerini temsil etmektedir.

Type parametresi ise RecordSet'in tipini belirler.

```
DB_OPEN_TABLE  
DB_OPEN_DYNASET  
DB_OPEN_SNAPSHOT
```

Seçenekler parametresi ise bir yada daha fazla sabiti temsil eder. Bununla çok kullanıcıli veri erişimi yapılır.

```
Dim prog_db As DataBase  
Dim progset As RecordSet  
Set prog_db=DBEngine.Workspaces(0).Database(0)  
Set progset=prog_db.OpenRecordSet("dosya",DB_OPEN_TABLE)
```

Properties

BOF,EOF

```
Dim wspace AS Workspace  
Dim data As Database  
Dim r_set As Recordset  
Set wspace =DBEngine.Workspace(0)  
Set data =OpenDatabase("dosya.mdb")  
Set r_set =data.OpenRecordset("not")  
With r_set  
Do Until.EOF      'dosya sonuna kadar  
If not>50 then  
Printer.print !adi ;!soyadi ,!not  
End if  
.MoveNext  
Loop  
End With
```

Bu özellikler dosya sonu dosya basi kontrolleridir. EOF dosya sonu, BOF ise dosya basi kontrolü yapılmak istendiginde kullanılır.

Bookmark

Microsoft Visual Basic 6.0

Veritabanında kayıtların konumunu belirlemek için kullanılır. Bookmark ile aktif olan kaydın konumu bir degiskene atabilir,gerektiginde geri çağirip kullanabiliriz.

```
Dim w_space As Workspace
Dim data As Database
Dim r_set As Recordset
Dim bas As variant

Set w_space=DBEngine.workspace(0)
Set data =OpenDatabase("dosya.mdb")
Set r_set =data.OpenRecordSet("kayit")

With r_set
bas=.bookmark      'kaydin saklanmasi
Do Until.EOF
.MoveNext
Loop
.bookmark=bas      'eski yerine tekrar döndürülüyor.
End If
```

Bookmarkable

Kayit setinin Bookmark islemini onaylayip onaylamadigini kontrol etmek için kullanilir.

LastModified

Recordset nesnesinin en son yenileme yerini belirler.

CacheSize

ODBC veri kaynagindan alinacak ve saklanmis kayit sayisini belirler.

Connection

Recordset nesnesine sahip olan Connection nesnesini atarlar.

DataCreated

Bulunan nesnesinin olusturuldugu tarih ve saati gönderir.

EditMode

Bulunan kaydın bildi giriş durumunu belirler.

Fitler

Kayıtları filtrelemek için gerekli şartı ayarlar.

Name

DAO nesnesi için kullanıcının tanımladığı adı belirler.

NoMatch

Seek ve **Find** metodlarından sonra bir kaydın bulunup bulunmadığını belirler.

```
Data_Adi="C:Belgelerim\dosya.mdb"
t_adi="işçi"
Set w_space =DBEngine("dbTemp","admin","")
Set data=w_space.openDatabase(Data_adi)
Set kayitseti=db.OpenRecordSet(t_adi,dbOpenTable)
Kayitseti.index="Sirano"
Kayitseti.Seek "=",1
    Do Until Kayitseti.NoMatch
        Kayitseti.edit
        Kayitseti("Sirano")=2
    Kayitseti.Update
    Kayitseti.Seek "=",1
Loop
Kayitseti.Close
```

RecorCount

Aktif Recordset nesnesi içindeki kayıt sayısını belirler.

```
Text1.Text="Kayıt Sayısı" & kayitseti.RecordCount
```

Sort

Aktif Recordset nesnesi içindeki kayıtları sıralama işlemini yapar.

```
Kayitseti.Sort="(adsoy)"
```

Microsoft Visual Basic 6.0

ValidationText

Bir veri alanına bilgi giriş şartını belirler. Mesela sadece 1 ve 50 arası sayıların kullanılmasını istediğimizde aşağıdaki gibi bir kod yazılır.

```
Sayi.ValidationText="Gireceğiniz Sayı 1 ve 50 arasında olmalıdır"
```

Methods

AddNew

Yeni kayıt eklemek için kullanılır.

```
Option Explicit  
Dim d_adi As String  
Dim t_adi As String  
Dim w_space As Workspace  
Dim data As Database  
Dim r_set As RecordSet  
r_set.AddNew
```

Cancel

Tüm yapılan işlemleri iptal eder.

CancelUpdate

Bekleyen güncellemeleri iptal eder.

Clone

Bulunan Recordset nesnesinin bir kopyasını oluşturur.

Close

Çalıştırılan kayıt setini kapatır.

```
Private Sub Form_Unload(Cancel As Integer)  
r_set.close  
data.close  
set r_set=Nothing  
set data =Nothing  
End Sub
```

Delete

Aktif kaydın silinmesi islevini yapar.

Edit

Aktif kaydı düzenleme moduna alır.

FindFirst

Belirlenen sarta uyan ilk kaydı bulur.

```
Dim ara As String  
Ara=" adsoy"="Ayse Sahin"  
r_set.FindFirst ara
```

FindLast

Belirlenmiş olan sarta uyan son kaydı bulur.

FindNext

Belirlenmiş olan sarta uyan sonraki kaydı bulur.

FindPrevious

Belirlenmiş olan sarta uyan önceki kaydı bulur.

Move

Aktif olan kaydın yerini Recordset nesnesinde hareket ettirir.

MoveFirst

Recordset nesnesi içindeki ilk kayda yerlesir.

MoveLast

Recordset nesnesi içindeki son kayda yerlesir.

MoveNext

Recordset nesnesi içindeki sonraki kayda yerlesir.

MovePrevious

Recordset nesnesi içindeki önceki kayda yerlesir.

```
r_set.MovePrevious
```

Requery

Aktif olan veritabani bilgileriyle tabloyu yeniden gözden geçirir,sorgulama islemini yapar.

Seek

Index ile beraber kullanılan sart kriterine uyan kaydi bulur.

```
Private Sub List1_Click  
Dim ara  
r_set.Index="PrimaryKey"  
ara="10"  
r_set.Seek"=",ara  
End Sub
```

Update

AddNew ve Edit metodlari kullanildiktan sonra kullanilir. Yapilan degisiklikleri kaydeder.

```
Private Sub Kayit_Click  
r_set.AddNew  
r_set!(adsoy)=text1  
r_set.!(numara)=text2  
r_set.!(vize)=text3  
r_set.!(final)=text4  
r_set.Update  
End Sub
```

Field

Veritabanlarını meydana getiren en temel tasarlardan biridir. Her tablo değişik alanlardan(field) oluşur.

Properties

AllowZeroLength

Veri alanının uzunluğunun sıfır olup olmayacağını belirler.

```
Dim data As Database
Dim tdfogr As TableDef
Dim alan As Field

Set data=OpenDatabase("dosya.mdb")
Set ogr =data.TableDefs("ogr")
Set alan=tdfogr.CreateField("Numarasi",dbtext,12)
Alan.AllowZeroLength=True
Tdfogr.Fields.Append alan
```

DataUpdatable

Nesnenin güncellenebilme durumunu belirler.

DefaultValue

Alan için varsayılan değer belirlenir. Kullanıcının isteğine göre değişebilir.

```
Tdfogr.Fields!Not.DefaultValue="50"
```

FieldSize

Kullanılacak olan alanın büyüklüğünü belirler.

Name

DAO nesnesi için kullanıcı tanımlı adı belirler.

OrdinalPosition

Fields koleksiyonuna alanin sıra numarasini belirler.

Required

Bir alana bilgi girisinin zorunlu olup olmadigini belirler. Yani bir text kutusunun doldurulmadan geçilmemesi gibi.

Size

Alanin uzunlugunu belirler.

```
Dim data As Database
Dim tdfogr As TableDef
Dim alan As Field
Set alan=tdfogr.CreateField("Numara")
alan.Type=dbText
alan.Size=20
tdfogr.Fields.Append alan
```

SourceField

Aktif field nesnesi için veri kaynaginda yer alan alni belirler.

SourceTable

Aktif field nesnesi için veri kaynaginda yer alan tablo adini belirler.

Type

Nesnenin hangi islem tipinde oldugunu belirler.

ValidationRule

Bir alana girilecek olan bilginin sinirlari belirlenir. Mesela sirket personeli kayitlari kontrolü yapildiginda kayit disinda bir numara girilmesi engellenmis olur.

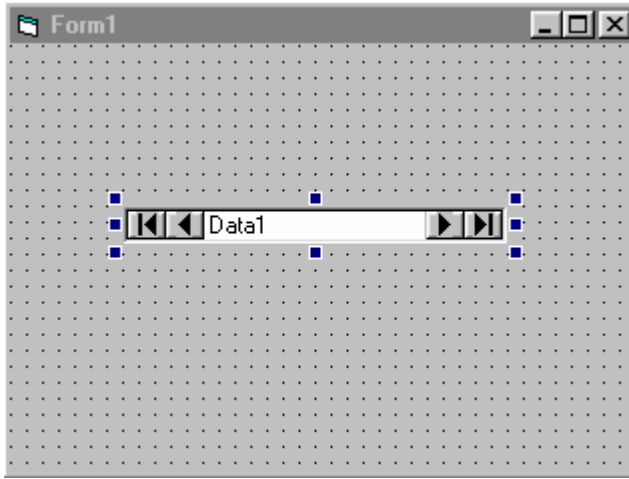
ValidaionText


Belirlenen sartlarin disinda bilgi veya mesaj girildiginde ekranda görünülenecek hata mesajini belirler.

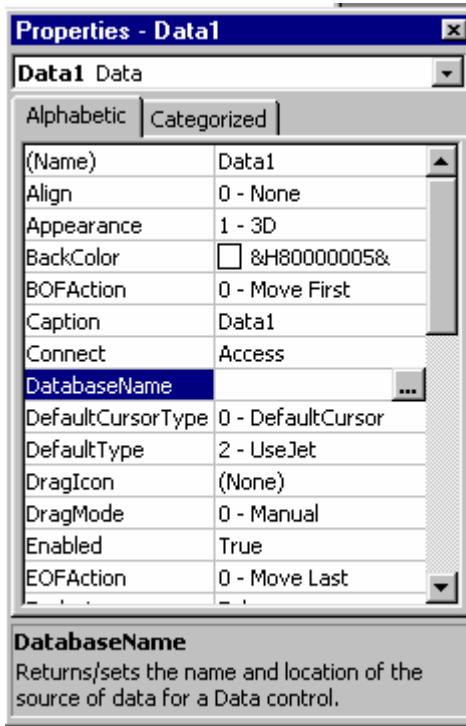
Value

Alanda sakli olan degeri gönderir.

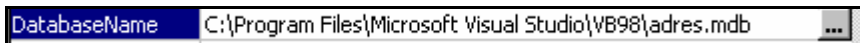
DAO'nun Forma Eklenmesi



Toobox penceresinden  resimli düğmeye bastigimiz zaman veri tabani nesnesi olan **Data1** eklenmis olacaktır. Bu nesneye ait özellikler properties penceresinden degistirilebilecektir. Data nesnesinin üzerinde olan oklara tiklandigi zaman veri tabani dosyasindaki kayitlar sıra ile dolasilmis olacaktır. Forma eklenen veri tabani nesnesi ile veri tabani nesnesi birbirine baglanmadan Data nesnesinin hiçbir görevi olmayacaktır.

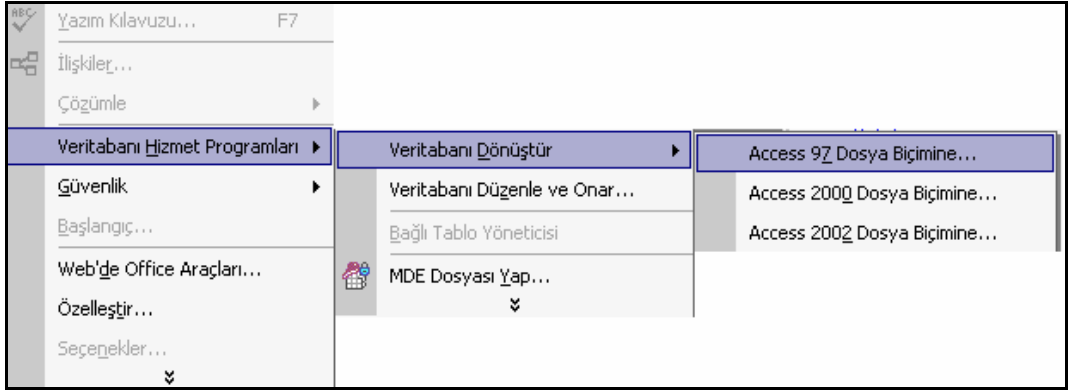


Forma eklenmiş olan veri tabanı nesnesi ile harddiskteki veri tabanı dosyasının bağlantısı bu **"Database Name"** seçeneğinden yapılır. Yan tarafında bulunan üç noktaya tıklandığında veri dosyasının bulunduğu sürücü-dizin-dosya seçimi yapılır.



Bu bağlantı henüz tamamlanmamış bir bağlantıdır. Çünkü bilindiği gibi bir dosyanın içinde birden fazla Tablo olabilir. Bu nedenle hangi tablo üzerinde işlem yapacağımızı da Data'ya belirtmemiz gerekmektedir. Bunun için;





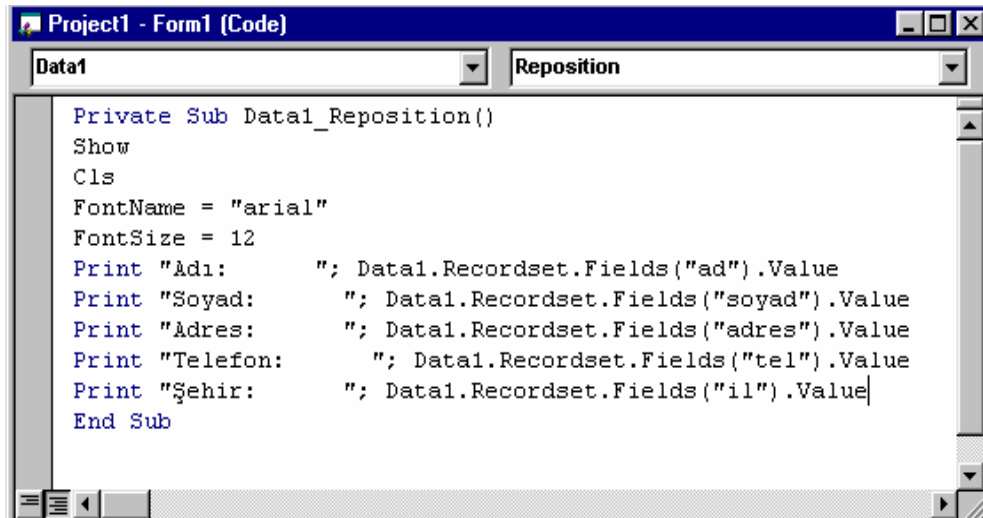
Yukarıda gördüğünüz gibi veri tabanı dönüştürme işlemleri yapılmalıdır. Kayıtlı olan Access programınızı açarak dönüştürme işlemlerinizi sırayla tamamlamanız gerekmektedir.

Daha sonra veri tabanı dosyasının hangi tablosundaki bilgiler okunmak isteniyorsa bu **"RecordSource"** değişkenine aktarılmalıdır.

RecordSource bölümünde dosyamızda yan tarafta bulunan okatıldığımız zaman kayıtlı olan tüm tablolar görüntülenecektir. Fakat biz sadece bir tabloda işlem yapabiliriz. Eğer bir dosyada birkaç tablomuz ve programımızın değişik yerlerinde bunları kullanmak istiyorsak her bir tablo için bir Data bulundurmamız ve onları da aynı şekilde bağlamamız gerekmektedir.

Şimdi bağlantılar tamamlanmış olmaktadır.

Bunu küçük bir programla pekiştirmek istersek;

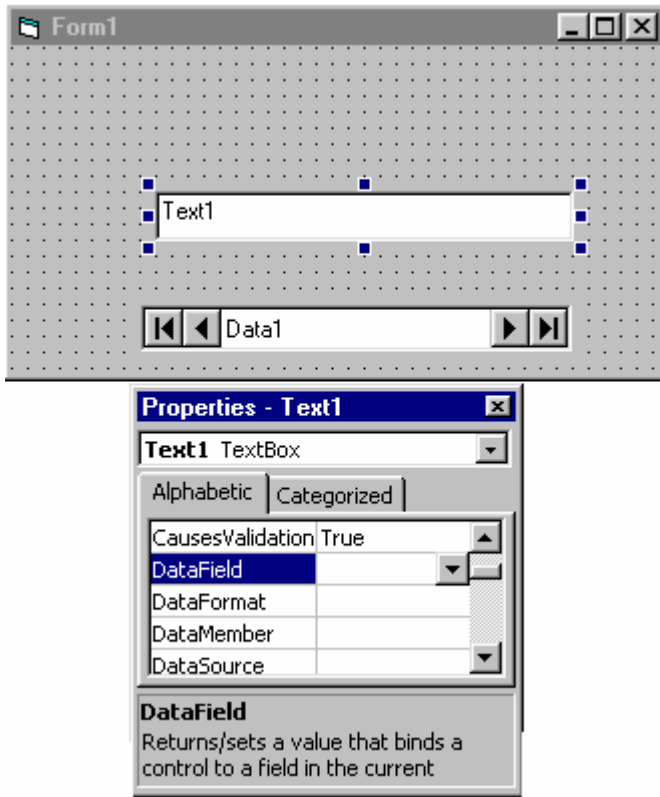


Bir MDB uzantili dosya açıldığında kayıt okuma kafası ilk olarak birinci kayda bakacaktır. Tablonun açılması ile birlikte ilk kayda konumlanması sırasında Data nesnesinin **Reposition** olayı meydana gelir. Bu olayı daha iyi görmek için yukarıdaki kodları yazıp programı çalıştırabilirsiniz. Tabii bunun için veri dosyanızda birkaç kaydınızın bulunması gerekir. Bu tablodaki kayıtlara **Recordset** denilmektedir. Kaydın hangi alanı ile işlem yapmak istiyorsanız, bunu belirlemek için de **Fields** deyimi kullanılır. Alanın içeriği de **Value** değişkeni ile tespit edilir.

Programimiz çalıştırıldığında ise aşağıdaki görüntüyü alacaktır:

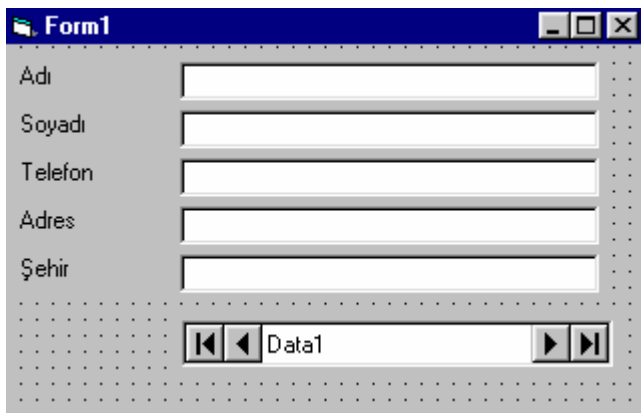


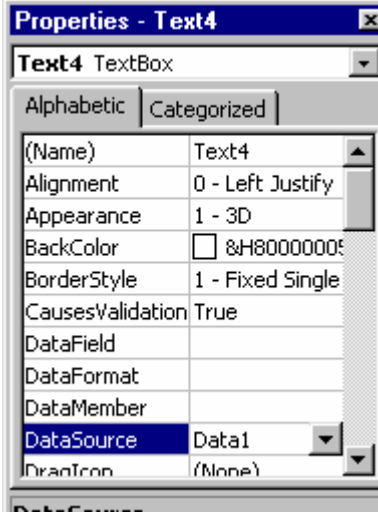
Text Özellikleri



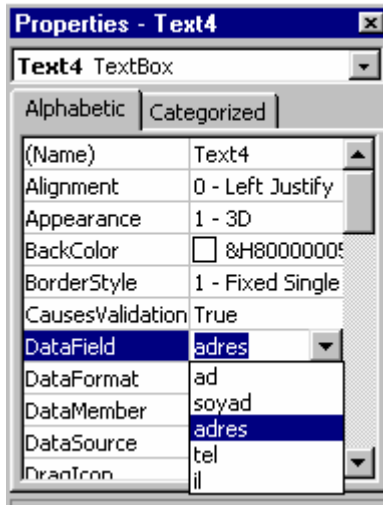
Forma eklenen Data nesnesi ile Textbox nesnesi arasında bağlantı kurmaya yarar. Data nesnesi ile birlikte Image, Label, PictureBox ve onay kutusu kontrolleri kullanılır.

Şimdi bu Text kontrollerini bir örnekle açıklayalım:





İlk olarak form oluşturulur, bundan sonra her text kutusunun **"DataSource"** degiskeni ile Data1'in baglantisi saglanir ve



Her Text kutusunun kendi ilisigi olan bölüm ile baglanmasi saglanir.. Bunun için **"DataField"** degiskenine tiklanir ve tablonun ilgili bölümü seçilir.

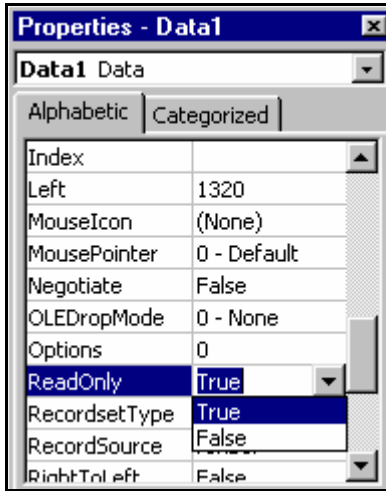
Yani baglantilari tamamlamak için DataSource ve DataField seçimleri yapılır. Asagidaki görünüm elde edilir:

The screenshot shows a Visual Basic form titled 'Form1'. It contains five text boxes for data entry: 'Adı' (Name) with the value 'Atiye', 'Soyadı' (Surname) with 'Yilmaz', 'Telefon' (Phone) with '6341236', 'Adres' (Address) with 'Sancaktepe Mah.', and 'Şehir' (City) with 'İstanbul'. At the bottom of the form, there is a 'Data1' control with navigation buttons (back, forward, first, last).

Değişiklik yapmak yani diğer kayıtları görmek için Data nesnesinin oklarına tıklanır..

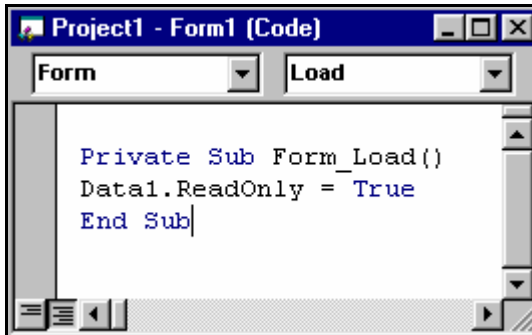
The screenshot shows the same Visual Basic form titled 'Form1', but with different data entered: 'Adı' is 'Semiha', 'Soyadı' is 'Ateş', 'Telefon' is '6541789', 'Adres' is 'Dağlarbaşı Mah.', and 'Şehir' is 'İzmir'. The 'Data1' control and its navigation buttons are still present at the bottom.

Böylece Vbasic üzerinde tek satır yazmadan kayılar arası geçiş veya her türlü işlemi yapmak mümkündür.



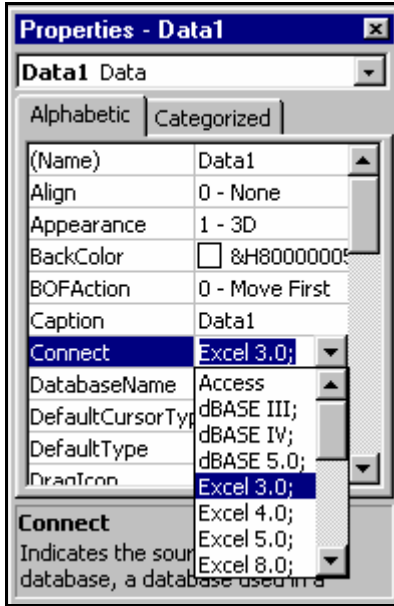
Veri tabanı kayıtları üzerinde değişiklik yapılması istenmiyorsa Properties penceresinden Data nesnesine ait "ReadOnly" değişkenine True değerini atamamız yeterli olacaktır. İlk olarak ReadOnly seçeneği False olarak yüklüdür yani her türlü değişiklik yapılabilir fakat şimdi program çalıştırıldığında hiçbir değişiklik yapılamayacaktır.

Yada program kodu ile bunu kontrol ettirmek istersek



Visual Basic uygulamaları dahilinde MDB uzantılı Access veri tabanı dosyalarından başka dBaseIII, dBaseIV, Paradox ve Foxpro 2.5 dosyalarına erişim sağlama imkanı da bulunmaktadır. Data nesnesi aracılığıyla erişim sağlanan veri tabanı dosyasının tipi veya ait olduğu veri tabanı programının adı Data nesnesine ait **"Connect"** değişkeninde tutulur. Access formatında

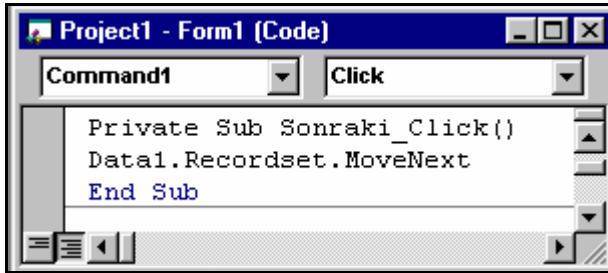
olan dosyalarda Connect degiskenine atama yapmaya gerek yoktur. Fakat digerlerinde durum böyle degildir.



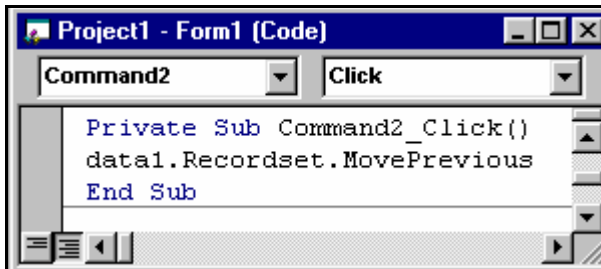
Tablodaki Kayitlar Arasinda Dolasmak

Önceki örneklerimizde kayitlar rasida dolasmak için data nesnesinin düğmelerinden yararlaniliyordu. Fakat simdi biraz görsellik biraz da programciligimiza bir seyler katmak için ek islemlerden faydalanacagiz.

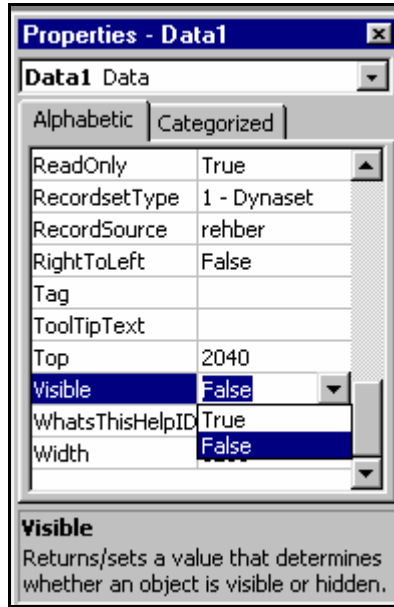
Visual Basic projesi dahilinde açilan veri tabani dosyasinin tablosu açilir açilmaz tablonun üzerinde bulunan kayit için otomatik olarak bir nesne olusturulur. Bu nesneye de **Recordset** adi verilmektedir. Buna göre de data nesnesinin üzerindeki düğmeler yardimiyla hareket etmek ve Recordset'in içeriğini degistirmek arasında bir fark yoktur. Bir sonraki kaydi görmek için **MoveNext** methodundan yararlanilir. Bir kod satirina bakarsak:



MoveNext ile bir önceki kayda konumlanırken **MovePrevious** methodu ile okuma kafası bir önceki kayda konumlandırılır.



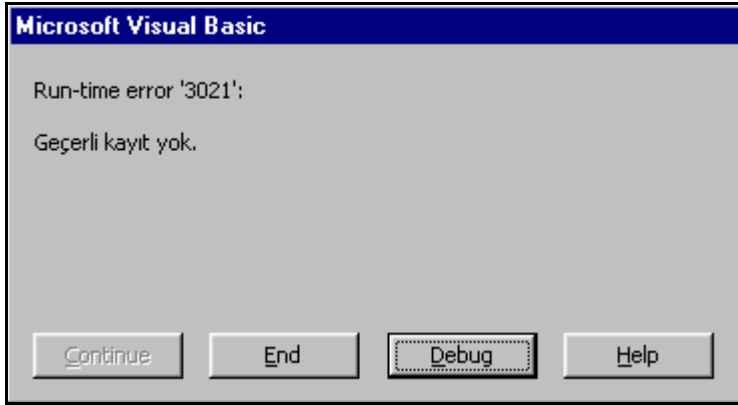
Kayıt okuma kafasını ilk kayda konumlandırmak için MoveFirst en son kayda konumlandırmak için ise MoveLast metodu kullanılır. Bu komutları öğrendikten sonra Data nesnesinin ekranda görünmesine gerek kalmamıştır. Data nesnesinin görünmemesi için Properties penceresinden "**Visible**" değişkeni False olarak ayarlanır.



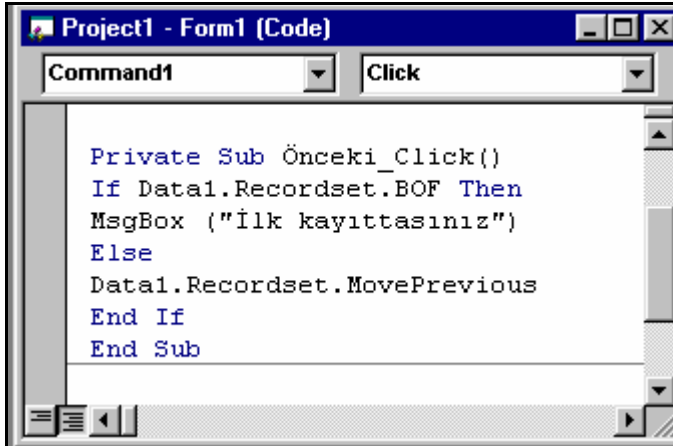
Projemiz en son haliyle çalıştırıldığında:

The screenshot shows the 'Form1' application window. It contains five text boxes for data entry: 'Adı' (Atiye), 'Soyadı' (Yılmaz), 'Telefon' (6341236), 'Adres' (Sancaktepe Mah.), and 'Şehir' (İstanbul). At the bottom, there are three buttons: 'Sonraki', 'Önceki' (highlighted with a dashed border), and 'En Son Kayıt'.

Kayıtlar arasında dolasmak için forma eklenen düğmeler kullanılmak istendiği zaman, yani program kodu yazılarak kayıtlar arası dolasılmak istendiği zaman sorunlar çıkar. İlk kaydın üzerinde iken Data nesnesine tekrar MovePrevious methodu uygulanırsa programı çalışması kırılır.



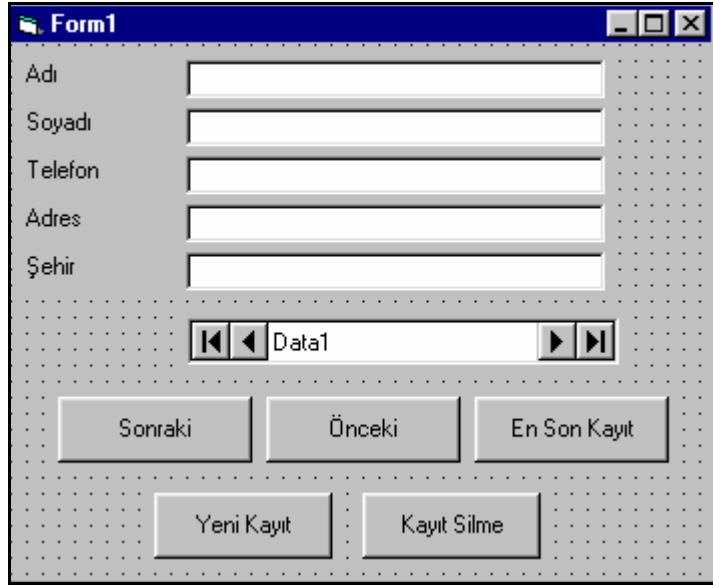
Eğer kayıtlar arası program ile dolasilmak isteniyorsa bazı önlemlerin alınması gerekmektedir. Bunun için Data nesnesinin BOF ve EOF özelliklerinden yararlanmalıdır. İlk kayıdın üzerinde iken MovePrevious metodunun kullanılmasını önlemek için:



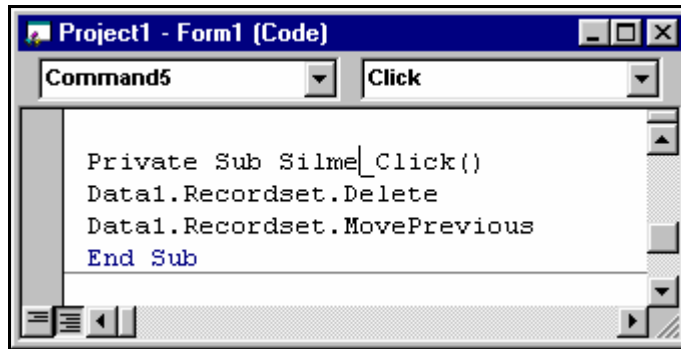
Komutları kullanılır. BOF dosya başı kontrolünü EOF ise dosya sonu kontrolünü gerçekleştirmektedirler. MoveNext metodunun kullanılmasını önlemek için EOF özelliğinden yararlanılır.

Tabloya Yeni Kayıt Girmek ve Kayıt Silmek

Bu projemiz ile adres.mdb içerisinde degisiklikler yapabiliriz. Yeni kayıt ekleme ekledigimiz kayitlari silme...Simdi formumuzu daha da genisleterek iki buton daha ekleyelim. "Yeni kayıt ve kayıt silme"

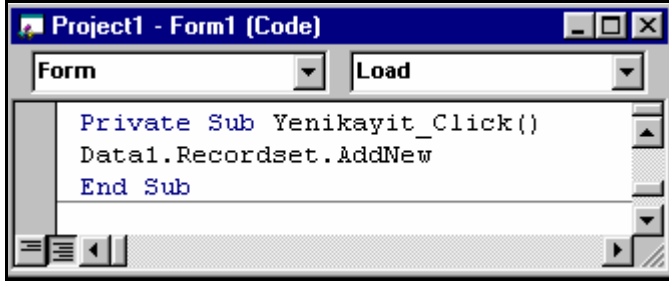


Tablonun üzerinde bulunan kaydi silmek için "Delete" metodundan yararlanilir. Delete komutunu kullandıktan sonra projemizin içeriğini görmemiz için güncelleştirmemiz "**ReFresh**" veya "**MovePrevious**" deyiminden yararlanmamız gerekmektedir.



```
Private Sub Silme_Click()  
Data1.Recordset.Delete  
Data1.Recordset.MovePrevious  
End Sub
```

Adres dosyamıza yeni kayıt eklemek için "**Addnew**" metodundan yararlanilir.



Yeni kayıt butonuna tıkladığımız zaman bir hata ile karşılaşmamak için **“Readonly”** özelliğinin **“False”** olmasına dikkat ediniz. Daha önce de anlattığımız gibi Readonly ile data üzerinde hiçbir değişiklik yapamıyorduk.

Validate Olayı

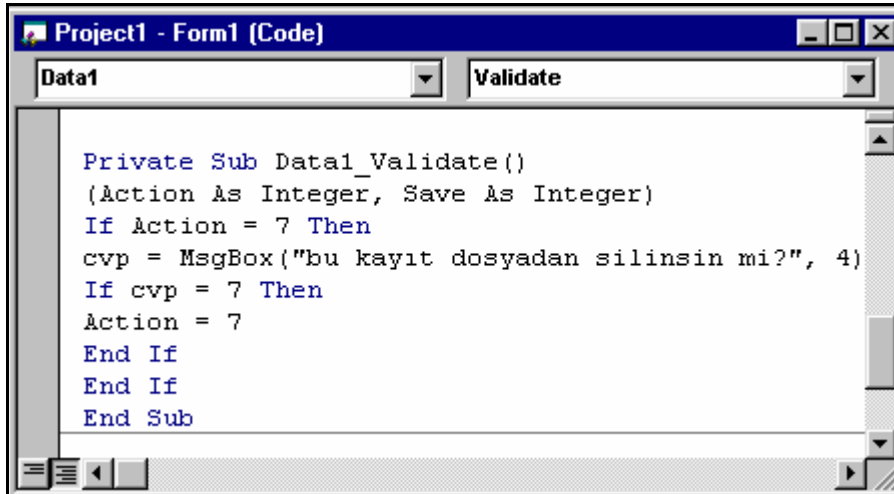
Data nesnesi ile ilgili olarak yapılan MoveNext, MoveLast, AddNew, Delete vb. olaylardan önce **“Validate”** olayı meydana gelir. Data nesnesi üzerinde onay alınması gereken, uyarma gereken tüm işlemlerin Validate olayı içine yazılması gerekir. Validate olayının kendinden iki değişkeni vardır: Action ve Save adlı parametreler. Action adlı parametrede olayın ne olduğu belirlenir(kayıt, silme..). action parametrelerinin kendine ait sayısal değerleri vardır. Her olay için action bir sayısal değer üretmiştir.

Action Değişkeninin İçeriği	Data Nesnesi Üzerinde Yapılmak İstenen İşlem
-----------------------------	--

- | | |
|---|--|
| 1 | MoveFirst ile dosyadaki ilk kayda gitmek |
| 2 | MovePrevious ile bir önceki kayda gitmek |
| 3 | MoveNext ile bir sonraki kayda gitmek |
| 4 | MoveLast ile dosyadaki en son kayda gitmek |
| 5 | AddNew ile dosyaya yeni kayıt yazmak |
| 6 | Update ile dosyaya güncelleme yapmak |
| 7 | Delete ile dosyadan kayıt silmek |
| 8 | Find ile kayıt arama işlemi yapmak |
| 9 | Bookmark ile kayıt işaretleme |

- 10 Close ile dosyayi kapatmak
- 11 UnLoad ile formu yani dosyayi bellekten silmek

Bunlari öğrendikten sonra simdi küçük bir örnek yapalım:

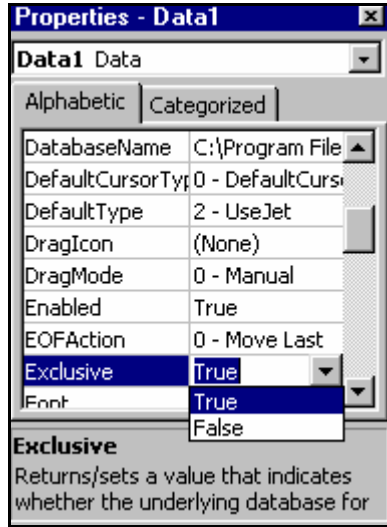


Bu örnek ile kayıt dosyadan silinmek istendiğinde ilk olarak karsimiza Validate yordaminin kontrolü gelir karsimiza ve kaydi silip silmedigimiz sorulur. Böylece yapılacak olan yanlisliklar engellenmis olur. Önümüze çıkacak ekran ise su sekilde olmalıdır:

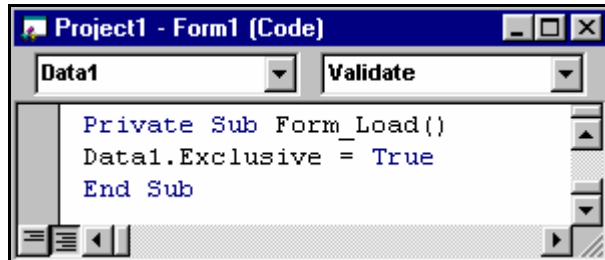
The screenshot shows a Visual Basic 6.0 form titled 'Form1'. It contains five text boxes for user registration: 'Adı' (Name) with 'Atiye', 'Soyadı' (Surname) with 'Yılmaz', 'Telefon' (Phone) with '6341236', 'Adres' (Address) with 'Sancaktepe Mah.', and 'Şehir' (City). A modal dialog box is open over the 'Şehir' field, titled 'adece'. The dialog asks 'bu kayıt dosyadan silinsin mi?' (Should this record be deleted from the file?). It has two buttons: 'Evet' (Yes) and 'Hayır' (No). Below the dialog, there are two buttons: 'Yeni Kayıt' (New Record) and 'Kayıt Silme' (Delete Record).

Eger kullanıcı kaydı silmeyi onaylıyorsa ek işlem yapılamaz ve Validate yordamından çıkılır. Kayıt silme işlemi onaylamadığı anda Validate'e "0" aktarılır ve çıkılır, fakat eğer kayıt silmek isteniyorsa Validate'in içeriği "7" olur ve kayıt silinmiş olur. Action değişkeninin içeriği "0" olduğu zaman o nesne üzerinde hiçbir işlem yapılmaz.

Dosyayı Baska Kullanıcılara Kapatmak Exclusive Özelliği



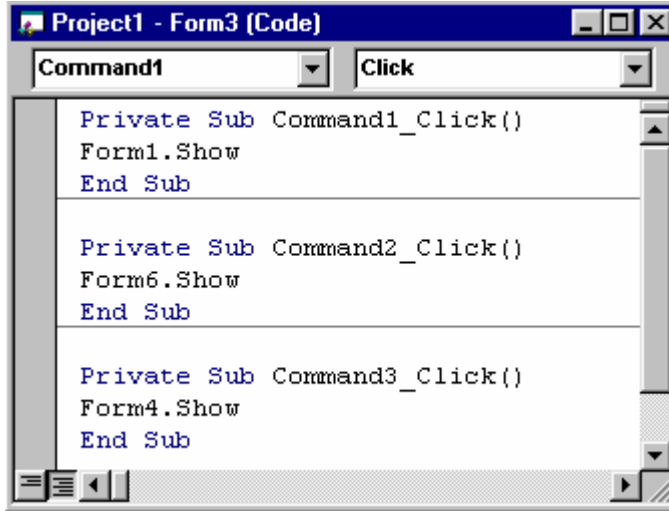
Çok kullanicili ortamlarda üzerinde çalıştığınız projenizin baska bir kullanıcı tarafından açılmamasini, degisiklik yapılmamasini istiyorsanız Table nesnesine ait **“Exclusive”** özelliginden faydalanabilirsiniz. Baslangıçta bu Data nesnesinin Exclusive özelligi False degerini içerir.



İlk olarak Microsoft Access’de degiskenleri (adsoy,no,bolum,odev,vize,quiz) olan bir tablo(ham) hazırlayacağız,sonra da ikinci tablo(tanim)’ya bölümleri aktaracağız...



Giris için bir ana form hazırladık. Buradan kayıt/arama/silme/degistirme/listeleme bölümlerine geçiş yapacağız.



```
Private Sub Command1_Click()  
Form1.Show  
End Sub  
  
Private Sub Command2_Click()  
Form6.Show  
End Sub  
  
Private Sub Command3_Click()  
Form4.Show  
End Sub
```

Data1 "DatabaseName" bölümünden Microsoft Access dosyasına bağlandırılır,daha sonra hangi tablo ile işlem yapacaksa "RecordSource" ile ona bağlandırılır. Bizim projemizde data1 "ham" adli tabloya, data2 ise dosyaya bağlandıktan sonra bölümlerin tutulduğu "tanım" adli tabloya bağlanmıştır. Bu bağlanmaları yaptıktan sonra aşağıdaki gibi kodları projemize yazalım:

KAYIT MODÜLÜ

```

Private Sub Kayit_Click()
Data1.Recordset.AddNew
Data1.Recordset("adsoyad") = Text1.Text
Data1.Recordset("no") = Text2.Text
Data1.Recordset("bolum") = Combol.Text
Data1.Recordset("odev") = Text3.Text
Data1.Recordset("vize") = Text4.Text
Data1.Recordset("quiz") = Text5.Text
Data1.Recordset.Update
Data1.Refresh
Text1.Text = ""
Text2.Text = ""
Combol.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Combol.ListIndex = 0
Text1.SetFocus
End Sub

```

```

Private Sub Temizle_Click()
Text1.Text = ""
Text2.Text = ""
Combol.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Combol.ListIndex = 0

```

Microsoft Visual Basic 6.0

```
Text1.SetFocus
End Sub
Private Sub cikis_Click()
Unload Me
End Sub

Private Sub Form_Activate()
On Error GoTo k
Data2.RecordSource = "TANIM"
Data2.Refresh
Data2.Recordset.MoveLast
a = Data2.Recordset.RecordCount
Data2.Recordset.MoveFirst
For i = 1 To a
Combol.AddItem Data2.Recordset("Bolum1")
Data2.Recordset.MoveNext
Next i
Combol.ListIndex = 0
Text1.SetFocus
Exit Sub
k:
Select Case Err
Case Is = 3021
MsgBox ("Tanimlanmis Bölüm Kaydi Yok")
End Select
Text1.SetFocus
End Sub
```

ARAMA MODÜLÜ

```
Private Sub Form_Activate()  
On Error GoTo k  
Data2.RecordSource = "TANIM"  
Data2.Refresh  
Data2.Recordset.MoveLast  
a = Data2.Recordset.RecordCount  
Data2.Recordset.MoveFirst  
For i = 1 To a  
Comb1.AddItem Data2.Recordset("BOLUM1")  
Data2.Recordset.MoveNext  
Next i  
Comb1.ListIndex = 0  
Text1.SetFocus  
Exit Sub  
k:  
Select Case Err  
Case Is = 3021  
MsgBox ("Tanimlanmis Bölüm Kaydi Yok")  
End Select  
Text1.SetFocus  
End Sub  
  
Private Sub Text1_KeyPress(KeyAscii As Integer)  
On Error GoTo k  
If KeyAscii = 13 Then  
sql1 = "select      HAM.*  from  HAM  where(HAM.ADSOYAD  like'" &  
Text1.Text & "')" &  
Data1.RecordSource = sql1  
Data1.Refresh  
Text1.Text = Data1.Recordset("adsoyad")  
Text2.Text = Data1.Recordset("no")  
Comb1.Text = Data1.Recordset("bolum")  
Text3.Text = Data1.Recordset("odev")  
Text4.Text = Data1.Recordset("vize")  
Text5.Text = Data1.Recordset("quiz")  
End If  
Exit Sub  
k:  
Select Case Err  
Case Is = 3021  
MsgBox ("Bu Kayit Yok")  
End Select  
Text1.SetFocus  
End Sub
```

'Aramayi yapmak için aradiginiz ismi yazip "Enter" 'a basmaniz yeterli olacaktır.

LİSTELEME MODÜLÜ

The screenshot shows a Visual Basic application window titled "Listeleme". The window has a blue title bar with a close button. The main area contains a label "BÖLÜM SEÇİNİZ:" followed by a dropdown menu. Below the dropdown is a table with one row and one column containing an asterisk "*". The table is mostly empty. At the bottom, there are two data grids labeled "Data1" and "Data2", each with navigation buttons (back, forward, first, last). To the right of these is a button labeled "ÇIKIŞ".

```
Option Explicit
Dim a, i As Integer
Dim sql1 As String
Private Sub Combol_Click()
On Error GoTo k
sql1 = "select      HAM.*   from  HAM   where(HAM.BOLUM   like'"  &
Combol.Text & "')"
Data1.RecordSource = sql1
Data1.Refresh
If Combol.Text = "tüm kayıtlar" Then
Data1.RecordSource = "ham"
Data1.Refresh
End If
Exit Sub
k:
Select Case Err
Case Is = 3021
MsgBox ("Bu Kayıt Yok")
End Select
End Sub
```



```
Private Sub Command1_Click()  
Unload Me  
End Sub  
  
Private Sub Form_Activate()  
On Error GoTo k  
Data2.RecordSource = "TANIM"  
Data2.Refresh  
Data2.Recordset.MoveLast  
a = Data2.Recordset.RecordCount  
Data2.Recordset.MoveFirst  
For i = 1 To a  
Comb1.AddItem Data2.Recordset("BOLUM1")  
Data2.Recordset.MoveNext  
Next i  
Comb1.AddItem "tüm kayıtlar"  
Comb1.ListIndex = 0  
Data1.RecordSource = "ham"  
Data1.Refresh  
  
Exit Sub  
k:  
Select Case Err  
Case Is = 3021  
MsgBox ("Tanımlanmış Bölüm Kaydı Yok")  
End Select  
End Sub
```

'Listelemek için yani tüm kayıtları göstereceğimiz nesne için "**DbGrid**" kullandık. İstediginiz bölümü listeleyebilmeniz için combo kutusundan bir bölüm ismi veya tüm kayıtları seçiniz...

Çalıştırıldığında:

Listeleme

BÖLÜM SEÇİNİZ:

	adsoyad	no	bolum	odev	vize	
►	Ahmet Çak	11	işletme	80	70	
	Salih Bayar	12	bilgisayar prog	65	65	
	Talha Atik	13	bilgisayar prog	100	75	
	Faruk Tınık	14	elektrik	50	56	
	Rana Lal	17	elektrik	90	45	
	Arzu Tuncay	19	elektrik	90	70	
	Atiye Akman	23	işletmecilik	100	100	

Bir baska proje:

İlk olarak formumuza bir data bir hareketli gösteri için timer bir listeleme yapmak için bir DBGrid 8 tane command button ve bir label ve seçenekler için bir combo ekleyeceğiz.

Datamızın bağlantısı da şekilde gördüğünün şekilde

olacaktır. Bundan sonra ise oluşturulan tablonun bağlanması

Microsoft Visual Basic 6.0



şeklinde olacaktır. Access veritabanına baktığımızda ise



Form1:

```
Dim ilk As Boolean
Dim veri As Database
Dim silkod As String
Dim ileri As Boolean

Private Sub Combol_Click()
    Select Case Combol.ListIndex
        Case 0
            Data1.RecordSource = "select * from stok order by parcano"
        Case 1
            Data1.RecordSource = "select * from stok order by
parcaadi"
        Case 2
            Data1.RecordSource = "select * from stok order by
miktar"
        Case 3
            Data1.RecordSource = "select * from stok order by
rafnumarasi "
        Case 4
            Data1.RecordSource = "select * from stok order by
kritik "
        Case 5
            Data1.RecordSource = "select * from stok order by
birim "
    End Select
    Data1.Refresh
    DBGrid1.Columns(0).Caption = "Parça Numarasi"
    DBGrid1.Columns(1).Caption = "Parça Adi"
    DBGrid1.Columns(2).Caption = "Miktar"
```

```

        DBGrid1.Columns(3).Caption = "Raf Numarasi"
        DBGrid1.Columns(4).Caption = "Kritik Seviye"
        DBGrid1.Columns(5).Caption = "BirimFiyat"
        DBGrid1.Columns(5).NumberFormat = "#,##0.00"
        DBGrid1.Columns(0).Width = 1500
        DBGrid1.Columns(1).Width = 3350
        DBGrid1.Columns(2).Width = 800
        DBGrid1.Columns(3).Width = 1300
        DBGrid1.Columns(4).Width = 1000
        DBGrid1.Columns(5).Width = 1000
    End Sub

Private Sub YeniKayit_Click()
    Form2.Show 1
End Sub

Private Sub KayitSil_Click()
    If Datal.Recordset.EOF Then
        Exit Sub
    End If
    silkod = Trim(DBGrid1.Columns(0).Value)
    mesaj = MsgBox(silkod + " kodlu ürün" + Chr(13) + "ve ona ait  
tüm hareket bilgileri kayıtlardan silinecektir.", vbOKCancel)
    If mesaj = 1 Then
        Datal.Recordset.Delete
        Datal.Refresh
        DBGrid1.Columns(0).Caption = "Parça Numarasi"
        DBGrid1.Columns(1).Caption = "Parça Adi"
        DBGrid1.Columns(2).Caption = "Miktar"
        DBGrid1.Columns(3).Caption = "Raf Numarasi"
        DBGrid1.Columns(4).Caption = "Kritik Seviye"
        DBGrid1.Columns(5).Caption = "BirimFiyat"
        DBGrid1.Columns(5).NumberFormat = "#,##0.00"
        DBGrid1.Columns(0).Width = 1500
        DBGrid1.Columns(1).Width = 3350
        DBGrid1.Columns(2).Width = 800
        DBGrid1.Columns(3).Width = 1300
        DBGrid1.Columns(4).Width = 1000
        DBGrid1.Columns(5).Width = 1000
        Set veri = OpenDatabase(Form1.Datal.DatabaseName)
        veri.Execute "delete * from hareket where parcano='" +
silkod + "'"
        veri.Close
    End If
End Sub

Private Sub KayitAra_Click()
    Form3.Show 1
End Sub

```

Microsoft Visual Basic 6.0

```
Private Sub Kritik Seviye_Click()  
    Data1.RecordSource = "select * from stok where miktar<=kritik  
order by parcano"  
    Data1.Refresh  
    DBGrid1.Columns(0).Caption = "Parça Numarasi"  
    DBGrid1.Columns(1).Caption = "Parça Adi"  
    DBGrid1.Columns(2).Caption = "Miktar"  
    DBGrid1.Columns(3).Caption = "Raf Numarasi"  
    DBGrid1.Columns(4).Caption = "Kritik Seviye"  
    DBGrid1.Columns(5).Caption = "BirimFiyat"  
    DBGrid1.Columns(5).NumberFormat = "#,##0.00"  
    DBGrid1.Columns(0).Width = 1500  
    DBGrid1.Columns(1).Width = 3350  
    DBGrid1.Columns(2).Width = 800  
    DBGrid1.Columns(3).Width = 1300  
    DBGrid1.Columns(4).Width = 1000  
    DBGrid1.Columns(5).Width = 1000  
End Sub  
  
Private Sub Yazdir_Click()  
    mesaj = MsgBox("Tüm liste yaziciya dökülecek", vbOKCancel)  
    If mesaj = 2 Then Exit Sub  
    yuksek = Printer.Height  
    genis = Printer.Width  
    satir = 3  
    Printer.CurrentX = 0  
    Printer.CurrentY = 500  
    Printer.FontSize = 20  
    Printer.Print "Stok Listesi"  
    Printer.FontSize = 10  
    Printer.CurrentX = 0  
    Printer.CurrentY = 1000  
    Printer.Print "Parça Numarasi"  
    Printer.CurrentX = 1800  
    Printer.CurrentY = 1000  
    Printer.Print "Parça Adi"  
    Printer.CurrentX = 6000  
    Printer.CurrentY = 1000  
    Printer.Print "Miktar"  
    Printer.CurrentX = 7000  
    Printer.CurrentY = 1000  
    Printer.Print "Raf Numarasi"  
    Printer.CurrentX = 8500  
    Printer.CurrentY = 1000  
    Printer.Print "Kritik Seviye"  
    Printer.CurrentX = 10000  
    Printer.CurrentY = 1000  
    Printer.Print "Birim Fiyat"
```

```

Printer.DrawWidth = 10
Printer.Line (0, 1250)-(11500, 1250)
Form1.Data1.Recordset.MoveFirst
Do While Not Form1.Data1.Recordset.EOF
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 0
    Printer.Print Form1.Data1.Recordset.Fields(0).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 1800
    Printer.Print Form1.Data1.Recordset.Fields(1).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 6000
    Printer.Print Form1.Data1.Recordset.Fields(2).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 7000
    Printer.Print Form1.Data1.Recordset.Fields(3).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 8500
    Printer.Print Form1.Data1.Recordset.Fields(4).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 10000
    Printer.Print Form1.Data1.Recordset.Fields(5).Value
    Form1.Data1.Recordset.MoveNext
    satir = satir + 1
If satir = 45 Then
    Printer.NewPage
    satir = 3
    Printer.CurrentX = 0
    Printer.CurrentY = 500
    Printer.FontSize = 20
    Printer.Print "Stok Listesi"
    Printer.FontSize = 10
    Printer.CurrentX = 0
    Printer.CurrentY = 1000
    Printer.Print "Parça Numarasi"
    Printer.CurrentX = 2000
    Printer.CurrentY = 1000
    Printer.Print "Parça Adi"
    Printer.CurrentX = 8000
    Printer.CurrentY = 1000
    Printer.Print "Miktar"
    Printer.CurrentX = 9500
    Printer.CurrentY = 1000
    Printer.Print "Raf Numarasi"
    Printer.CurrentX = 10800
    Printer.CurrentY = 1000
    Printer.Print "Kritik Seviye"
    Printer.CurrentX = 11800
    Printer.CurrentY = 1000

```

Microsoft Visual Basic 6.0

```
        Printer.Print "Birim Fiyat"
        Printer.DrawWidth = 10
        Printer.Line (0, 1100)-(11500, 1100)
    End If
Loop
Printer.EndDoc
End Sub

Private Sub Cikis_Click()
    Unload Me
End Sub

Private Sub TümKayitlar_Click()
    Data1.RecordSource = "select * from stok order by parcano"
    Data1.Refresh
    DBGrid1.Columns(0).Caption = "Parça Numarasi"
    DBGrid1.Columns(1).Caption = "Parça Adi"
    DBGrid1.Columns(2).Caption = "Miktar"
    DBGrid1.Columns(3).Caption = "Raf Numarasi"
    DBGrid1.Columns(4).Caption = "Kritik Seviye"
    DBGrid1.Columns(5).Caption = "BirimFiyat"
    DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    DBGrid1.Columns(0).Width = 1500
    DBGrid1.Columns(1).Width = 3350
    DBGrid1.Columns(2).Width = 800
    DBGrid1.Columns(3).Width = 1300
    DBGrid1.Columns(4).Width = 1000
    DBGrid1.Columns(5).Width = 1000
End Sub

Private Sub StokHareketi_Click()
    Form4.Show 1
End Sub

Private Sub Form_Activate()
    If ilk Then
        Data1.RecordSource = "select * from stok order by parcano"
        Data1.Refresh
        DBGrid1.Columns(0).Caption = "Parça Numarasi"
        DBGrid1.Columns(1).Caption = "Parça Adi"
        DBGrid1.Columns(2).Caption = "Miktar"
        DBGrid1.Columns(3).Caption = "Raf Numarasi"
        DBGrid1.Columns(4).Caption = "Kritik Seviye"
        DBGrid1.Columns(5).Caption = "BirimFiyat"
        DBGrid1.Columns(5).NumberFormat = "#,##0.00"
        DBGrid1.Columns(0).Width = 1500
        DBGrid1.Columns(1).Width = 3350
        DBGrid1.Columns(2).Width = 800
        DBGrid1.Columns(3).Width = 1300
```



```
        DBGrid1.Columns(4).Width = 1000
        DBGrid1.Columns(5).Width = 1000
        ilk = False
    End If
End Sub

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    Form6.Show 1
End Sub

Private Sub Form_Load()
    ilk = True
    ileri = True
    Data1.DatabaseName = App.Path + "\" + Data1.DatabaseName
    Data1.RecordSource = "select * from stok order by parcano"
    Data1.Refresh
End Sub

Private Sub Timer1_Timer()
    If ileri Then
        Line1.X1 = Line1.X1 + 250
        Line2.X1 = Line2.X1 + 250
        Line1.X2 = Line1.X2 + 250
        Line2.X2 = Line2.X2 + 250
        If Line1.X1 = 9870 Then ileri = False
    Else
        Line1.X1 = Line1.X1 - 250
        Line2.X1 = Line2.X1 - 250
        Line1.X2 = Line1.X2 - 250
        Line2.X2 = Line2.X2 - 250
        If Line1.X1 = 120 Then ileri = True
    End If
End Sub
```

Form2:

Kayıt Ekle

Parça Numarası

Parça Adı

Miktar

Raf Numarası

Kritik Seviye

Birim Fiyat

Kaydet

Vazgeç

```
Private Sub Vazgec_Click()  
    Unload Me  
End Sub  
  
Private Sub Kaydet_Click()  
    If Len(Text1.Text) = 0 Then  
        mesaj = MsgBox("Lütfen Parça Numarasini Giriniz...")  
        Text1.SetFocus  
        Exit Sub  
    End If  
    If Len(Text4.Text) = 0 Then  
        mesaj = MsgBox("Lütfen Raf Numarasini Giriniz...")  
        Text4.SetFocus  
        Exit Sub  
    End If  
    If Len(Text2.Text) = 0 Then  
        Text2.Text = " "  
    End If  
    If Len(Text3.Text) = 0 Then  
        Text3.Text = "0"  
    End If  
    Text3.Text = Val(Text3.Text)  
    If Len(Text5.Text) = 0 Then  
        Text5.Text = "0"  
    End If  
    Text5.Text = Val(Text5.Text)  
    veritab = Form1.Data1.RecordSource  
    Form1.Data1.RecordSource = "select * from stok where  
    parcano='" + Text1.Text + "'"
```

```

Form1.Data1.Refresh
If Not Form1.Data1.Recordset.RecordCount = 0 Then
    Form1.Data1.Recordset.Edit
    Form1.Data1.Recordset.Fields(1) = Text2.Text
    Form1.Data1.Recordset.Fields(2) = Text3.Text
    Form1.Data1.Recordset.Fields(3) = Text4.Text
    Form1.Data1.Recordset.Fields(4) = Text5.Text
    Form1.Data1.Recordset.Fields(5) = Text6.Text
    Form1.Data1.Recordset.Update
    Text1.Text = ""
    Text2.Text = ""
    Text3.Text = ""
    Text4.Text = ""
    Text5.Text = ""
    Text6.Text = ""
    Form1.Data1.RecordSource = veritab
    Form1.Data1.Refresh
    Form1.DBGrid1.Columns(0).Caption = "Parça Numarasi"
    Form1.DBGrid1.Columns(1).Caption = "Parça Adi"
    Form1.DBGrid1.Columns(2).Caption = "Miktar"
    Form1.DBGrid1.Columns(3).Caption = "Raf Numarasi"
    Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
    Form1.DBGrid1.Columns(5).Caption = "Birim Fiyat"
    Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form1.DBGrid1.Columns(0).Width = 1500
    Form1.DBGrid1.Columns(1).Width = 3350
    Form1.DBGrid1.Columns(2).Width = 800
    Form1.DBGrid1.Columns(3).Width = 1300
    Form1.DBGrid1.Columns(4).Width = 1000
    Form1.DBGrid1.Columns(5).Width = 1000
    Text1.SetFocus
Exit Sub
End If
Form1.Data1.Recordset.AddNew
Form1.Data1.Recordset.Fields(0) = Text1.Text
Form1.Data1.Recordset.Fields(1) = Text2.Text
Form1.Data1.Recordset.Fields(2) = Text3.Text
Form1.Data1.Recordset.Fields(3) = Text4.Text
Form1.Data1.Recordset.Fields(4) = Text5.Text
Form1.Data1.Recordset.Fields(5) = Text6.Text
Form1.Data1.Recordset.Update
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
Form1.Data1.RecordSource = veritab
Form1.Data1.Refresh

```

Microsoft Visual Basic 6.0

```
Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
Form1.DBGrid1.Columns(2).Caption = "Miktar"
Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
Form1.DBGrid1.Columns(5).Caption = "Birim Fiyat"
Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
Form1.DBGrid1.Columns(0).Width = 1500
Form1.DBGrid1.Columns(1).Width = 3350
Form1.DBGrid1.Columns(2).Width = 800
Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
Text1.SetFocus
End Sub

Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then SendKeys "{TAB}"
End Sub

Sub Form_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then KeyAscii = 0
End Sub

Private Sub Text1_Change()
    veritab = Form1.Data1.RecordSource
    Form1.Data1.RecordSource = "select * from stok where
parcano='" + Text1.Text + "'"
    Form1.Data1.Refresh
    If Not Form1.Data1.Recordset.RecordCount = 0 Then
        Text2.Text = Form1.Data1.Recordset.Fields(1).Value
        Text3.Text = Form1.Data1.Recordset.Fields(2).Value
        Text4.Text = Form1.Data1.Recordset.Fields(3).Value
        Text5.Text = Form1.Data1.Recordset.Fields(4).Value
        Text6.Text = Form1.Data1.Recordset.Fields(5).Value
        Text3.SetFocus
    End If
    Form1.Data1.RecordSource = veritab
    Form1.Data1.Refresh
    Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
    Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
    Form1.DBGrid1.Columns(2).Caption = "Miktar"
    Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
    Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
    Form1.DBGrid1.Columns(5).Caption = "Birim Fiyat"
    Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form1.DBGrid1.Columns(0).Width = 1500
    Form1.DBGrid1.Columns(1).Width = 3350
    Form1.DBGrid1.Columns(2).Width = 800
```

```

Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
End Sub

Private Sub Text1_LostFocus()
deger = ""
For i = 1 To Len(Text1.Text)
    harf = UCase(Mid(Text1.Text, i, 1))
    If harf = "*" Then
        harf = "-"
    End If
    deger = deger & harf
Next i
Text1.Text = deger
End Sub

```

Form3:

```

Private Sub Command1_Click()
    If Len(Text1.Text) = 0 Then
        mesaj = MsgBox("Lütfen Parça Numarasi verin")
        Text1.SetFocus
        Exit Sub
    End If
    deger = ""
    For i = 1 To Len(Text1.Text)
        harf = UCase(Mid(Text1.Text, i, 1))
        If harf = "*" Then
            harf = "-"
        End If
    Next i
    Text1.Text = deger
End Sub

```

Microsoft Visual Basic 6.0

```
        deger = deger & harf
Next i
Text1.Text = deger
Form1.Data1.Recordset.FindFirst "parcano = '" + Text1.Text +
'''
If Form1.Data1.Recordset.NoMatch Then
    mesaj = MsgBox("Aradiginiz kayit bulunamamistir.")
    Text1.SetFocus
    Exit Sub
End If
Form1.DBGrid1.Refresh
Unload Me
End Sub

Private Sub Command2_Click()
    If Len(Text2.Text) = 0 Then
        mesaj = MsgBox("Lütfen Parça Adi verin")
        Text2.SetFocus
        Exit Sub
    End If
    Form1.Data1.Recordset.FindFirst "parcaadi = '" + Text2.Text +
'''
    If Form1.Data1.Recordset.NoMatch Then
        mesaj = MsgBox("Aradiginiz kayit bulunamamistir.")
        Text2.SetFocus
        Exit Sub
    End If
    Form1.DBGrid1.Refresh
    Unload Me
End Sub

Private Sub Command3_Click()
    Unload Me
End Sub

Private Sub Command4_Click()
    If Len(Text3.Text) = 0 Then
        mesaj = MsgBox("Lütfen Raf Numarasi verin")
        Text3.SetFocus
        Exit Sub
    End If
    Form1.Data1.Recordset.FindFirst "rafnumarasi = '" + Text3.Text
+ '''
    If Form1.Data1.Recordset.NoMatch Then
        mesaj = MsgBox("Aradiginiz kayit bulunamamistir.")
        Text3.SetFocus
        Exit Sub
    End If
    Form1.DBGrid1.Refresh
```



```
Dim ilk As Boolean
Public Sub gduz()
    DBGrid1.Columns(0).Caption = "Parça Numarasi"
    DBGrid1.Columns(1).Caption = "Hareket Tipi"
    DBGrid1.Columns(2).Caption = "Tarih"
    DBGrid1.Columns(3).Caption = "Miktar"
    DBGrid1.Columns(4).Caption = "Çagri Numarasi"
    DBGrid1.Columns(5).Caption = "Tutar"
    DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    DBGrid1.Columns(0).Width = 1200
    DBGrid1.Columns(1).Width = 1000
    DBGrid1.Columns(2).Width = 1000
    DBGrid1.Columns(3).Width = 1000
    DBGrid1.Columns(4).Width = 1200
    DBGrid1.Columns(5).Width = 1200
End Sub
Private Sub Cikis_Click()
```



```
Unload Me
End Sub
```

```
Private Sub HepsiniGöster_Click()
    Datal.RecordSource = "select * from hareket order by  
parcano,hareket,tarih"  
    Datal.Refresh  
    gduz  
End Sub
```

```
Private Sub KayitAra_Click()  
    Form7.Show 1  
End Sub
```

```
Private Sub KayitSil_Click()  
    If Datal.Recordset.EOF Then  
        Exit Sub  
    End If  
    silkod = Trim(DBGrid1.Columns(0).Value)  
    mesaj = MsgBox(silkod + " kodlu ürüne ait bu stok hareketi  
kayitlardan silinecektir.", vbOKCancel)  
    If mesaj = 1 Then  
        Datal.Recordset.Delete  
        Datal.Refresh  
        gduz  
    End If  
End Sub
```

```
Private Sub Yazdir_Click()  
    mesaj = MsgBox("Tüm liste yaziciya dökülecek", vbOKCancel)  
    If mesaj = 2 Then Exit Sub  
    yuksek = Printer.Height  
    genis = Printer.Width  
    satir = 3  
    Printer.CurrentX = 0  
    Printer.CurrentY = 500  
    Printer.FontSize = 20  
    Printer.Print "Stok Hareket Listesi"  
    Printer.FontSize = 10  
    Printer.CurrentX = 0  
    Printer.CurrentY = 1000  
    Printer.Print "Parça Numarasi"  
    Printer.CurrentX = 1800  
    Printer.CurrentY = 1000  
    Printer.Print "Islem"
```

```
Printer.CurrentX = 3000
Printer.CurrentY = 1000
Printer.Print "Tarih"
Printer.CurrentX = 4200
Printer.CurrentY = 1000
Printer.Print "Miktar"
Printer.CurrentX = 5800
Printer.CurrentY = 1000
Printer.Print "Tutar"
Printer.CurrentX = 7100
Printer.CurrentY = 1000
Printer.Print "Çagrı Numarasi"
Printer.DrawWidth = 10
Printer.Line (0, 1250)-(11500, 1250)
Form4.Data1.Recordset.MoveFirst
Do While Not Form4.Data1.Recordset.EOF
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 0
    Printer.Print Form4.Data1.Recordset.Fields(0).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 1800
    Printer.Print Form4.Data1.Recordset.Fields(1).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 3000
    Printer.Print Form4.Data1.Recordset.Fields(2).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 4200
    Printer.Print Form4.Data1.Recordset.Fields(3).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 5800
    Printer.Print Form4.Data1.Recordset.Fields(5).Value
    Printer.CurrentY = 500 + satir * 300
    Printer.CurrentX = 7100
    Printer.Print Form4.Data1.Recordset.Fields(4).Value
    Form4.Data1.Recordset.MoveNext
    satir = satir + 1
If satir = 45 Then
    Printer.NewPage
    satir = 3
    Printer.CurrentX = 0
    Printer.CurrentY = 500
    Printer.FontSize = 20
    Printer.Print "Stok Hareket Listesi"
    Printer.FontSize = 10
    Printer.CurrentX = 0
    Printer.CurrentY = 1000
    Printer.Print "Parça Numarasi"
    Printer.CurrentX = 1800
    Printer.CurrentY = 1000
```

```

Printer.Print "Islem"
Printer.CurrentX = 3000
Printer.CurrentY = 1000
Printer.Print "Tarih"
Printer.CurrentX = 4200
Printer.CurrentY = 1000
Printer.Print "Miktar"
Printer.CurrentX = 5800
Printer.CurrentY = 1000
Printer.Print "Tutar"
Printer.CurrentX = 7100
Printer.CurrentY = 1000
Printer.Print "Çagri Numarasi"
Printer.DrawWidth = 10
Printer.Line (0, 1100)-(11500, 1100)
End If
Loop
Printer.EndDoc
End Sub

Private Sub DBGrid1_DblClick()
Form5.Show 1
End Sub

Private Sub DBGrid1_HeadClick(ByVal ColIndex As Integer)
Select Case ColIndex
Case 0
deger = InputBox("Parça Numarasi")
If Len(deger) = 0 Then Exit Sub
Datal.RecordSource = "select * from hareket where
parcano='" + deger + "'" order by hareket,tarih"
Datal.Refresh
gduz
Case 1
deger = InputBox("Hareket Tipi")
If Len(deger) = 0 Then Exit Sub
Datal.RecordSource = "select * from hareket where
hareket='" + deger + "'" order by parcano,tarih"
Datal.Refresh
gduz
Case 2
deger = InputBox("Islem Tarihi")
If Len(deger) = 0 Then Exit Sub
Datal.RecordSource = "select * from hareket where
tarih=#" + deger + "#" order by parcano,hareket"
Datal.Refresh
gduz
Case 3
deger = InputBox("Miktar")

```

Microsoft Visual Basic 6.0

```
        If Len(deger) = 0 Then Exit Sub
        Data1.RecordSource = "select * from hareket where
miktar=" + deger + " order by parcano,hareket,tarih"
        Data1.Refresh
        gduz
    Case 4
        deger = InputBox("Çagri Numarasi")
        If Len(deger) = 0 Then Exit Sub
        Data1.RecordSource = "select * from hareket where
cagri=" + deger + " order by parcano,hareket,tarih"
        Data1.Refresh
        gduz
    Case 5
        deger = InputBox("Tutar")
        If Len(deger) = 0 Then Exit Sub
        Data1.RecordSource = "select * from hareket where
tutar=" + deger + " order by parcano,hareket,tarih"
        Data1.Refresh
        gduz
    End Select
End Sub

Private Sub Form_Load()
    ilk = True
    Data1.DatabaseName = Form1.Data1.DatabaseName
End Sub
Private Sub Form_Activate()
    If ilk Then
        Data1.RecordSource = "select * from hareket order by
parcano,hareket,tarih"
        Data1.Refresh
        gduz
        ilk = False
    End If
End Sub
```

Form5:

```

Private Sub Command1_Click()
    If Len(Text3.Text) = 0 Or Val(Text3.Text) = 0 Then
        mesaj = MsgBox("Lütfen bir stok giriş/çıkış miktarını giriniz..")
        Text3.SetFocus
        Exit Sub
    End If
    If (Len(Text4.Text) = 0 Or Val(Text4.Text) = 0) And
        Combol1.ListIndex = 1 Then
        mesaj = MsgBox("Lütfen bir çağrı numarası giriniz..")
        Text4.SetFocus
        Exit Sub
    End If
    veritab = Form1.Datal.RecordSource
    Form1.Datal.RecordSource = "select * from stok where
    parcano='" + Text1.Text + "'"
    Form1.Datal.Refresh
    Form1.Datal.Recordset.Edit
    If Combol1.ListIndex = 0 Then
        Form1.Datal.Recordset.Fields(2) =
        Form1.Datal.Recordset.Fields(2) + Val(Text3.Text)
    Else
        Form1.Datal.Recordset.Fields(2) =
        Form1.Datal.Recordset.Fields(2) - Val(Text3.Text)
    End If
    Form1.Datal.Recordset.Update
    Form1.Datal.RecordSource = veritab
    Form1.Datal.Refresh

```

Microsoft Visual Basic 6.0

```
Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
Form1.DBGrid1.Columns(2).Caption = "Miktar"
Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
Form1.DBGrid1.Columns(0).Width = 1500
Form1.DBGrid1.Columns(1).Width = 3350
Form1.DBGrid1.Columns(2).Width = 800
Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
Form4.Data1.Recordset.AddNew
Form4.Data1.Recordset.Fields(0) = Text1.Text
If Combol.ListIndex = 0 Then
    Form4.Data1.Recordset.Fields(1) = "GIRIS"
    Form4.Data1.Recordset.Fields(4) = "0"
Else
    Form4.Data1.Recordset.Fields(1) = "ÇIKIS"
    Form4.Data1.Recordset.Fields(4) = Text4.Text
End If
Form4.Data1.Recordset.Fields(2) = DTPicker1.Value
Form4.Data1.Recordset.Fields(3) = Text3.Text
Form4.Data1.Recordset.Fields(5) = Text5.Text
Form4.Data1.Recordset.Update
Text1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Unload Me
End Sub

Private Sub Command2_Click()
    Unload Me
End Sub

Private Sub Form_Activate()
    Combol.ListIndex = 0
    veritab = Form1.Data1.RecordSource
    Text1.Text = Form4.Data1.Recordset.Fields(0)
    Form1.Data1.RecordSource = "select * from stok where parcano='" + Text1.Text + "'"
    Form1.Data1.Refresh
    If Form1.Data1.Recordset.RecordCount = 0 Then
        mesaj = MsgBox("Ana dosyada böyle bir parça numarası bulunmamaktadır." + Chr(13) + "Lütfen önce parça girişini yapınız.")
        Form1.Data1.RecordSource = veritab
    End If
End Sub
```

```

Form1.Datal.Refresh
Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
Form1.DBGrid1.Columns(2).Caption = "Miktar"
Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
Form1.DBGrid1.Columns(0).Width = 1500
Form1.DBGrid1.Columns(1).Width = 3350
Form1.DBGrid1.Columns(2).Width = 800
Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
Unload Me
Else
    Label7.Caption = Form1.Datal.Recordset.Fields(1)
    Label9.Caption = Form1.Datal.Recordset.Fields(2)
    Label11.Caption = Form1.Datal.Recordset.Fields(5)
    Text3.SetFocus
    Form1.Datal.RecordSource = veritab
    Form1.Datal.Refresh
    Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
    Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
    Form1.DBGrid1.Columns(2).Caption = "Miktar"
    Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
    Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
    Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
    Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form1.DBGrid1.Columns(0).Width = 1500
    Form1.DBGrid1.Columns(1).Width = 3350
    Form1.DBGrid1.Columns(2).Width = 800
    Form1.DBGrid1.Columns(3).Width = 1300
    Form1.DBGrid1.Columns(4).Width = 1000
    Form1.DBGrid1.Columns(5).Width = 1000
    Text3.SetFocus
End If
End Sub

Private Sub Text3_Change()
    Text5.Text = Val(Label11.Caption) * Val(Text3.Text)
End Sub

```

Form6:

The screenshot shows a Visual Basic form titled "Hareket Ekranı". The form has a blue title bar with a yellow icon. It contains several input fields: "Parça Numarası" (text box), "Hareket Tipi" (dropdown menu with "Giriş" selected), "Tarih" (dropdown menu with "07.10.2001" selected), "İşlem Miktarı" (text box), "Stok Miktarı" (text box), "Çağrı Numarası" (text box), "Tutar" (text box), and "Birim Fiyat" (text box). At the bottom, there are two buttons: "Kaydet" and "Vazgeç".

```
Dim veri As Database
Dim tablo As Recordset
Private Sub Command1_Click()
    If Len(Text3.Text) = 0 Or Val(Text3.Text) = 0 Then
        mesaj = MsgBox("Lütfen bir stok giris/çikis miktarini giriniz..")
        Text3.SetFocus
        Exit Sub
    End If
    If (Len(Text4.Text) = 0 Or Val(Text4.Text) = 0) And Combol.ListIndex = 1 Then
        mesaj = MsgBox("Lütfen bir çağrı numarasi giriniz..")
        Text4.SetFocus
        Exit Sub
    End If
    veritab = Form1.Data1.RecordSource
    Form1.Data1.RecordSource = "select * from stok where parcano='" + Text1.Text + "'"
    Form1.Data1.Refresh
    Form1.Data1.Recordset.Edit
    If Combol.ListIndex = 0 Then
        Form1.Data1.Recordset.Fields(2) =
Form1.Data1.Recordset.Fields(2) + Val(Text3.Text)
    Else
        Form1.Data1.Recordset.Fields(2) =
Form1.Data1.Recordset.Fields(2) - Val(Text3.Text)
    End If
    Form1.Data1.Recordset.Update
    Form1.Data1.RecordSource = veritab
End Sub
```



```

Form1.Data1.Refresh
Form1.DBGrid1.Columns(0).Caption = "Parça Numarasi"
Form1.DBGrid1.Columns(1).Caption = "Parça Adi"
Form1.DBGrid1.Columns(2).Caption = "Miktar"
Form1.DBGrid1.Columns(3).Caption = "Raf Numarasi"
Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
Form1.DBGrid1.Columns(0).Width = 1500
Form1.DBGrid1.Columns(1).Width = 3350
Form1.DBGrid1.Columns(2).Width = 800
Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
Set veri = OpenDatabase(Form1.Data1.DatabaseName)
Set tablo = veri.OpenRecordset("hareket")
tablo.AddNew
tablo.Fields(0) = Text1.Text
If Combol.ListIndex = 0 Then
    tablo.Fields(1) = "GIRIS"
    tablo.Fields(4) = "0"
Else
    tablo.Fields(1) = "ÇIKIS"
    tablo.Fields(4) = Text4.Text
End If
tablo.Fields(2) = DTPicker1.Value
tablo.Fields(3) = Text3.Text
'Text5.Text = Text3.Text * Val(Label9.Caption)
tablo.Fields(5) = Text5.Text
tablo.Update
Text1.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
veri.Close
Text1.SetFocus
End Sub

Private Sub Command2_Click()
    Unload Me
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
    Combol.ListIndex = 0
    If KeyAscii = 13 Then
        deger = ""
        For i = 1 To Len(Text1.Text)
            harf = UCase(Mid(Text1.Text, i, 1))
            If harf = "*" Then

```

Microsoft Visual Basic 6.0

```
        harf = "-"
    End If
    deger = deger & harf
Next i
Text1.Text = deger
veritab = Form1.Data1.RecordSource
Form1.Data1.RecordSource = "select * from stok where
parcano='" + Text1.Text + "'"
Form1.Data1.Refresh
If Form1.Data1.Recordset.RecordCount = 0 Then
    mesaj = MsgBox("Ana dosyada böyle bir parça numarası
bulunmamaktadır." + Chr(13) + "Lütfen önce parça girişini
yapınız.")
    Form1.Data1.RecordSource = veritab
    Form1.Data1.Refresh
    Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
    Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
    Form1.DBGrid1.Columns(2).Caption = "Miktar"
    Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
    Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
    Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
    Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form1.DBGrid1.Columns(0).Width = 1500
    Form1.DBGrid1.Columns(1).Width = 3350
    Form1.DBGrid1.Columns(2).Width = 800
    Form1.DBGrid1.Columns(3).Width = 1300
    Form1.DBGrid1.Columns(4).Width = 1000
    Form1.DBGrid1.Columns(5).Width = 1000
    Unload Me
Else
    Label7.Caption = Form1.Data1.Recordset.Fields(1)
    Label9.Caption = Form1.Data1.Recordset.Fields(2)
    Label11.Caption = Form1.Data1.Recordset.Fields(5)
    Text3.SetFocus
    Form1.Data1.RecordSource = veritab
    Form1.Data1.Refresh
    Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
    Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
    Form1.DBGrid1.Columns(2).Caption = "Miktar"
    Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
    Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
    Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
    Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form1.DBGrid1.Columns(0).Width = 1500
    Form1.DBGrid1.Columns(1).Width = 3350
    Form1.DBGrid1.Columns(2).Width = 800
    Form1.DBGrid1.Columns(3).Width = 1300
    Form1.DBGrid1.Columns(4).Width = 1000
    Form1.DBGrid1.Columns(5).Width = 1000
```

```

        End If
    End If
End Sub

Private Sub Text1_LostFocus()
    If Len(Text1.Text) = 0 Then
        Exit Sub
    End If
    Combol.ListIndex = 0
    deger = ""
    For i = 1 To Len(Text1.Text)
        harf = UCase(Mid(Text1.Text, i, 1))
        If harf = "*" Then
            harf = "-"
        End If
        deger = deger & harf
    Next i
    Text1.Text = deger
    veritab = Form1.Datal.RecordSource
    Form1.Datal.RecordSource = "select * from stok where
parcano='" + Text1.Text + "'"
    Form1.Datal.Refresh
    If Form1.Datal.Recordset.RecordCount = 0 Then
        mesaj = MsgBox("Ana dosyada böyle bir parça numarası
bulunmamaktadır." + Chr(13) + "Lütfen önce parça girisini
yapınız.")
        Form1.Datal.RecordSource = veritab
        Form1.Datal.Refresh
        Form1.DBGrid1.Columns(0).Caption = "Parça Numarası"
        Form1.DBGrid1.Columns(1).Caption = "Parça Adı"
        Form1.DBGrid1.Columns(2).Caption = "Miktar"
        Form1.DBGrid1.Columns(3).Caption = "Raf Numarası"
        Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
        Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
        Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
        Form1.DBGrid1.Columns(0).Width = 1500
        Form1.DBGrid1.Columns(1).Width = 3350
        Form1.DBGrid1.Columns(2).Width = 800
        Form1.DBGrid1.Columns(3).Width = 1300
        Form1.DBGrid1.Columns(4).Width = 1000
        Form1.DBGrid1.Columns(5).Width = 1000
        Unload Me
    Else
        Label7.Caption = Form1.Datal.Recordset.Fields(1)
        Label9.Caption = Form1.Datal.Recordset.Fields(2)
        Label11.Caption = Form1.Datal.Recordset.Fields(5)
        Text3.SetFocus
        Form1.Datal.RecordSource = veritab
        Form1.Datal.Refresh
    End If
End Sub

```

```
Form1.DBGrid1.Columns(0).Caption = "Parça Numarasi"
Form1.DBGrid1.Columns(1).Caption = "Parça Adi"
Form1.DBGrid1.Columns(2).Caption = "Miktar"
Form1.DBGrid1.Columns(3).Caption = "Raf Numarasi"
Form1.DBGrid1.Columns(4).Caption = "Kritik Seviye"
Form1.DBGrid1.Columns(5).Caption = "BirimFiyat"
Form1.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
Form1.DBGrid1.Columns(0).Width = 1500
Form1.DBGrid1.Columns(1).Width = 3350
Form1.DBGrid1.Columns(2).Width = 800
Form1.DBGrid1.Columns(3).Width = 1300
Form1.DBGrid1.Columns(4).Width = 1000
Form1.DBGrid1.Columns(5).Width = 1000
End If
End Sub

Private Sub Text3_Change()
    Text5.Text = Val(Label11.Caption) * Val(Text3.Text)
End Sub
```

Form7:

```
Public Sub gduz()
    Form4.DBGrid1.Columns(0).Caption = "Parça Numarasi"
    Form4.DBGrid1.Columns(1).Caption = "Hareket Tipi"
    Form4.DBGrid1.Columns(2).Caption = "Tarih"
    Form4.DBGrid1.Columns(3).Caption = "Miktar"
    Form4.DBGrid1.Columns(4).Caption = "Çagri Numarasi"
    Form4.DBGrid1.Columns(5).Caption = "Tutar"
    Form4.DBGrid1.Columns(5).NumberFormat = "#,##0.00"
    Form4.DBGrid1.Columns(0).Width = 1200
    Form4.DBGrid1.Columns(1).Width = 1000
    Form4.DBGrid1.Columns(2).Width = 1000
    Form4.DBGrid1.Columns(3).Width = 1000
    Form4.DBGrid1.Columns(4).Width = 1200
    Form4.DBGrid1.Columns(5).Width = 1200
End Sub
```

```
Private Sub Command1_Click()
    If Len(Text1.Text) = 0 Then Exit Sub
```

Microsoft Visual Basic 6.0

```
Form4.Data1.RecordSource = "select * from hareket where  
parcano='" + Text1.Text + "' order by hareket,tarih"  
Form4.Data1.Refresh  
gduz  
Unload Me  
End Sub  
  
Private Sub Command2_Click()  
If Option1.Value Then  
Form4.Data1.RecordSource = "select * from hareket where  
hareket='GIRIS' order by parcano,hareket,tarih"  
Else  
Form4.Data1.RecordSource = "select * from hareket where  
hareket='ÇIKIS' order by parcano,hareket,tarih"  
End If  
Form4.Data1.Refresh  
gduz  
Unload Me  
End Sub  
  
Private Sub Command4_Click()  
If Len(Text2.Text) = 0 And Len(Text3.Text) = 0 Then Exit Sub  
If Len(Text2.Text) = 0 Then  
deg3 = Val(Text3.Text)  
Form4.Data1.RecordSource = "select * from hareket where  
miktar=" + Str(deg3) + " order by parcano,hareket,tarih"  
End If  
If Len(Text3.Text) = 0 Then  
deg2 = Val(Text2.Text)  
Form4.Data1.RecordSource = "select * from hareket where  
miktar=" + Str(deg2) + " order by parcano,hareket,tarih"  
End If  
If Len(Text2.Text) > 0 And Len(Text3.Text) > 0 Then  
deg2 = Val(Text2.Text)  
deg3 = Val(Text3.Text)  
Form4.Data1.RecordSource = "select * from hareket where  
miktar>=" + Str(deg2) + " and tutar<=" + Str(deg3) + " order by  
parcano,hareket,tarih"  
End If  
Form4.Data1.Refresh  
gduz  
Unload Me  
End Sub  
  
Private Sub Command5_Click()  
If Len(Text4.Text) = 0 Then Exit Sub  
Form4.Data1.RecordSource = "select * from hareket where  
cagri=" + Text4.Text + " order by parcano,hareket,tarih"  
Form4.Data1.Refresh
```

```

        gduz
        Unload Me
    End Sub

Private Sub Command6_Click()
    If Len(Text5.Text) = 0 And Len(Text6.Text) = 0 Then Exit Sub
    If Len(Text5.Text) = 0 Then
        deg6 = Val(Text6.Text)
        Form4.Data1.RecordSource = "select * from hareket where tutar=" + Str(deg6) + " order by parcano,hareket,tarih"
    End If
    If Len(Text6.Text) = 0 Then
        deg5 = Val(Text5.Text)
        Form4.Data1.RecordSource = "select * from hareket where tutar=" + Str(deg5) + " order by parcano,hareket,tarih"
    End If
    If Len(Text5.Text) > 0 And Len(Text6.Text) > 0 Then
        deg5 = Val(Text5.Text)
        deg6 = Val(Text6.Text)
        Form4.Data1.RecordSource = "select * from hareket where tutar>=" + Str(deg5) + " and tutar<=" + Str(deg6) + " order by parcano,hareket,tarih"
    End If
    Form4.Data1.Refresh
    gduz
    Unload Me
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        Command1_Click
    End If
End Sub

Private Sub Text4_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
        Command5_Click
    End If
End Sub

Private Sub Text5_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then Text6.SetFocus
End Sub

Private Sub Text6_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then Command6.SetFocus
End Sub

Private Sub Text2_KeyPress(KeyAscii As Integer)

```

Microsoft Visual Basic 6.0

```
If KeyAscii = 13 Then Text3.SetFocus
End Sub

Private Sub Text3_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then Command4.SetFocus
End Sub
```

RDO

RDO(Remote Data Object) hızlı bir client/server uygulama geliştirme aracıdır. Temel veritabanı işlemlerinizi kolaylıkla yapmanızı sağlar. Ayrıca gelişmiş özellikleriyle bir ODBC arabirimidir.

RDO,ODBC üzerinde ince bir katmandır. Sadece 32bit Visual Basic ile çalışır. İlişkisel veritabanlarına erişimde DAO'dan daha hızlıdır. ODBC sürücüsünün tanımlanması gerekmektedir. RDO ayrıca görsel arabirime de sahiptir.

RDO ve Jet programlamanın birbirine çok benzer yanları vardır. Veritabanı bağlantısı **rdoEnvironments** nesnesinin **OpenConnection** metodu ile sağlanır. Bağlantı kurulduktan sonra **OpenResultSet** metodu ile kayıtlara bir SQL sorgu ile erişilir. RDO ayrıca SQL deyimlerinden de yararlanmaktadır. RDO kullanmak için bazı gereksinimler vardır.

- Visual Basic 32bit Enterprise Edition
- ODBC sürücülerinin yüklü olması gerekir.
- SQL deyimleri
- Veri kaynağının kayıt edilmesi. DSN oluşturulması .

Ayrıca mevcut DAO nesnesi kolaylıkla RDO'ya çevrilir. RDO'da kayıtlar yerine satırlar, alanlar yerine de kolonlar kullanılır.

RDO 2.0'in Yenilikleri

- Yeni client batch cursor kütüphanesi
- Batch optimistik query'ler
- Olay temelli programlama
- Veriyi güncelleştirebilen RDO kontrolü
- Tek basına kullanılabilen RDO nesneleri
- Metodlar gibi kullanılabilen query'ler

RDO'nun Üstünlükleri

RDO, DAO kadar güçlü olmasıyla birlikte daha kolay bir erişim yöntemidir.

RDO, DAO ve Jet gibi veri-bağlı yöntemleridir. ODBC ve VB-SQL'e bağlantı için denetimler yoktur. Bu nedenle bu verilere erişim için kod yazmak gereklidir.

Jet/DAO bir ISAM veritabanı olarak tasarlanmıştır. Sadece veri içerirler. Herhangi bir veritabanı motoruna sahip değildir. Ancak SQL server gibi ODBC veri kaynakları kendi veritabanı motoruna sahiptirler. RDO, ODBC veri kaynakları için tasarlanmıştır. Bu nedenle bir veritabanı motoru içermez, RDO daha az sistem kaynağına sahiptir. RDO'nun sağladığı Remote Data Control ile hiç kod yazmadan ODBC veri kaynaklarına erişim sağlanır.

Bir Bağlantıyı Açmak

```
Dim env As rdoEnvironment
Dim con As rdoConnection
Set env=rdoEngine.rdoEnvironment(0)
Set con =env.OpenConneection("tablo")
rcon.Close
```

Veriyi Elde Etmek

```
Dim rQuery As rdoResultset
Dim sSQL As String
sSQL="select *from tablo alan="yyy""
Set rQuery=rCon.OpenResultset(sSQL)
Do While Not rQuery.EOF
    IstDocuments.AddItem rQueryTablo
    rQuery.MoveNext
Loop
rQuery.Close
```

Veritabanı Degistirme

RDO ile veritabanı üzerinde degistirme yapmak için iki yöntem vardır. Toplu degisiklik yada belli bir kaydın degistirilmesi. Toplu degisiklikte action queries (işlem sorguları) yada stored procedure'ler kullanılır. Belli bir kaydın degistirilmesinde ise Edit, AddNew, Update ve Delete metodları kullanılır.

```
Set env=rdoEngine.rdoEnviroments(0)
Set con=env.Open Connection("Ali")
Set tablo=con.OpenResultset("Tablo")
tablo.Addnew
```

Remote Data Control'un Kullanimi

ODBC üzerinden verilere daha hizli erisim için RDC (Remote Data Control) kullanilir. Remote Data Control, Data Control nesnesine benzer biçimde kullanilir. RDO görsel programla için RDC kontrolüne sahiptir. RDC kontrolü belli özellikler ile RDO üzerinde çalışarak verilere erisimi saglar. Bu özellikler sunlardir:

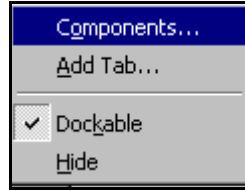
- **Connect** özelligi
- **Connenction** özelligi
- **ResultSet** özelligi

RDC'nin birçok özelligi DAO'nun DataControl'ü ile aynidir:

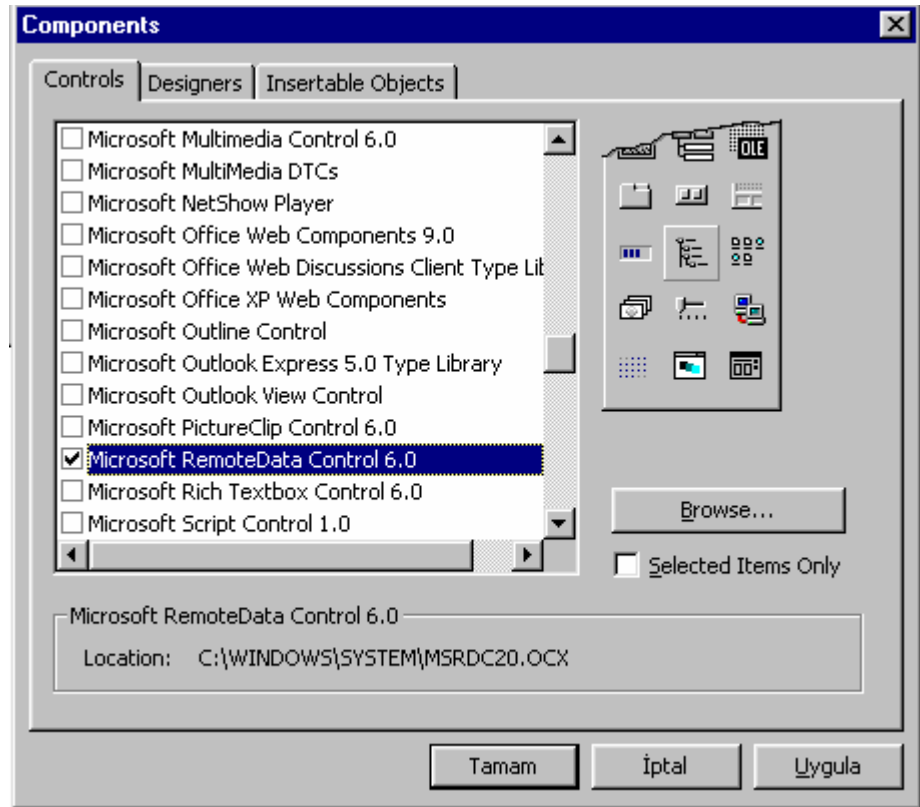
RDC Özellikleri	Data Control Özellikleri	Islevi
BOFAction	BOFAction	MovePrevious metodu kullaniminda dosya basina ulasildigini belirler.
DataSourceName	DatabaseName	Verilerine erisilecek olan veritabanini belirtir.
EOFAciton	EOFAciton	MoveNext metodunun kullaniminda dosyanin sonuna ulasildigini belirtir.
ResultSettype	RecordSetType	Elde edilen verilerin tipini belirler.
SQL	RecordSource	Elde edilecek belli bir veriyi belirler.


Rdc'nin kullanimi için önce RDC'nin projeye eklenmesi gerekir.

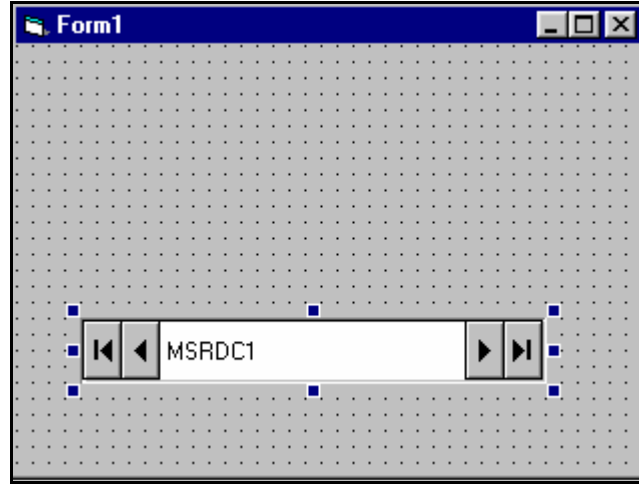
1. İşlem ilk olarak Componentes menüsünü açarak olusmaktadır.



2. Componentes menüsü ekrana geldiginde "Controls" bölümünden "Microsoft Remote Data Control" özelligine tiklariz.



3. Ve ToolBox menüsüne RDC'miz eklenmis olur. 
4. Bu Tool'a tikladigimiz zaman formumuza RDC'miz eklenir.



5. Name ve Caption özellikleri düzenlenir.
6. DataSourceName özelliği düzenlenir.

RDO ve RDC ilişkisel veritabanlarının erişim için kullanılan bir alternatif veri erişim yöntemidir. RDO ve RDC ile veri erişiminin yapılmasında üç önemli faktör vardır.

- Cursor Type(islem tipi)
- Dataset type(veri seti type)
- Lock Type(kilitleme tipi)

Cursor tipleri veri üzerinde hareket etmeyi ve işlem yapmayı sağlar. Cursor tipleri yönetimi client ve server olmak üzere ikiye ayrılır. Ayrıca Cursor tipi belirtilmede de işlem yapılabilir.

Dataset tipi ise uzak verilere erişimde önemli bir bilgidir. Değişebilir veri setleri kayıtların eklenmesine ve silinmesine olanak sağlar.

Kilitleme tipleri ise veri setlerinin güncellenmesinde kullanılır. Microsoft Jet veri erişiminde iki seçenek vardır.

- Pessimistic(kötümser)
- Optimistic(iyimser)

Pessimistic Cursor tiplerinde çok kullanıcıli ortamlarda bir kullanıcının programını başka bir kullanıcı kullanamaz.

DAO ve RDO Karsilastirilmesi

RDO ve DAO birbirlerine çok benzerler. Benzer nesneler:

RDO nesneleri	DAO nesneleri
rdoEngine	DBEngine
rdoEnvironment	Workspace
rdoConnection	Database
rdoTable	TableDef
rdoResultSet	Recordset
yok	Tablo tipi
Keyset tipi	DynaSet tipi
Static tipi	Snapshot tipi
rdoColumn	Field
rdoQuery	QueryDef
rdoParameter	Parameter

RDO ve DAO metodlari karsilastirilmesi

RDO metod	RDO nesne	DAO metod	DAO nesne
rdoCreateEnvironment	rdoEngine	CreateWorkspace	DBEngine
BeginTrans	rdoConnection	BeginTrans	Workspace
CommitTrans	rdoEnviroment	CommitTrans	Workspace
OpenConnection	rdoConnection	OpenDatabase	Workspace
RollBackTrans	rdoConnection	RollBack	Workspace
CreateQuery	rdoConnection	CreateQueryDef	Database
Execute	rdoConnection	Execute	Database
OpenResultSet	rdoConnection	OpenRecordset	Database

RDO ve DAO ortak metod ve islevleri:

AddNew	Yeni bir kayıt ekler.
Delete	Aktif kaydı siler.
Edit	Aktif kaydın üzerinde değişiklik yapar
MoveFirst	İlk kayda gider.
MoveLast	Son kayda gider.
MoveNext	Bir sonraki kayda gider.
MovePrevious	Bir önceki kayda gider.
Update	Yapılan değişikliklerin etkin olmasını sağlar.

RDO'nun zayıf Yönleri

- Sadece 32bit uygulamalar için tasarlanmıştır.
- ODBC sürücüsünün tanımlanması gerekir.
- ODBC API kadar hızlı değildir.

Access ve ISAM veritabanları üzerinde çalışabilen Jet kadar hızlı değildir.

ODBC

ODBC (Open Database Connectivity), Visual Basic kullanarak yerel ya da uzak server veritabanlarına ulaşmak için kullanılan yöntemdir. ODBC Microsoft tarafından oluşturulmuş bir yapıdır, standartlaşmıştır. Kullanıcı ODBC ile sağlanan kaynağa bağlantı yapar ve bu bağlantı ile bağdasan program ile ODBC nin ulaştığı kaynaktaki veriler üzerinde işlem yapılır.ODBC nin yapısı API fonksiyonlarından oluşur. ODBC sayesinde SQL server veya diğer veri sağlayıcıların sağladığı veri hizmetlerine ulaşmak mümkündür.

ODBC sürücüleri aslında birer DLL dosyasıdır. Bu DLL dosyaları bir veri sağlayıcısına bağlanmayı sağlayan spesifik fonksiyonları içerir.ODBC sürücüsü iletişimin bütün aşamalarını sağlar.

ODBC'nin temel görevleri;

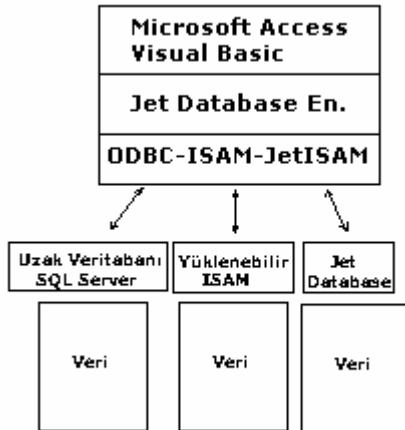
- SQL deyimlerini hazırlar ve iletir.
- Veritabanına bağlantıyı sağlar.
- Transaction'ları iletir.

- Sonuç bilgisini üretir.
- Uygulama hatalari konusunda kullanıcıyı uyarır, sorunun çözülmesini sağlar.

ODBC API leri, Microsoft Jet Database Engine (veritabanı motoruna) alternatif bir erişim olanakı sağlamaktadır, Microsoft Jet Database En.'den daha hızlı bir çalışma şekli vardır. ODBC SQL-Access veritabanlarına ulaşmanın yanı sıra ISAM olarak adlandırılan dBase, FoxPro gibi database kaynaklarına da erişim sağlar.

Visual basicte ise ODBC bağlantısı değişik biçimlerde yapılabilir.

- DAO/DC (Data Access Objects/Data Control)
- RDO/RDC (Remote Data Object/Remote Data Control)
- ODBC Direct
- ADO/ADODC (ActiveX Data Objects/Ado Data Control)



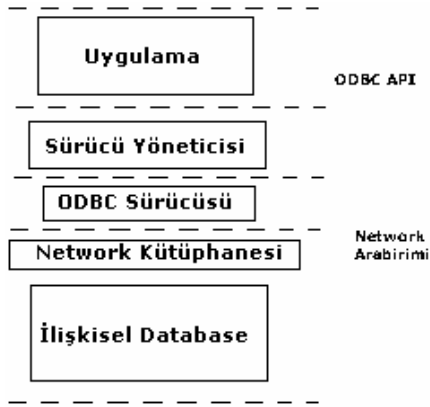
Sisteme yüklenmiş olan ODBC arayüzü Jet database engine tarafından Microsoft Access ve Visual Basic tarafından uzak veritabanlarına bağlanılmada kullanılır. Örneğin Microsoft SQL Server üzerindeki veritabanına Visual Basic ile ulaşmak için ODBC yüklenir.

ODBC Mimarisi

ODBC mimarisi dört bileşenden oluşur. Bu bileşenler;

Uygulama	ODBC fonksiyonlarını çağırarak onları SQL deyimlerine gönderir ve sonuçları elde eder.
Sürücü Yöneticisi	(Driver Manager) Bir uygulama yerine sürücülerini yükler.
Sürücü (Driver)	ODBC fonksiyonları çağırma işlerini yapar. SQL deyimlerini belli verilere yönlendirir.
Veri Kaynağı	Kullanıcıların erişeceği kaynaktır.

Sürücü yöneticisi seçilen sürücü (driver) ile Veri kaynağına ulaşır ve Uygulamanın veri gönderebileceği ve alabileceği şekilde tutar, hatalar olursa bu hataları tutar.



ODBC uygulamasının temel mimarisi

ODBC iki tip sürücü yapısına sahiptir.

- Tek katli sürücüler (single tier)
- Çok katli sürücüler (multi tier)

Tek katli sürücüler hem ODBC çağrılarını hem de SQL deyimlerini işleyebilirler. Çok katli sürücüler ise sürücü ODBC çağrısı işlerler ve SQL deyimini veri kaynağına aktarırlar. Tek katli sürücüler SQL olmayan veritabanları için kullanılır. Veritabanı dosyası doğrudan sürücü tarafından işletilir. Çok katli sürücülerde ise; sürücü isteğini hizmet birimine (sunucuya) gönderir. İstek SQL ya da özel veritabanı biçiminden olabilir. Çok katli sürücülerin çeşitleri ise İki katli sürücüler ve üç katli sürücülerdir.

ODBC Verisine Erisim

Varolan ODBC kaynağına erişim için aşağıdaki bileşenler gereklidir.

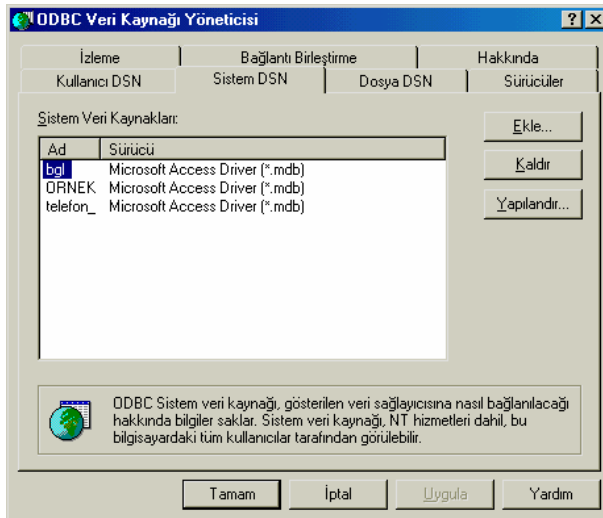
- Kaynak verinin adı.
- Sürücü ve diğer ilgili yazılım.
- Time-out değeri.(zaman asımı)

Sürücüler yüklenmesi gereken DLL dosyalarıdır. Time out değeri bağlantı zaman asımını temsil eder.

DSN Yapı

DSN (Data Source Name) veri kaynağına bağlanmak için kullanılan bir anahtardır.ODBC veri kaynakları çeşitlidir.

1-System DSN (Sistem DSN) : Sistem DSN Windows NT ve NT tabanlı sistemlerde kullanılır. Bütün uygulamalar ve servisler ona ulaşabilir.



Sistem DSN örneği

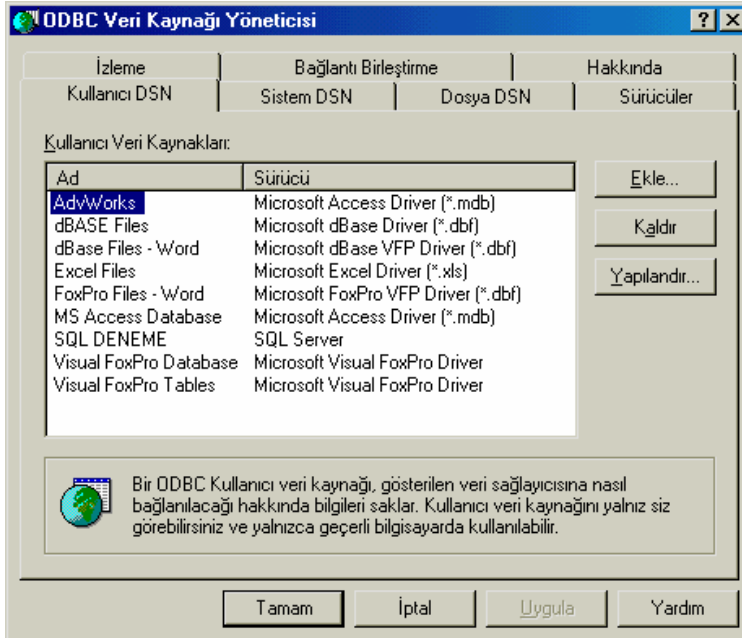
Microsoft Visual Basic 6.0

2-File DSN (Dosya DSN): DSN bilgileri bir text dosyasında saklanır.(Ayar dosyası)Bu dosya veritabanı sürücüsü ve yeri hakkında bilgi içerir. File DSN makineye özel değildir. Herhangi bir network sürücüsü üzerinde olabilir.



(Dosya DSN (File DSN) örneği)

3-User DSN (Kullanıcı DSN): Belirli bir user profile için kullanılabilir. DSN bilgisi lokal bilgisayarın Registry defterinde saklanır.

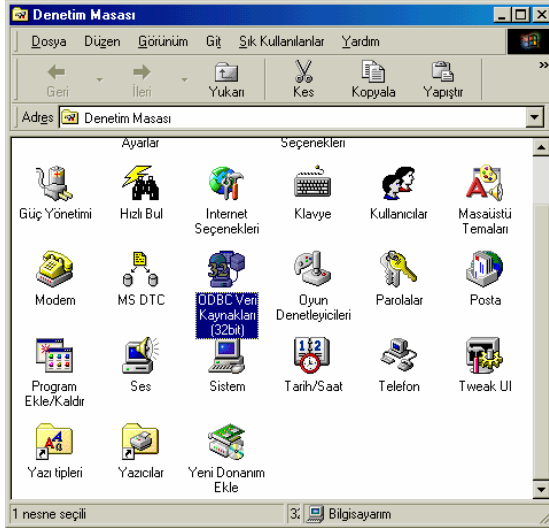


(User DSN (Kullanıcı DSN) Örneği)

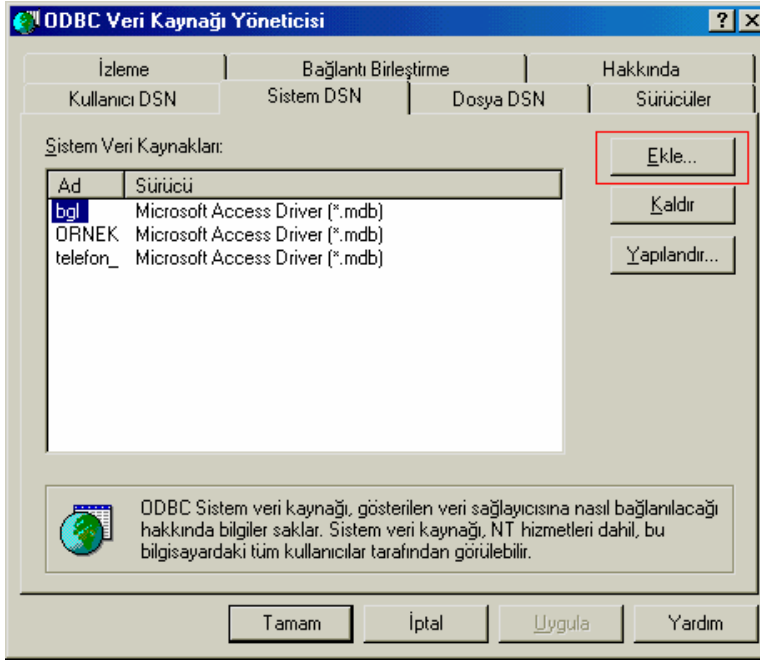
Adım Adım Windows 9x te DSN Tanimi

Visual basic ile hazırlanmış bir programda eger ODBC kullaniliyorsa programin kullanilabilmesi için ODBC DSN taniminin yapılması gerekir.

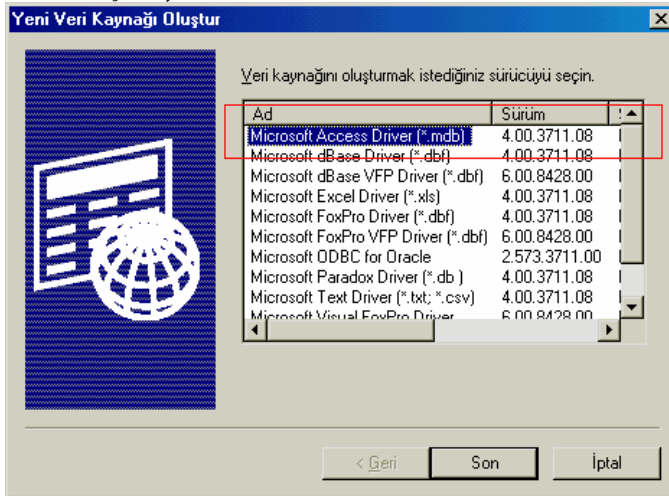
- Bu işlem için Denetim Masasındaki ODBC simgesine çift tıklanır.



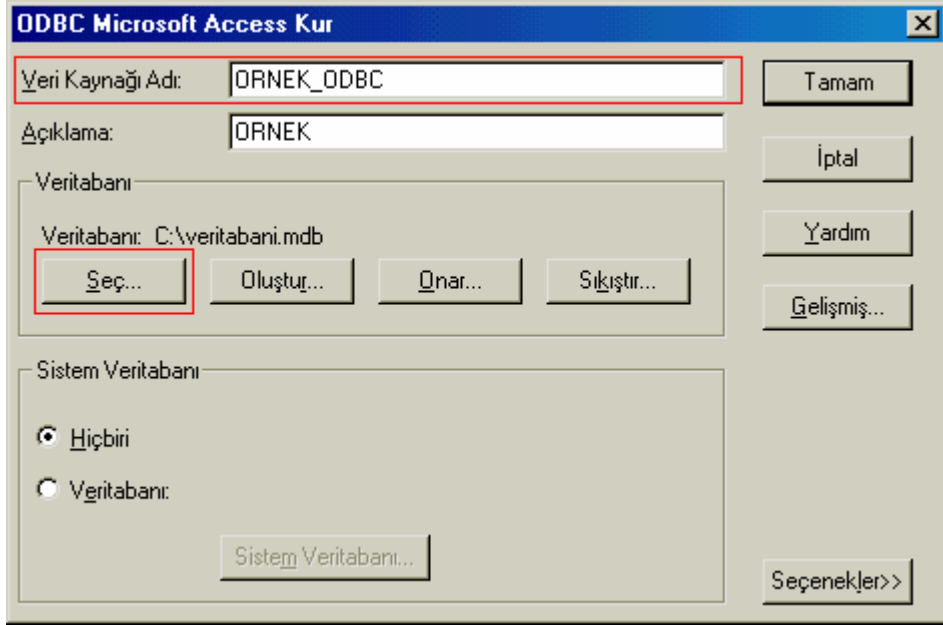
- Sonra tüm sistemde kullanılmak üzere tanımlanması gereken DSN olan Sistem DSN sekmesine geçilir ve Ekle.. (Add..) butonuna tıklanır.



- Ve eklenenecek ODBC yapısında kullanılacak veri sağlayıcısının kullandığı kitaplık seçimi yapılır (Örnekte Microsoft Access Driver (*.mdb)) (Not: Bu kitaplıkta yerel mdb dosyaları ODBC de kullanıldığı için bunda sonrası Microsoft Access Driver driverine spesifik işlemleri içerir)



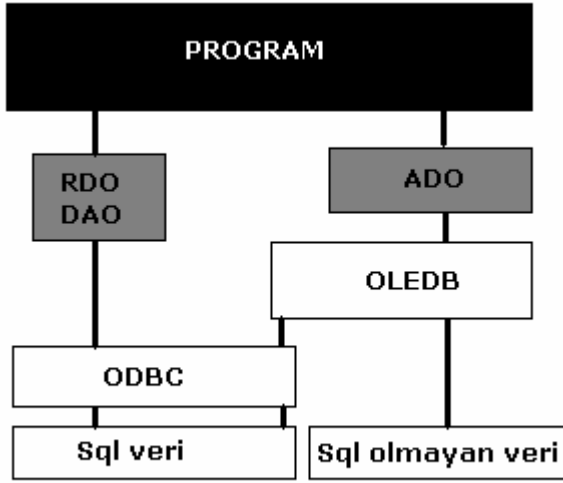
- İşlem onaylanır ve Database dosyasının seçildiği kısım ekrana gelir. Burada Veri kaynağı adı önemlidir çünkü kullanılacak DSN isim anahtarıdır. Ayrıca Seç... butonuna tiklanarak mevcut .mdb dosyaları içerisinde istenen dosya ODBC ye atanır.



Bu işlemler sonucunda sistem genelinde kullanılacak bir DSN yapısı oluşturuldu. Kitabın diğer bölümlerinde DAO, ADO, RDO gibi veri erişim yöntemleri ile ODBC DSN leri üzerinde yapılabilecek işlemler detaylı anlatılmaktadır.

ADO Veri Erişim Tekniği

Visual basic dilinde database uygulamaların fazla sayıda yapması Visual basic te database erişimi için birçok teknik olması ve yazılımı hazırlayan kişi(ler) için zaman kazandırabilecek derecede az kod yazılıp kolayca veritabanı üzerinde işlemler yapabilen bileşenlerin olmasından ileri gelmektedir. Database erişimi için kullanılan bir başka erişim yöntemi de ADO (ActiveX Data Objects) yöntemidir. ADO ve ADO Data Control (ADODC) kullanılarak çok az kod ile veritabanı erişimli uygulamalar geliştirebiliriz. Küçük uygulamalarda ADO ve ADO ya bağlı bileşenler ile çok basit şekillerde yazılım geliştirebiliriz. ADO OLEDB üzerinden verilere erişir.



Resim 1

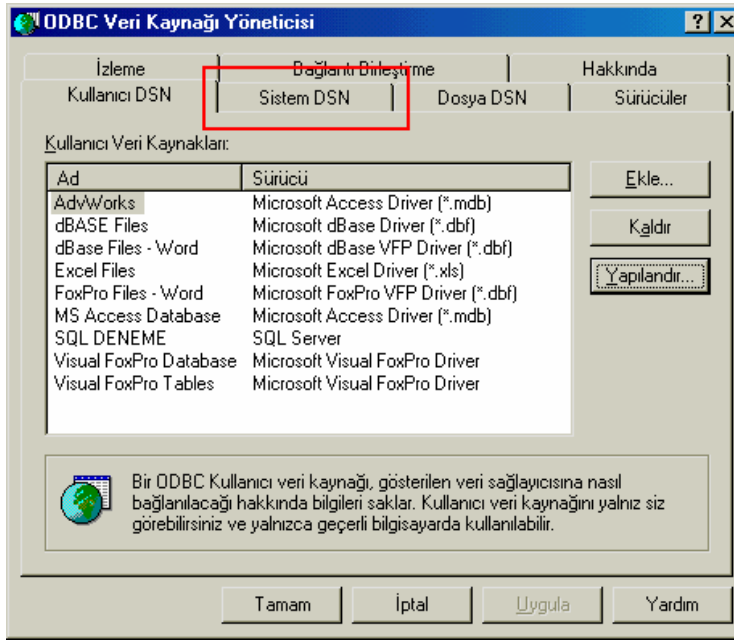
ADO çalışma şekli dolayısıyla ODBC bağlantısı kullanma ihtiyacı doğar. Yani programda kullandığımız Ado nesnemizi ODBC deki bir DSN (Data Source Name)'e bağlamalıyız. Yani bu işlemler için bir DSN yaratmalıyız.

Bu işlemler ODBC bölümünde daha kapsamlı anlatılmaktadır. Fakat aşağıdaki adım adım örneği ile bir DSN yaratma konusu anlatılmıştır

DSN Oluşturma;

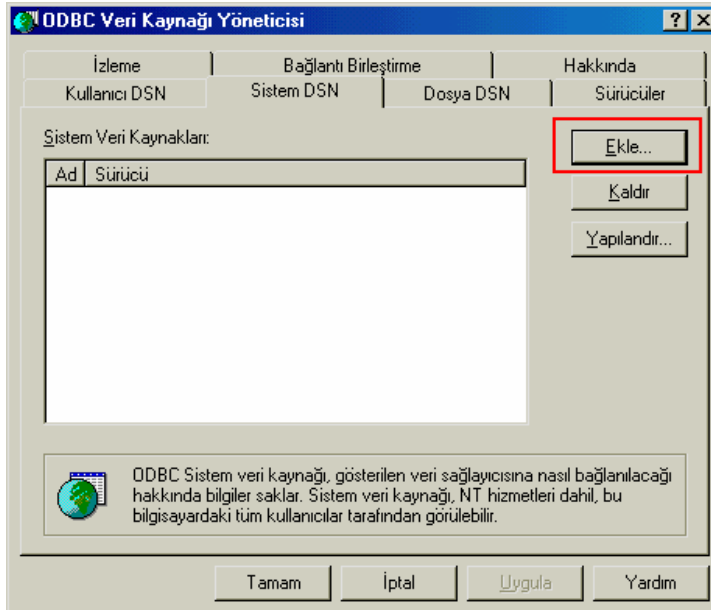
Aşağıdaki ekran görüntüleri Win98- ODBC den alınmıştır. Genelde ODBC diialogları birbirinin aynıdır.

1-Öncelikle ODBC çalıştırılır



(Resim 2)

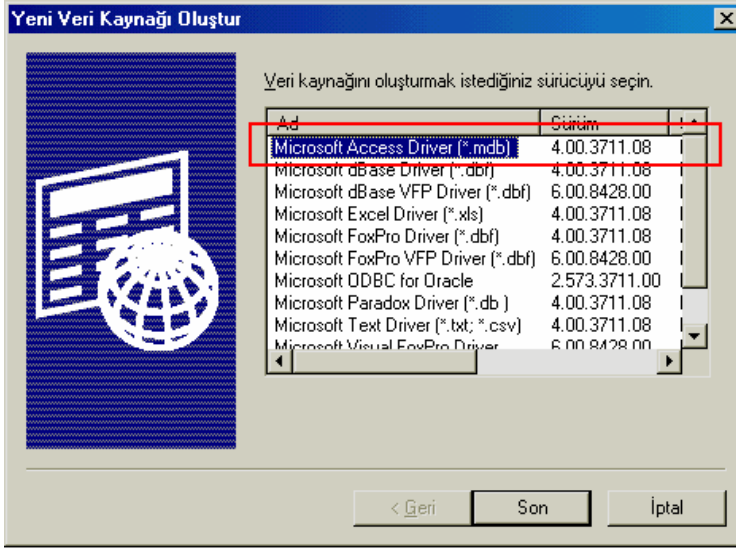
2-Sistem DSN Sekmesine geçilir Ekle butonuna tıklanır



(Resim 3)

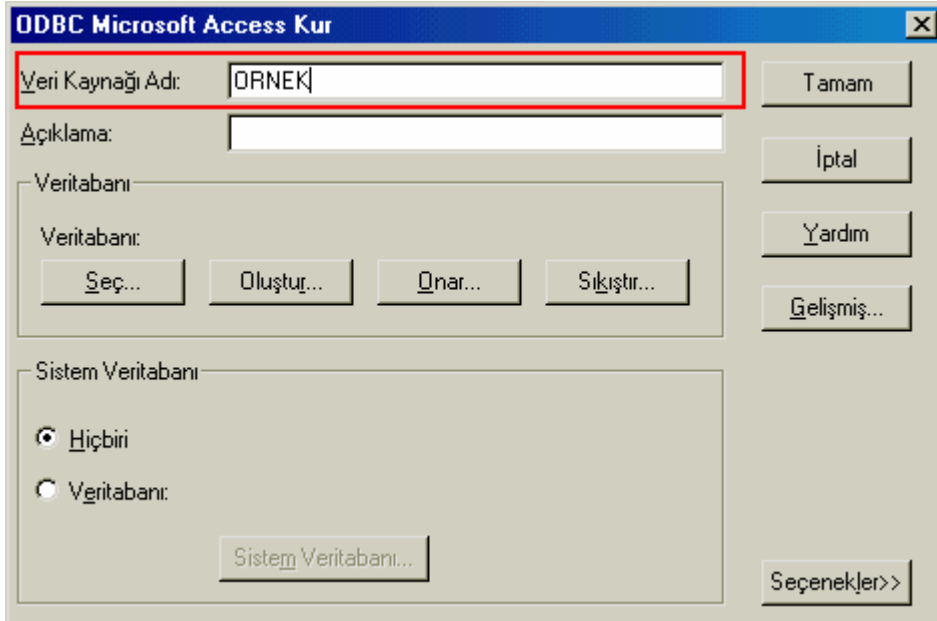
Microsoft Visual Basic 6.0

3-Açılan diolog kutusundan Mevcut sürücüler arasından kullanacağımız veritabanına uygun olan sürücü seçilir ve İşlem onaylanır.



(Resim 4)

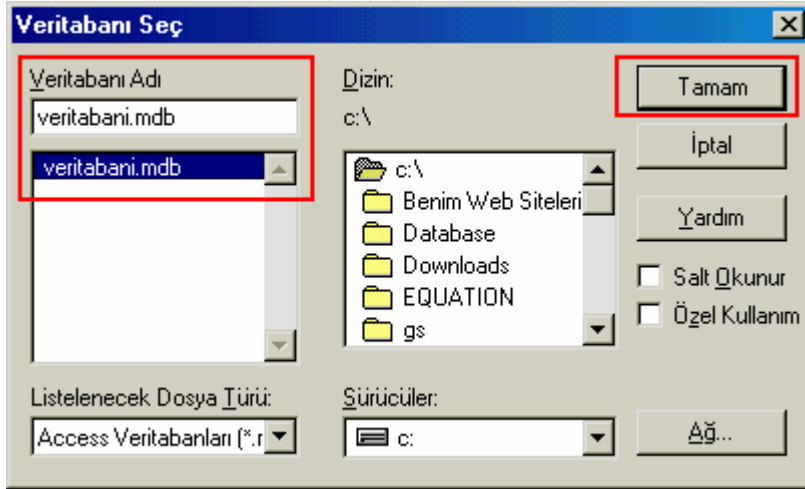
4-Açılan pencerede Veri kaynağı Adi kısmına DSN ismini yazarız .



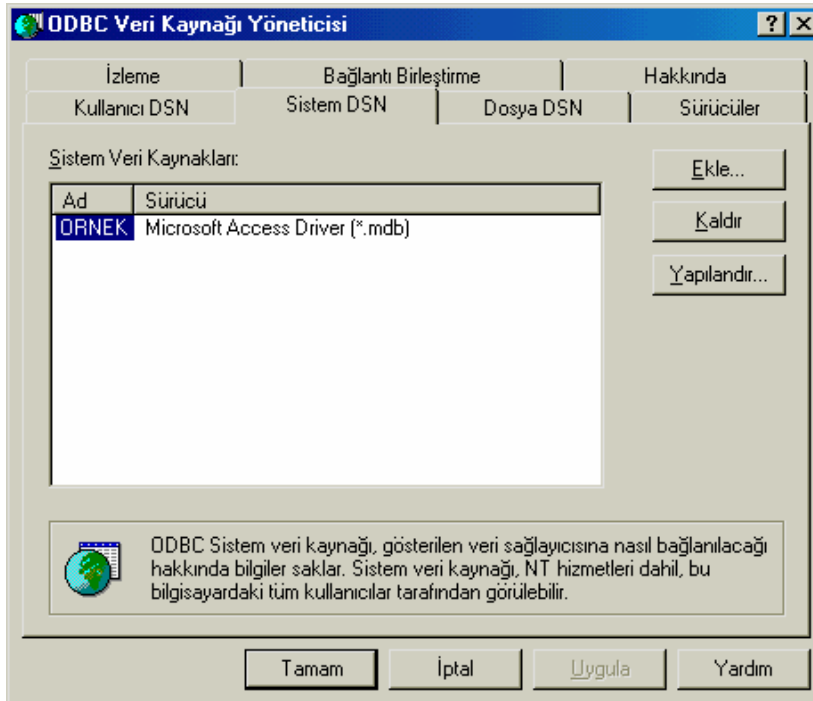
(Resim 5)

5- Seç... butonuna tıklayarak açılan pencereden veritabanını işaretleriz.

Ve veritabanı DSN ile bağlanmış olur bu andan itibaren DSN e ulanan programlar bu veritabanını kullanabilirler.



(Resim 5)

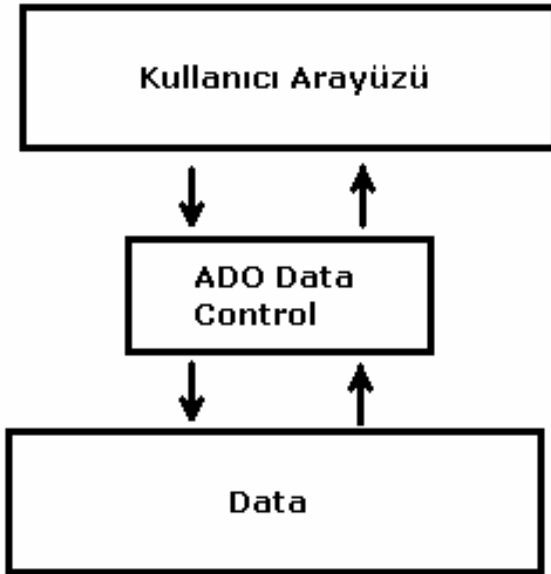


(Resim 6)

DSN olusturulduktan sonra ODBC de Sistem DSN altinda görünülenebilir.

ADO Data Control

ADO Data Control'un (ADODC) temel görevi data kaynaklari ile iletisim kurmaktır. Aslında ADO Data Control Oledb'yi kullandigi için fazla disk alanı kullanmayan ve daha hızlı bir yapı olduğu için daha öncelerden veritabanı uygulamalarında yaygın olarak kullanılan RDO ve DAO nun yerine geçebilecek şekilde tasarlanmıştır. Kullanım olarak DAO ve RDO ya benzer fakat daha kullanışlı ve daha hızlıdır. Bu fark ta ADODCnin ADO-OleDB kullanmasından kaynaklanmaktadır.



(Resim 7)

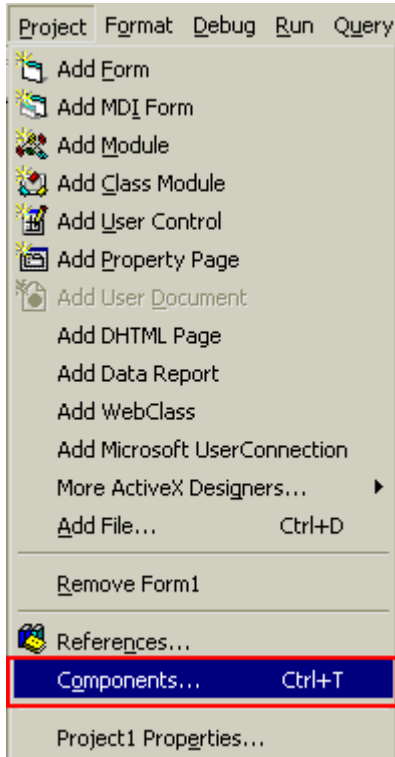
En basitinden ADO Data Control nesnesini form üzerinde kullanmak için

1-Ado Data Object'in kullanılması için ilkönce diğer componentler gibi projeye dahil edilmesi gereklidir.

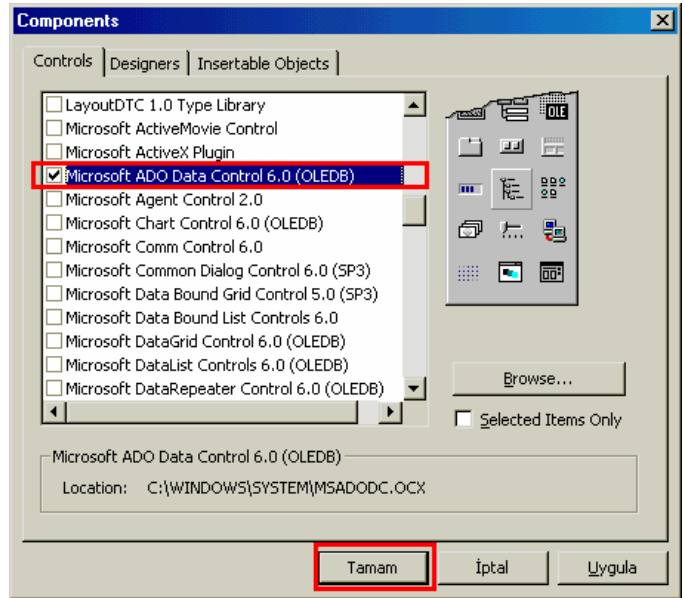
2-Daha sonra ADO Data Control nesnesi form üzerine yerleştirilir.

3-Gerekli ayarlar ve yapılır. (DSN – bağlantıları gibi)

1-Ado Data Object'ın kullanılması için ilkönce diğer componentler gibi projeye dahil edilmesi gereklidir.(Bu işlem için Project menüsündeki Components seçeneğine ardından açılan dialog kutusundaki Microsoft ADO Data Control 6.0 (OLEDB) seçeneğine tiklanarak yapılabilir)



(Resim 8)



(Resim 9)

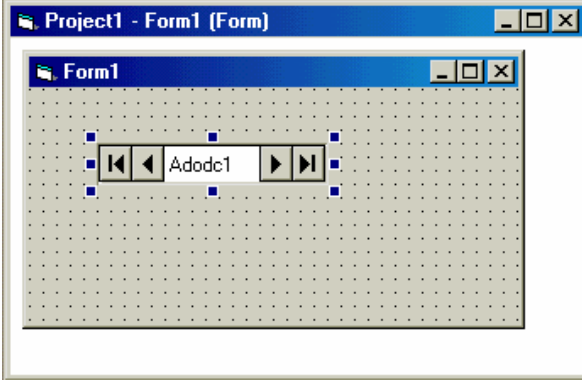
2-Daha sonra ADO Data Control nesnesi form üzerine yerleştirilir.

Component ekleme işleminden sonra Toolbox üzerinde aşağıdaki simge belirir. Bu simge ADO Data Control'un simgesidir.



(Resim 10)

ADO'nun form üzerine yerleştirilmesi diğer componentlerden farklı değildir.



(Resim 11)

Resim Form üzerine ADO Data Object eklendikten sonra çekilmiştir

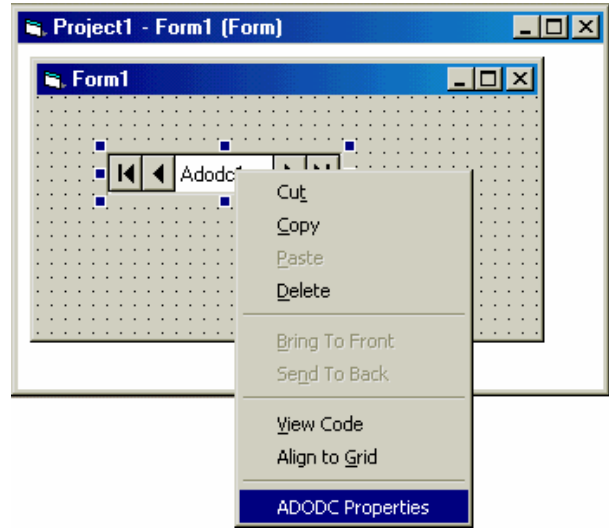
AdoDC nin yapılandırılması ;

ADO Data Control Form üzerine yerleştirildikten sonra kullanılacağı amaca göre bazı ayarların yapılması gerekmektedir. Bu ayarlar ADODC üzerine farenin sağ tuşu ile tıklandığında açılan popup menüden **ADODC Properties** seçeneğine tıklanarak yapılabileceği gibi ADODC işaretli iken **Properties** penceresindeki (Custom...) seçeneğine tıklanarak da yapılabilir.

Connection String Ayarları ;

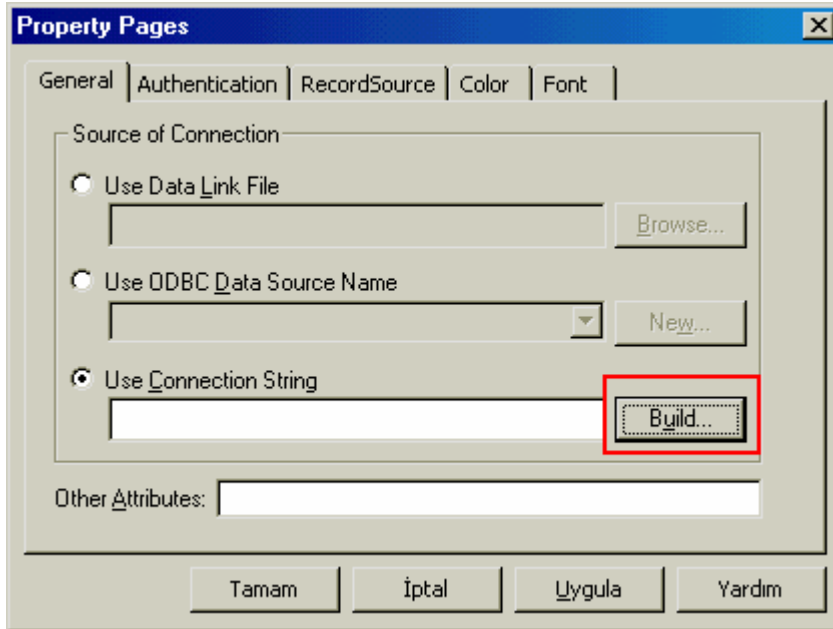
Connection String , bağlantı cümlecigidir bu cümlecikte ADODC nin hangi data kaynağına bağlanılacağı, DSN ismi vb. bilgiler bulunur. Örneğin Sistemimizde MS-Access ile hazırlanmış veritabanı.mdb database dosyamızı ORNEK isimli DSN ile tanımlamış olalım (DSN tanımlama işlemi önceki ADO Veri Erişim tekniği başlığı altında ayrıntılı olarak anlatılmıştır.) Bu işlem için aşağıdaki adımlar izlenebilir;

Adım 1-Form üzerinde mevcut olan ADODC ' ye farenin sağ tusu ile tıklayıp **ADODC Properties** seçeneği seçilir.



(Resim 12)

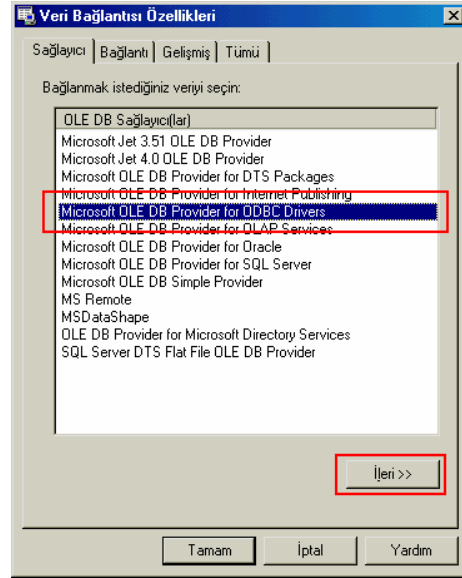
Adım 2-İşlem sonucunda **Property Pages** penceresi açılır.



(Resim 13)

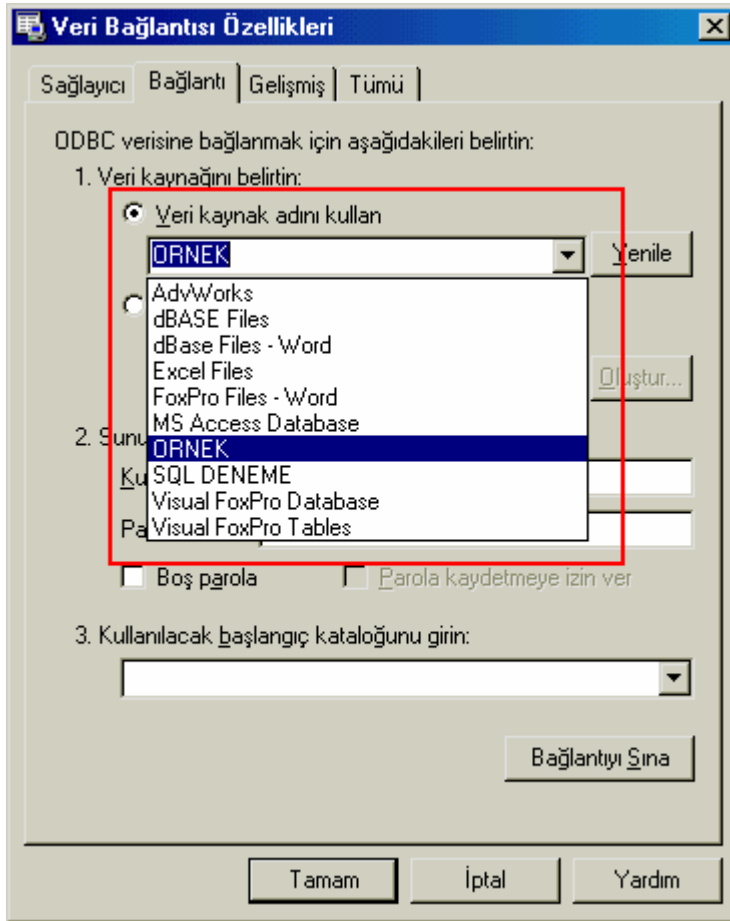
Microsoft Visual Basic 6.0

Adim3- Bu pencerede General, Authentication, RecordSource, Color, Font gibi ayar sekmeleri bulunur. General sekmesi DSN nesnesi ile baglanti kurulmasi için gerekli ayarlarin yapilacagi kisimdir. **Use Data Link File** seçenegi seçilirse DSN, UDL formatindaki daha önceden hazirlanmis sablon dosyalari kullanilir. **Use ODBC Data Source Name** seçenegi seçilirse olusturulmus olan DSN direkt olarak ADODC ye baglanir. Bu sekmeden **Use Connection String** seçenegi ile baglanti cümlesi olusturma dialog penceresi ekrana getirilir. Bu pencereden Microsoft OLE DB Provider for ODBC Drivers seçenegi tiklanarak ileri (next) butonuna tiklanidiginda ekrana ODBC DSN isminin seçildigi pencere gelir.



(Resim 14)

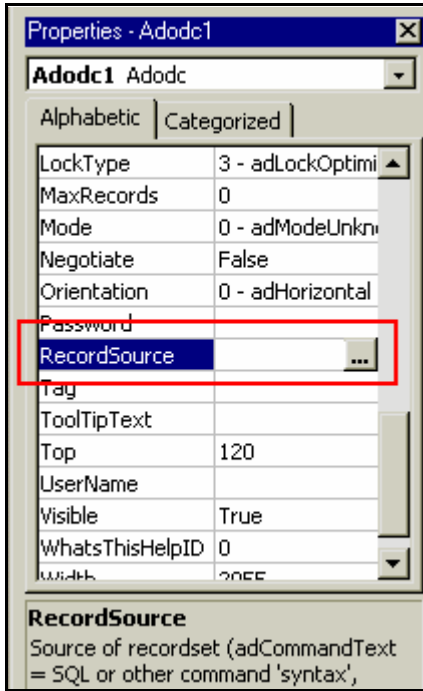
Adim 4- Bu pencerede daha önceden tanımlanmış olan ODBC DSN ' i seçilir. Eger baglanti kontrolu yapılmak isteniyorsa Baglantiyi sina (Test Connection) butonuna tiklanarak kontrol yapılabilir.



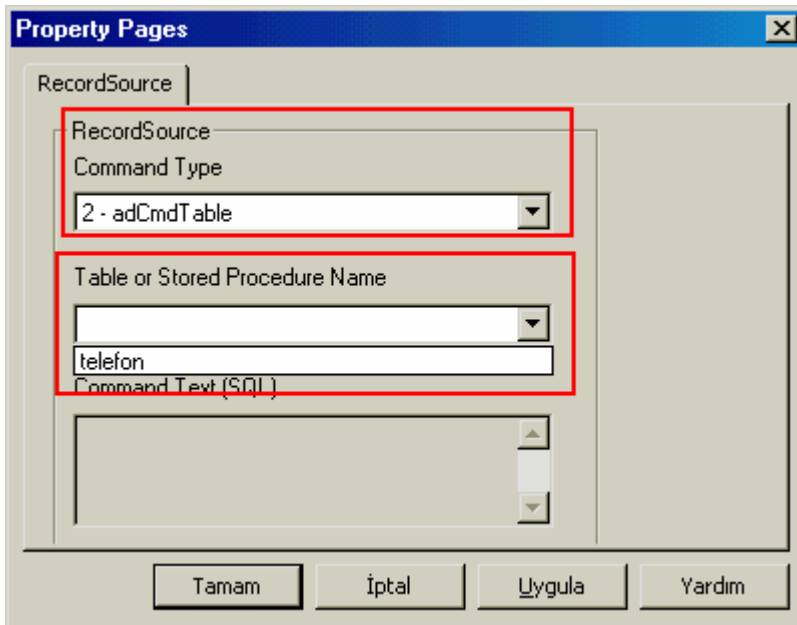
(Resim 15)

Yukarıdaki işlemler sonucunda bağlantı cümlecigi "Provider=MSDASQL.1;Persist Security Info=False;D" şeklinde oluşturulmuş olur.

Bu işlemler sonucunda ADODC bileşeni Database nesnesine bağlanmış olur ancak işlem yapılması için **Recordsource** (Kayıt kaynağı) bölümünde ayarlanması gerekir. Recordsource ayarı **Properties** penceresindeki Recordsource seçeneğine tiklanarak yapılabilir.



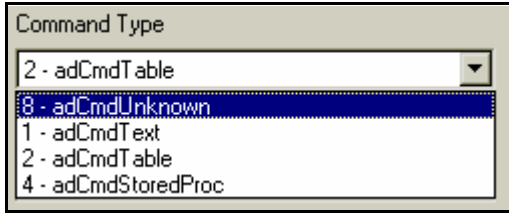
(Resim 16: ADODC nesnesinin properties bilgileri)



(Resim 17 ADODC nesnesinin **Properties** penceresindeki recordsource kısmına tıklandığında açılan pencere)

Resim 17 de diolog penceresinde **Commandtype** seçeneği 2-adCmdTable olarak seçilmiştir bu değer ADODC nin var olan DSN in bağlı olduğu database dosyasının içindeki bir tabloya bağlanacağını gösterir. Commandtype seçeneği Resim 17 deki gibi 2-adCmdTable olarak seçilirse hemen altındaki **Table or Stored Procedure** seçeneğine tiklandığında mevcut tablolar görüntülenir. Resim 17 de, örneğimizde de telefon adındaki tablo recordsource olarak seçilmiştir.

Resim 17 de görüntülenen CommandType seçenekleri Resim 18 de gösterilmektedir



(Resim 18)

Burada listelenen seçeneklerden

1-adCmdText seçeneği SQL dili ile recordsource tanımının yapılmasını,

2-adCmdTable seçeneği recordsource olarak veritabanını içindeki belirli bir tablonun seçilmesini,

4-adCmdStoredProc seçeneği ise recordsource tanımının StoredProcedureler yardımı ile yapılmasını sağlar.

Buraya kadar anlatılan konfigürasyon işlemleri projenin minimal olarak database dosyası ile iletişimini sağlamak için tasarım aşamasında iken yapılan ayarlamalardır.

ADODC yapısı diğer araçlar gibi projenin çalışması esnasında da konfigure edilebilir.

ADO Data Control'un Proje Çalışırken Düzenlenmesi

ADODC nesnesinin çalışma esnasında yapılacak ayarlamaları tasarım aşamasında yapılabilecek ayarlar ile aynıdır.

Örneğin Yukarıda bağlantı nesnesi yaratılmıştı, bağlantı nesnesi ODBC DSN'ine bağlanmıştı, ve CmdType seçilip Recordsource seçilmiştir. Yukarıdaki ayarlar eğer proje çalışırken ayarlanmak veya değiştirilmek istenirse aşağıdaki gibi bir yol izlenmelidir.

(Not :Öncelikle ADODC forma dahil edilmelidir.)

Projenin ilgili kismina;

```
Adodc1.ConnectionString = "ORNEK"  
Adodc1.CommandType = adCmdTable  
Adodc1.RecordSource = "telefon"
```

Kodlarinin eklenmesi yeterlidir.

DSN kullanılmadan Baglantinın Yapılması

DSN kullanılmaksizin JET OLEDB saglayicisi ile bir veritabanına baglanti yapilacaksa önemli olan ConnectionString özelliginin dikkatlice ayarlanması gerekmektedir.

Yukarıdaki baglanti DSN 'siz olarak JET OLEDB ile yapilacaksa kodlar

```
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=C:\veritabani.mdb;Persist Security Info=False"
```

```
Adodc1.CommandType = adCmdTable  
Adodc1.RecordSource = "telefon"
```

sekinde olmalıdır.

Form Elemanlarının ADO Data Kontrol ile ilişkisi

Visual Basic'te kullanicidan bilgi girisi yapilasi istenen bileşenlerin ADODC ile baglanması hem proje dizayn asamasında hemde proje çalışırken mümkündür.

Visual basic bileşenleri incelendiğinde (textbox,label,listbox,combobox gibi) ortan özelliklere sahip oldukları görülür bu ortak özelliklerden ADODC ile bağlantılarının yapılmasını sağlayan özellikleri Datasource ve Datafield özellikleridir.Bu özellikler kullanılarak ADODC 'nin record source bilgisindeki veriler kolayca form elemanları tarafından listelenebilir.

Örneğin veritabani.mdb dosyasına ADODC ile OLEDB ile bağlanan, ve label nesnelerinde veritabanının içindeki telefon tablosundaki verileri görüntülemek istersek aşağıdaki adımları izleyebiliriz (Projenin çalışması esnasında).

(Not : veritabani.mdb dosyasının içindeki telefon tablosunda kisi_ad, kisi_soyad,kisi_tel,kisi_adres,kisi_no gibi alanlar bulunmaktadır.)

Adım 1:Form üzerine gerekli nesneler yerleştirilir.




(Resim 19)

Adım 2: Resim 19 da Form üzerine nesnelerin yerseltirildiği görülmektedir. Burada label6 kisi_ad, label7 kisi_soyad, label8 kisi_tel, label9 kisi_adres, label10 kisi_no alanlarını göstermek üzere yerlerine yerleştirilmiştir. Command2 butonu ("Bağlantıyı Yap Ve Label Nesnelerini Ayarla" captionlu buton nesnesi) ise ADODC yi veritabanına bağlayan, form üzerindeki nesneleri de ADODC ye bağlayan kodları barındırır. İçeriği aşağıdaki gibidir.

```
Private Sub Command2_Click()  
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=\veritabani.mdb;Persist Security Info=False"  
Adodc1.CommandType = adCmdTable  
Adodc1.RecordSource = "telefon"  
Set Label6.DataSource = Adodc1  
Set Label7.DataSource = Adodc1  
Set Label8.DataSource = Adodc1  
Set Label9.DataSource = Adodc1  
Set Label10.DataSource = Adodc1  
Label6.DataField = "kisi_ad"  
Label7.DataField = "kisi_soyad"  
Label8.DataField = "kisi_tel"  
Label9.DataField = "kisi_adres"  
Label10.DataField = "kisi_no"  
End Sub
```

Microsoft Visual Basic 6.0

Program alistirildiginda label nesneleri ilk etapta alanlarin ierigini listelememektedir.Fakat Command2 butonuna tiklandiginda gerekli baglantilar yapilir ve form nesneleri ADODC1 nesnesine baglanir ve eger adodc1 nesnesinin birsonraki kayit anlamindaki  simgesine tiklanmasi sonucunda label nesnelerinin ieriklerinin degistigi grlr.

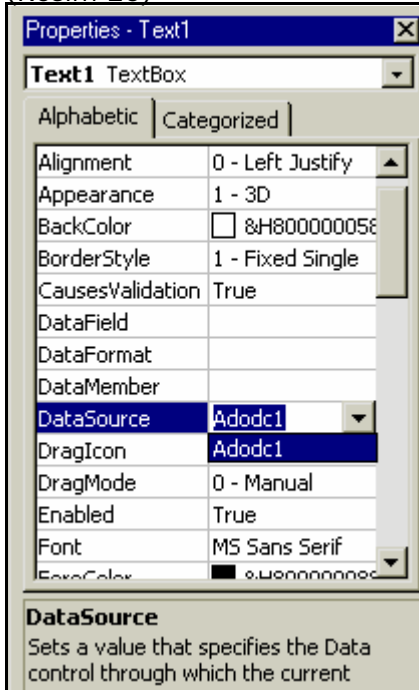
rnek Program kitap ile verilen rnek Program CD si ierisinde bulunmaktadır.

Yukarida form nesnelerinden Label nesnesinin adodc1 nesnesine program alisirken baglanmasi ile ilgili bir rnek incelendi. Konunun basinda form elemanlarinin program alismasi esnasinda ve dizayn asamasinda ADODC nesnesine baglanabilecegi sylenmistir.Dizayn asamasinda bir text nesnesinin ADODC nesnesine baglanmasi iin asagidaki yol izlenebilir.

Adim 1: ncelikle form zerindeki nesneler yerlerine yerlestirilirler.(ADODC, Label, Text nesneleri)

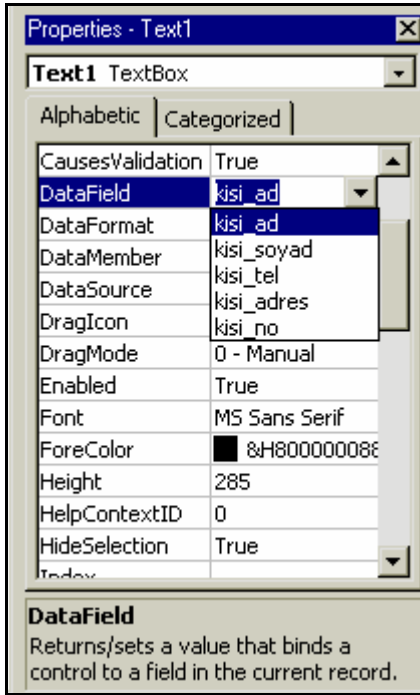
Adim 2:Adodc ConnectionString ve RecordSource ayarlamalari yapilir

Adim 3: Textbox nesnesinin Datasource ayari Ado nesnesi ile baglanir
(Resim 20)



(Resim 20)

Adim 4: Textbox nesnesinin DataField ayari bagli bulundugu ado nesnesinin de bagli bulundugu recordsource içerisindeki listelenen alanlarin birinin seçilmesi ile son bulur (Resim 21).



(Resim 21)

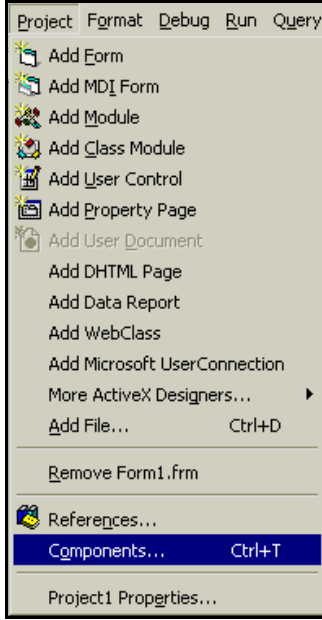
Bu işlemler sonunda program çalıştırıldığında ADODC nesnesi veritabanı dosyasından verileri aldığı anda textbox un içeriği yapılan ayarlamalar doğrultusunda değişir.

DataGrid ,DataList,DataCombo Bileşenleri

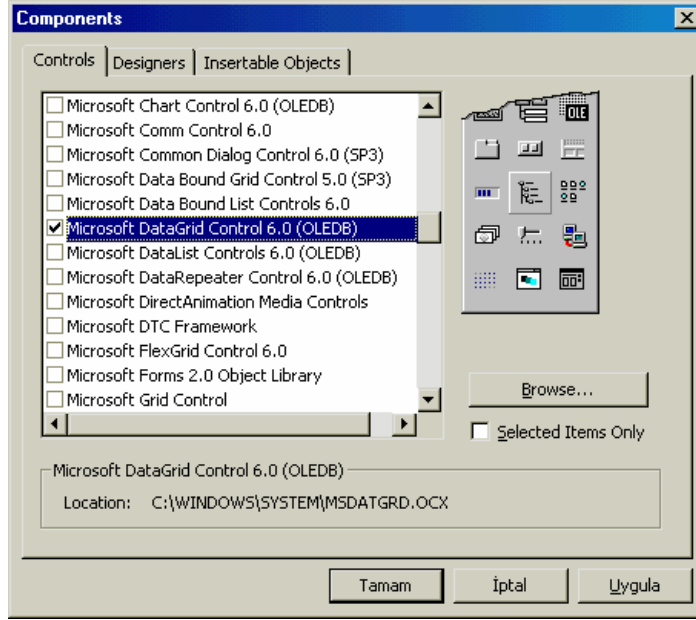
DataGrid Bileşeni;

DataGrid bileşeni Dbgrid bileşenin daha gelişmiş seklidir denilebilir.DataGrid bileşeninin kullanılması için diğer bileşenler gibi projeye dahil edilmesi gereklidir.

Bu işlem için Projects menüsünden Components (bkz resim 22) seçeneği seçilir.Açılan pencereden Microsoft Datagrid Control 6.0 (bkz resim 23) seçilir.Unutulmaması gereken nokta projeye Microsoft Ado Data Control 6.0 bileşeninin de eklenmiş olması gerektiridir.

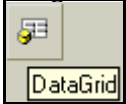


(Resim 22)



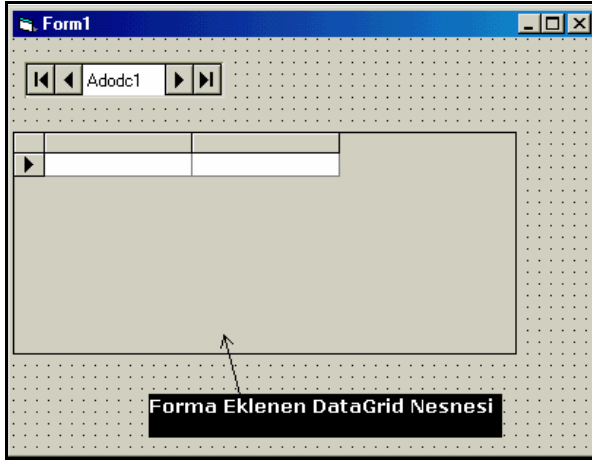
(Resim 23)

Projeye eklenen DataGrid nesnesinin ikonu Toolbox'ta Resim 24 te simge şeklinde yer almaktadır



(Resim 24)

Datagrid nesnesi diğer grid nesnelere benzer bir görüntüye sahiptir. (bkz Resim 25)



(Resim 25)

DataGrid nesnesinin ADO Data Control nesnesi ile beraber kullanımı diğer nesneler ile aynıdır. DataGrid nesnesinin ADO data control ile beraber kullanılması için DataSource özelliğinin formdaki ADO Data Control nesnesinin ismine esitlenmesi gerekmektedir.

DataGrid nesnesinin DataSource ayarlaması dizayn aşamasında yapılabileceği gibi projenin çalışma esnasında herhangi bir yordamda da yapılabilir.

Örneğin datagrid1 isimli datagrid nesnesi nin formda Command1 butonunun Click yordamında DataSource ayarının değiştirilmesi için aşağıdaki kod kullanılabilir.

```
Private Sub Command_Click()  
Set DataGrid1.DataSource = ADODC1  
End Sub
```

DataGrid nesnesi Dbgrid nesnesinde olduğu gibi gösterilen veriler üzerinde yeni kayıt ekleme, kayıt silme, güncelleme gibi işlemlerin yapılmasına izin verir.

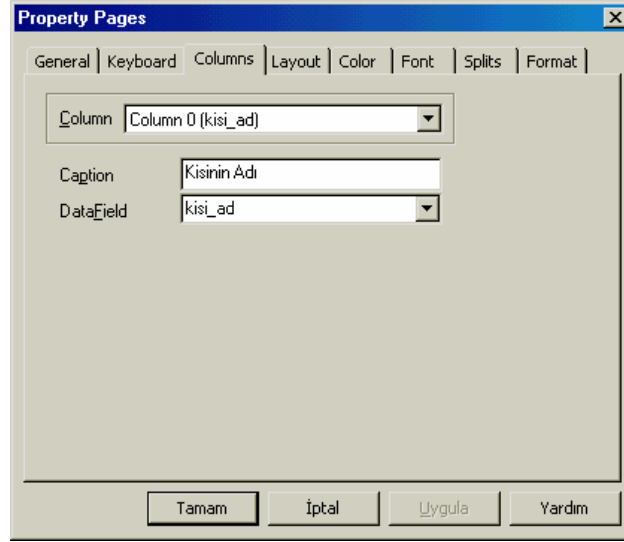
(DataGrid nesnesinin Properties penceresindeki özellikleri incelendiğinde bu işlemlere izin verilip verilmeyeceğine dair ayar özellikleri görülebilir (Bkz Resim 26))

AllowAddNew	False
AllowArrows	True
AllowDelete	False
AllowUpdate	True

(Resim 26)

DataGrid bagli bulunduđu Ado Data Control nesnesinin RecordSource'indeki bilgileri listeler, bu listelemede eger istenirse sütun basliklari degistirilebilir. Bu yarlara kolaylikla DataGrid bileseninin properties penceresindeki özelliklerinden (Custom...) seçenegine tiklanmasi sonucunda açılan diolog

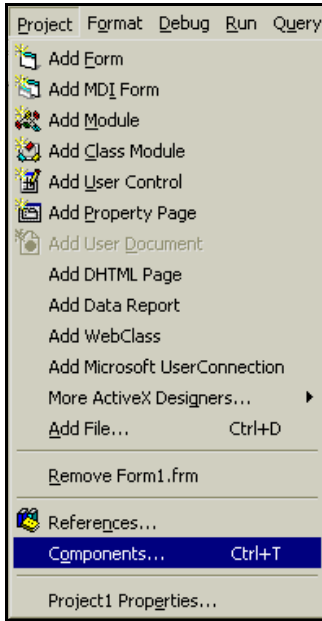
Penceresinden yapılabilir. (Resim 27 'de kisi_ad isimli alanin Kisinin Adi basliginda listelenmesi için gerekli ayar yapılmaktadır.)



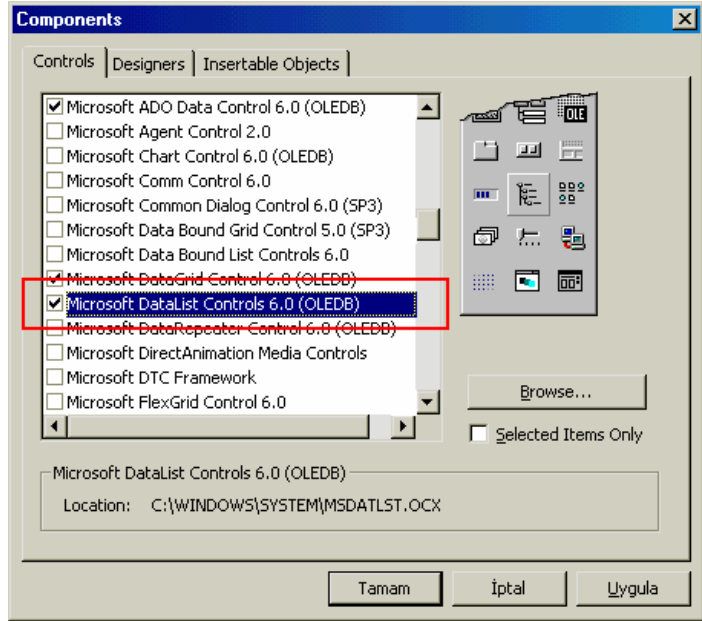
(Resim 27)

DataList Bileseni

Datalist bilezeni DBList bileseninin yerine kullanilmasi için tasarlanmistir.Verilerin listelenmesi amaçli kullanilir, kullanilmasi için öncelikle Projeye eklenmesi gerekir.Bunun için **Projects** menüsünden **Components** (bkz resim 28) seçenegi seçilir.Açılan pencereden **Microsoft DataList Controls 6.0** (bkz resim 29) seçilir.Unutulmaması gereken nokta projeye Microsoft Ado Data Control 6.0 bileseninin de eklenmiş olması gerektigidir.



(Resim 28)



(Resim 29)

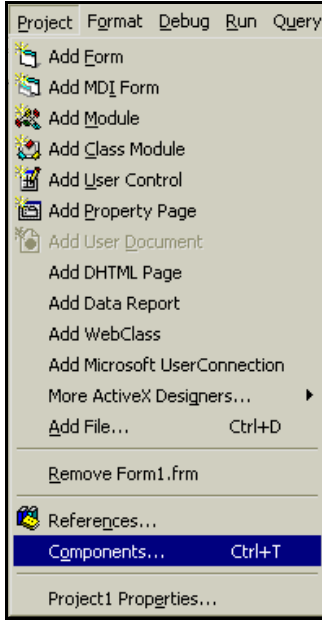
Datalist kontrolünde ListField ve RowSource özellikleri kullanılır.(Bu özelliklerin kullanım amacı liste içindeki gösterilecek verileri belirlemektir.).DataSource ve DataField özellikleri bilesen üzerindeki verinin nereye yazılması gerektiğini belirler.

Aşağıdaki örnekte Command1 butonunun Click yordamında Datalist1 nesnesinin "kisi_ad" isimli alanı listelemesi için gerekli ayar yapılmaktadır.

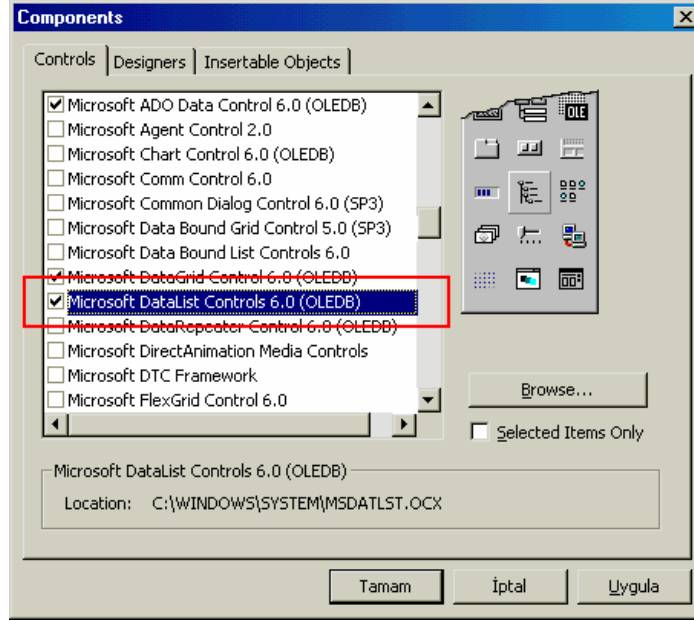
```
Private Sub Command1_Click()  
DataList1.ListField="kisi_ad"  
Set DataList1.RowSource=ADODC1  
End Sub
```

DataCombo Bileseni;

DataCombo bileseni DBCombo bileseninin yerine kullanılması için tasarlanmıştır.Verilerin listelenmesi amaçlı kullanılır, kullanılması için öncelikle Projeye eklenmesi gerekir.Bunun için **Projects** menüsünden **Components** (bkz resim 30) seçeneği seçilir.Açılan pencereden **Microsoft DataList Controls 6.0** (bkz resim 31) seçilir.Unutulmaması gereken nokta projeye Microsoft Ado Data Control 6.0 bileseninin de eklenmiş olması gerektiridir.



(Resim 30)



(Resim 31)

DataCombo kontrolünde ListField ve RowSource özellikleri kullanılır. (Bu özelliklerin kullanım amacı liste içindeki gösterilecek verileri belirlemektir.) DataSource ve DataField özellikleri bileşen üzerindeki verinin nereye yazılması gerektiğini belirler.

Aşağıdaki örnekte Command1 butonunun Click yordamında DataList1 nesnesinin "kisi_ad" isimli alanı listelemesi için gerekli ayar yapılmaktadır.

```
Private Sub Command1_Click()  
DataList1.ListField="kisi_soyad"  
Set DataList1.RowSource=ADODC1  
End Sub
```

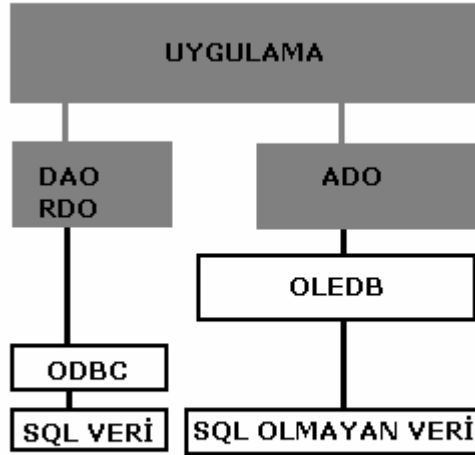
NOT : Eger listelenen kayıtlar sadece okunabilir kılınmak isteniyorsa DataSource ve DataField kısımları boş bırakılır

ADO (ActiveX Data Objects) Temelleri

Ado nesnesi mevcut bir veri sağlayıcısı sayesinde uzak yada yerel database'lere erişmeyi sağlayan bir yöntemdir. ADO bağımsız olarak kullanılabilen ve hiyerarsik olmayan bir yapıdadır, bu tipte tanımlanan nesneler ile verilere kolayca ulaşılabilir.

ADO hızlı bir şekilde çalışır böylece veritabanı uygulamaları daha etkin bir şekilde gerçekleştirilebilir. Bellek gereksinimi diğer tekniklere göre daha

azdır.RDO,DAO gibi veri erişim tekniklerinden daha gelişmiştir, daha anlaşılabilir ve kolay bir yapıda veritabanı nesneleri ile veritabanı işlemlerini sadeleştirir.



**Diğer Veri Erişim Yöntemleri İle
ADO Mimarisinin Karşılaştırılması**

(Resim 32)

ADO sunucu,istemci veya wrb uygulamalari gelistirmek için birçok özellige sahiptir.

ADO'nun Özellikleri;

- Sunucu-istemci uygulama gelistirmeyi kolaylastirmistir.
- Nesneler bagimsiz yaratilir.
- Nesne modeli hiyerarsik degildir.
- Batch halinde güncelleme yapılabilir.(İşlemler toplu halde yapılabilir.)

ADO OLEDB veri saglayicisini kullanir. RDO,DAO,ODBC ise Jet veri veri saglayicisini kullanir.

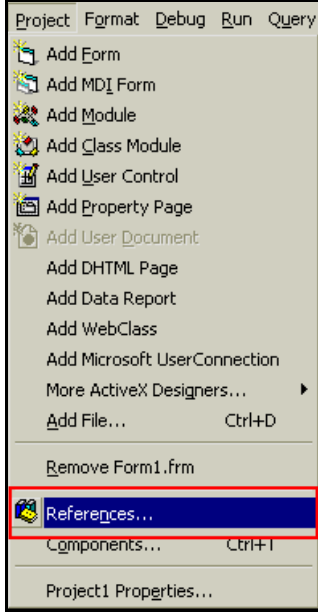
OLEDB Hakkında;

OLEDB ilişkisel ve ilişkisel olmayan veritabanı kaynaklarına erişimde kullanılır, hızlı bir arabirimdir. Daha az disk alanı ve bellek harcar OLEDB ODBC'ye göre daha fazla veri tipine erişim sağlayabilir.(Örn posta,dosya

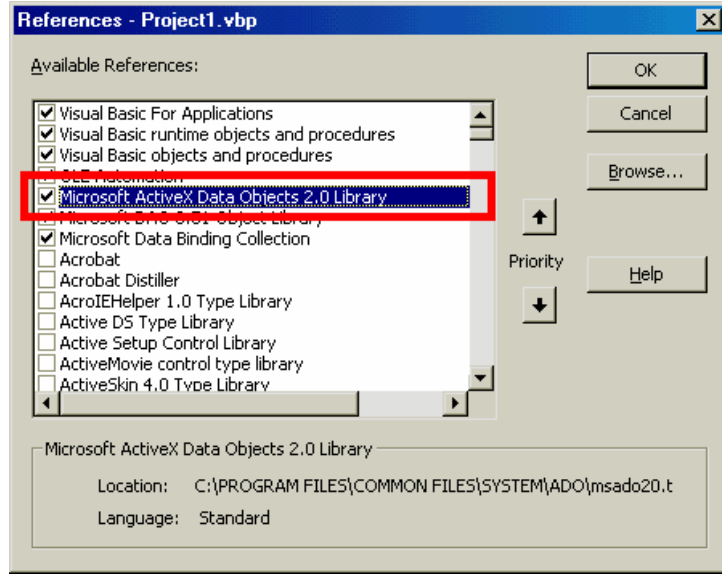
Microsoft Visual Basic 6.0

sistemi, grafik ve text gibi veriler.) OLE DB bir veri saglayicisidir. OLE DB araciligi ile uygulanan arabirimlere , program parçalarına Consumer denir.

ADO nun projede kullanilmasi için ilkönce referansin projeye dahil edilmesi gerekir.Bunun için **Project** menüsünden **References** () seçenegi seçilir. **Microsoft ActiveX Data Objects 2.0 Library** () seçilir. Ve islem onaylanır.



(Resim 33)



(Resim 34)

ADO Nesneleri

ADO yapı itibari ile oldukça basit bir nesne modeline sahiptir.ADO modelinde birçok nesne seçimli olarak veya bagimsiz olarak kullanılabilir.

ADO Modelinde Yeralan Nesneler;

Nesne	Amaci
Connection	Veri kaynagina dogrudan baglanti olusturur.
Command	Veri kaynagi üzerinde query (Sorgu) isletir.
Field	Kayit içindeki alan. (kolon)
Error	Veri kaynagindaki hatayi barindirir.
Parameter	Bir Komut için parametre barindirir.
Recordset	Bir komut tarafından üretilen (dönen) verileri barindirir.

Örneğin bir Access veritabanına bağlantı için şu nesneler kullanılmalıdır.

Veri: veritabani.mdb
 Connection : veritabani.mdb
 Command : SELECT * FROM telefon
 Recordset : Elde edilen bilgi.

Aşağıdaki örnekte sadece Recordset nesnesi kullanılmış ve bir Microsoft Access veritabanına erişilmiştir.

```
Dim kayıt As New ADODB.Recordset
kayıt.open "SELECT * FROM telefon","DSN=ORNEK", adOpenKeyset
Set DataGrid1.DataSource = kayıt
```

Connection Nesnesi

Connection nesnesi bir veri kaynağına bağlantı yapmak için kullanılır. Connection nesnesi sayesinde bir veri sağlayıcı aracılığı ile bir komut iletilir. Command nesnesi oluşturmaya gerek kalmaz. Connection nesnesi ile **ConnectionString** (Bağlantı cümlesi) özelliği kullanılır. Bu özellik sayesinde bağlanılacak veri kaynağı belirlenir. Ve **Open** methodu ile de bağlantı açılır

İstemci-sunucu veritabanlı sistemlerde connection nesnesi sunucuya bir ağ bağlantısını oluşturur. Önceden tanımlanmış nesnelerden bağımsız olarak istenilen şekilde yaratılabilir.

Temel Özellikleri ve Methodları;

Connection Nesnesinin temel özellikleri;

Özellik	Açıklama	Varsayılan	Değer Aralığı
CommandTimeout	Bir komutun işleme zamanıdır	30	0-Mx Ulong
ConnectionString	Veri kaynağını belirtir.	-	String değer
ConnectionTimeout	Bağlantı için bekleme zamanı	15	>=1
CursorLocation	Bir client yada server erişim biçimi		
Errors	Olusan hatayı tutar.	N/A	N/A
Mode	Bağlantının erişim izinleri.	Unknown	

ConnectionString, **Mode** ve **ConnectionTimeout** özellikleri ile bağlantının ne kadar süre içinde olmazsa hata verileceği, bağlantının hangi provider ile sağlanacağı ve bağlantı sonucundaki uyulması gereken izinler belirlenir.

DefaultDatabase (Varsayılan veritabanı) özelliği, bağlantı için önceden varsayılan olarak ayrılmış olan veritabanının imini belirtir.

Provider özelliğide bağlantı yapılacak olan veritabanının veri sağlayıcısı türünü belirtir. (Örn ODBC Provider)

Cursor Location Özelliği

Cursor veri sağlayıcısı ile erişimi düzenleyerek verilere ulaşım biçimini değiştiren ve istekte bulunan uygulamaya verileri döndüren ufak bir uygulamadır. Cursor veri seti konumunu izleyerek birçok işlemin aynı anda yapılmasını sağlar.

Cursor Location Değerleri;

adUseClient	Yerel cursor kütüphanesi tarafından sağlanır.
adUseServer	Varsayım olarak kullanılmaktadır. Veri sağlayıcısı veya ODBC tarafından varsayılan olarak tanımlı cursoru kullanır.

Cursor Location özelliğinin değiştirilmesi düzenleme işleminden sonra varolan bağlantılarıda etkilemektedir.

CursorLocation Özelliğinin Değerleri;

adUseServer	2	Server-side yada driver-supplied cursorlar kullanılır
AdUseClient	3	Client – side cursorlar kullanılır.

Eğer CursorLocation bir Recordset için belirtilmiş ise bir bağlantı ile birçok recordset kullanılabilir.

Mode Özelliği

Bağlantı ile elde edilen veriler üzerindeki erişim izinlerini belirtir.

adModeUnknown	0	Varsayım,İzin düzenlenmemiş
adModeRead	1	Read-Only izin.
adModeWrite	2	Write-Only izin.
adModeReadWrite	3	Read/Write izin.
adModeShareDenyRead	4	Diğer kullanıcıların bağlantıyı read iznini engeller.
adModeShareDenyWrite	8	Diğer kullanıcıların bağlantıyı write iznini engeller.
adModeShareDenyExclusive	12	Diğer kullanıcıların bağlantıyı açmasını engeller.
adModeShareDenyNone	16	Diğer kullanıcıların bağlantıyı herhangi bir izinle açmasını engeller.

Mode özelliği Connection nesnesi kapalıyken kullanılır, mevcut izinleri düzenler.

Connection Metotları

Connection Metotları, Connection nesnesi ile yapılabilecek işlemlerdir. Open ve Close metotları ile de veri sağlayıcısına bağlantı kurulur veya bağlantı kesilir. Open metodunun işletilmesi ile bağlantı başlar. Komutlar işletilir ve sonuçlar alınır.

BeginTrans	Aktif Connection üzerinde yeni bir transaction'a başlar.
Cancel	İşletilmekte olan komutu iptal eder.
Close	Aktif bağlantı kapatır.
CommitTrans	Mevcut Transaction işl. Tamamlar.
Execute	Bağlantıya ait komutu işler.
Open	Bir bağlantı açar.
OpenShema	Bir Recordset'in schema bilgisini verir.
RollbackTrans	Mevcut transaction'u geri alır.

Open methodu ile aktif bağlantı yaratılır ve sistem kaynakları kullanılmaya başlanır, Close methodu ile aktif bağlantı kapatılır ve sistem kaynakları serbest bırakılır.

Aşağıdaki örnekte Access veritabanına DSN aracılığı ile bağlantı kurulur.

```
Dim bagl As New ADODB.Connection
bagl.open "telefon"
```

Execute methodu ise aktif bağlantı üzerinde bir komut çalıştırır. Bu komut genelde SQL Update komutu gibi bir satır olarak veri elde etmeyen bir komuttur.

Set recordset = connection.Execute (CommandText,RecordAffected,Options)

CommandText bilgisi ise string bir değerdir. Bir tablo adı, SQL sorgusu, stored procedure yada text bir bilgi içerir. RecordsAffected bilgisi seçimli olarak kullanılır. Bu bilgi işlemde etkilenen kayıt sırasını döndürür. Options bilgisi de seçimlidir. Provider'in CommandText'i nasıl işleyeceğini belirtir.

CommandText Bilgisinin Değerleri-Açıklamaları;

AdCmdText	Komut metninin text olarak değerlendirilmesini sağlar.
AdCmdTable	Tablodaki tüm kayıtlar için bir SQL sorgusu kullanılması.
AdCmdTableDirect	Veri sağlayıcısının tablodaki tüm kayıtları döndürmesi.
AdCmdTable	Komut metninin bir tablo adı olarak değerlendirilmesi

AdCmdStoredProc	Komut metninin bir Stored Proc. Olarak degerlendirilmesi.
AdCmdUnknown	Komut tipinin argumaninin bilinmemesi.
AdExecuteAsync	Komutun zaman uyumsuz olarak isletilmesi.
AdFetchAsync	CacheSize özelligi ile belirtilen miktarin disinda kalan satirlari belirtir.

Örneğin bir komutu çalıştırmak için komutu ,

```
Set kayıt = bagl.Execute("SELECT * FROM telefon")
```

Seklinde kullanabiliriz.

Bir SQL Server Veritabanına Bağlantı Kurulması

```
Dim baglanti as New ADODB.Connection  
Baglanti.open "driver={SQL  
Server};Server=<Servername>;uid=<uid>;pwd=<sifre>;database=<databa  
sename>"
```

Access Veritabanına Bağlantı Kurulması

Asagidaki örnekte "JOLT OLE DB" sağlayicisi kullanılarak Access veritabanına bağlantı yapılmamıştır.

```
Dim baglanti as New ADODB.Connection  
Baglanti.ConnectionString = "c:\veritabani.mdb"  
Baglanti.Provider = "Microsoft.Jet.OLEDB.3.51"  
Baglanti.open
```

DSN Kullanarak Bağlantı Kurmak

Baglantı kurmanın en kolay yolu DSN kullanmaktadır, az önce yapılan ince ayarlamaları- detayların yapılmasına gerek kalmaz. O detayları DSN otomatik olarak işleme koyar.(Asagidaki örnekte ORNEK isimli DSN' e bağlanılmıştır.)

```
Dim baglanti As New ADODB.Connection  
Baglanti.open "ORNEK"
```


Error Nesnesi

Error Object (Hata nesnesi), içinde olusan hatalarla ilgili bilgi bulunduran bir degerdir.

Error nesnesi içindeki özellikler vasitasi ile hatalar kolaylikla tespit edilebilir

Description	Hata alarm Metnidir.
Number	Hata numarasidir.
Source (kaynak)	Hataya neden olan nesnedir.
Helpfile,HelpContext	Hata için uyuhun help dosyalaridir.
SQLState,NativeError	ODBC verikaynaklari hk. Bilgi içerir.

Command Nesnesi

Command nesnesi bir baglanti üzerinde isletilebilen sorgulari, Stored Procedureleri çalıştırmada kullanılır.Isletilmeden önce aktif olan baglanti ile iliskilendirilmelidir. Genelde Recordset nesnesinin yaratilmasi ve kayitlarin elde edilmesi için kullanılır.

Command Nesnesinin Özellikleri

Özellik	Açıklama	Varsayım	Deger Araligi
Active Connection	İlgili Baglantıyı geri döndürür	Yok	Baglanti nesnesi
Command Text	Komut Nesnesi	Yok	Deyim
Command Timeout	İsletilecen komut zaman asimi süresi	30	0-Ulong (Maks)
Command Type	İsletilecek komut tipi	AdCmdNothing	Enum
Name	Baglanti tipinin ismi	Yok	Deyim
Parameters	Parameter nesnesinin bileşimi	Yok	N/A
Prepared	Komutun derlenmiş olması	False	True/False
Properties	Özellikler Birleşimi	N/A	N/A

Active Connection özelliği ile açık bir bağlantıyla bir komut nesnesi birleştirilebilir. Command Text özelliği, komutun metinsel tanımını içerir. Bu metinsel tanımın içeriği kullanıcıya özeldir fakat veri sağlayıcısının destekleyebildiği bir komut olmalıdır. Bu özellik Command nesnesine bağlandığında nesne açık bir bağlantıya ilistilendirilmiş ise, Execute ve Open metodları çağırıldığı zaman sorgu ADO tarafından hazırlanır. Command Type özelliğinin değerine göre ADO tarafından CommandText özelliği değiştirilebilir.

Örnek kullanım Command nesnesi herhangi bir kayıtsesi göndermek zorunda değildir. Örnek olarak verilen kodlarda UPDATE işlemi yapılmaktadır. (Command nesnesi komut olarak adlandırılmıştır)

```
Dim baglanti As New ADODB.Connection
baglanti.CursorLocation = adUseClient
baglanti.ConnectionString = "C:\veritabani.mdb"
baglanti.Provider = "Microsoft.Jet.OLEDB.3.51"
baglanti.Open

sql="UPDATE borc set borc = borc * 1.1"
Set komut.ActiveConnection =baglanti
komut.CommandText = sql
komut.Execute intRecordsAffected
MsgBox Str(intRecordsAffected) & "degistirildi."
```

CommandTimeout özelliği ile bir komutun isletilmesi için geçecek süre saniye cinsinden belirtilebilir. Eger süre asılırsa hata oluşur. **CommandType** özelliği ise performansı artırmak için komut çalıştırılmadan önce komut tipini belirtir. Komut yaratıldığında komut işleyişini denetlemek için belirli özellikler düzenlenir. Aşağıda verilen örnekte bir komuta CommandText, CommandType ve CommandTimeout özellikleri düzenlenmektedir.

```
komut.CommandType = adCmdStoredProc
komut.CommandTimeout = 30
komut.CommandText = "UPDATEPROCEDURE"
Set komut.ActiveConnection = baglanti
komut.Execute intRecordsAffected
```

Execute methodu kullanılarak komut iletilir ve recordset oluşur, bu metod komut nesnesi üzerinde islenir, **CommandText** özelliğinde belirtilen sorguyu çalıştırır.

```
Set kayit.activeconnection = cnn
Baglanti.CommandText sql
Execute intRecordsAffect
```

Execute metodu ile recordset olusturulabilir.

```
Dim kayit As New ADODB.Recordset
Dim baglanti As New ADODB.Connection
Baglanti.Cursorlocation = adUseClient
Baglanti.ConnectionString = "c:\veritabani.mdb"
Baglanti.Provider = "Microsoft.Jet.OLEDB.3.51"
Baglanti.Open

Set kayit = baglanti.Execute("Select * From telefon")
Set Datagrid1.Datasource = kayit
```

CommandType Özelliğinin Değerleri

AdCmdText	1	CommandText değerini bir metin olarak girilmesi sağlanır.
AdCmdTable	2	CommandText'in bir tablo adı içermesi sağlanır.
AdCmdStoredProc	4	CommandText'in bir Stored Proc içermesi sağlanır
AdCmdUnknown	8	CommandType bilinmiyor.
AdCmdFile	256	Verilerin dosyadan alınmasını sağlar.

Varsayılan CommandType özelliği adCmdUnknown dur. Birçok durumda komutun tipini tanımlamak yeterli olur.

```
Komut.CommandText = "SELECT * FROM telefon"
```

Veya ;

```
Komut.CommandText ="telefon"
```

Yukarıdaki iki deyim de aynı sonucu üretmektedir.

Name Özelliği

Name Özelliği bir command işletildiğinde dinamik bir bağlantının oluşmasını sağlar.

State Özelliği

Mevcut nesnenin o andaki durumunu gösterir.

State özelliğinin değerleri;

adStateClosed	0	Varsayım. Command nesnesi kapalı.
---------------	---	-----------------------------------

adStateOpen	1	Command nesnesi açık.
adStateConnecting	2	Command nesnesi bağlanıyor.
AsStateExecuting	4	Command nesnesi bir komut işletiyor.

Command Nesnesinin Metodları

Cancel	Bir command'in işletimini iptal eder.
CreateParameter	Bir parametre nesnesi yaratır.
Execute	Bir komutu ya da query'yi işletir.

Parameter Nesnesi

Parameter nesnesi Command nesnesi ile birlikte parametreleri-argümanları temsil eder. Eger çağrılmak istenen sorgu veya stored-procedure ise bu sorgu ve stored-procedure'lerin parametreleri biliniyorsa **CreateParameter** metodu ile parametreler yaratılır.

Parameter nesnesinin özellikleri;

Name :Name özelliği ile parametrelerin ismi elde edilebilir.

Value :Value özelliği ile parametrenin değeri öğrenilebilir.

Attributes, Precision, Direction, Size, Type gibi özellikler ile nesnenin diğer özellikleri düzenlenir.

Recordset Nesnesi

En çok kullanılan nesne Recordset nesnesidir. Recordset nesnesi veri kaynagından gelen verileri barındırır. Veri kaynagina gönderilen sorgunun sonucu veriler burada tutulur. Recordset genelde aktif bağlantı nesneleri ile veya command nesnesi ile kullanılır. Bagimsiz olarak ta kullanılabilir. Istendigi kadar Recordset nesnesi olusturulabilir. Bir recordset nesnesi olusturulacagi zaman dört degisik cursor tipinden biri kullanılabilir.

- Dynamic Cursor, diğer kullanıcıların yaptıkları işlemlerin görülmesini sağlar, bütün işlemlere izin verir.
- Keyset Cursor, diğer istemcilerin eklediği kayıtların görülmesini engeller. Kullanıcı hertürlü izine sahiptir. Diğer istemcilerin yaptığı veri değişiklikleri görülebilir.

- Static Cursor, Belli bir veri bulunması ya da raporlama işlemi için süzülen verilerin değişmeyen kopyasını yaratır. Kullanıcı veriler üzerinde hertürlü izine sahiptir. Diğer istemcilerin yaptıkları eklemeler, değişiklikler ve silmeler görülmezler.
- Forward-Only-Cursor sabit göstericiye benzer bir çalışma sekline sahiptir. Fakat sadece ileriye doğru harekete izin verir.

Recordset aktif hale gelmeden önce onun CursorType özelliği ayarlanır. Eğer hiçbir ayarlama yapılmazsa ForwardOnly varsayım olarak kullanılır. Recordset nesnesi oluşturulduğu zaman o andaki ilk kaydı gösterecek şekilde ayarlanır. Bu anda **BOF** (Dosya Basi) ve **EOF** (Dosya Sonu) özellikleri false değerine sahiptir.

BOF özelliği, o andaki kaydın recordset nesnesi içindeki ilk kayıttan önceki kayıt olduğunu, EOF özelliği ise o anki kaydın son kayıttan sonraki kayıt olduğunu belirtir. BOF ve EOF özellikleri, recordset içinde kayıt olup olmadığına ya da kayıtlar arasında gezinirken oluşturulan recordset nesnesinin sınırlarının dışına çıkılıp çıkmadığına karar vermek için kullanılır.

BOF
Kayıt 1
Kayıt 2
Kayıt 3
EOF

RecordCount özelliği ise recordset nesnesinin barındırdığı kayıt sayısını verir. Eğer ADO kayıt sayısını belirleyemez ise değer -1 olarak geri döner.

Tüm kayıtlar üzerinde işlem yapılabilmek için aşağıdaki teknik kullanılabilir.

```
Kayit.movefirst
For j=1 to kayit.recordcount
-   Islemler           -
-   Islemler           -
-   Islemler           -
kayit.movenext
```

Next j

Örnek: Recordset tanımlaması;

```
Dim kayıt As New ADODV.Recordset
Kayit.open           "SELECT           *           FROM
TELEFON" , "DSN="ORNEK" ,adOpenKeyset , adLockOptimistic
```

Örnek: Recordset içeriğinin datagrid üzerinde gösterilmesi;

```
Dim baglanti As New ADODB.Connection
Dim komut As New ADODB.Command
Dim kayıt As New ADODB.Recordset
    baglanti.CursorLocation=adUseServer
    baglanti.Open "ORNEK"

    komut.CommandText="SELECT * FROM telefon"
    Set komut.ActiveConnection = baglanti
    Kayit.CursorType=adOpenDynamic
    Kayit.LockType=adLockOptimistic
    Kayit.Open
    Kayit.open komut
Set Datagrid1.DataSource = kayıt
```

Command ve Connection nesnelerini kullanmadan doğrudan recordset vasıtası ile verilere erismek daha kolay ve zahmetsiz bir işlemdir. Ancak bir komut işlendiğinde recordset yaratılmamışsa dönen kayıtlar recordse ve default içinde yer alır. Ancak default Recordset'in kullanımı için LockType ve CursorType özelliklerinin düzenlenmesi gerekmektedir.

```
Dim baglanti As New ADODB.Connection
Dim komut As New ADODB.Command
Dim def As New ADODB.Recordset
    baglanti.CursorLocation = adUseClient
    baglanti.Open "ORNEK"
Set def =baglanti.Execute("SELECT * FROM telefon")
```

Recordset oluşturulduğunda herhangi bir komut kullanılmadığında bos olur. Böylece Recordset nesnesi istendiğinde doldurulabilir. Bir diğer olanak ise recordset nesnesinin istendiği zaman kapatılabilmesi veya açılabilmesidir.

Örnek: Recordset nesnesi kapatılıyor ve tekrar açılıyor;

```
Dim kayıt As New ADODB.Recordset
Kayıt.close
Kayıt.open "SELECT tel, ad, FROM telefon"
```

Recordset Nesnesinin çalışma sırasında kullanılan özellikleri;

Ozellik	Açıklama	Varsayım	Aralık
AbsolutePage	Kayıtseti içindeki mutlak sayfayı döndürür.	-1	>=0
AbsolutePosition	Aktif kaydın mutlak komutunu döndürür.	-1	>=0
ActiveCommand	Aktif komutu verir.	Yok	Komut nesnesi
ActiveConnection	Aktif bağlantıyı verir.	Yok	Bağlantı nes. Degeri.
BOF	Kayıt gösterici ilk kayıttan önceye geçmesi.	True	True-false
Bookmark	Mevcut kaydı belirleme bilgisi.	Saglayiciya bagli	Saglayiciya bagli
CacheSize	Önbellege atılabilecek kayıt sayısı değeri.	1 sadece ileri 10 tüm gösterciler	>=-1
CursorLocation	Cursorun yerini belirler.	Saglayiciya bagli	Enum
CursorType	Cursor tipi.	AdOpen/Forward only	Enum
EditMode	Mevcut kayıt degistirme modu.	AdEditNone	Enum
EOF	Kayıt göstericinin son kayıttan önce olması	True	True/False
Fields	Veri alanı (tablonun içindeki alanlar)	N/A	N/A
Filter	Filtre değeri.	AdFilterNone	Enum
LockType	Kilitleme değeri.	AdLockRead	Enum
MaxRecords	Alınacak max kayıt sayısı.	0	>=0
PageCount	Sayfa sayısı.	0	>=0
PageSize	Bir sayfadaki kayıt sayısı.	0	>=0
RecordCount	Kayıt sayısı.	1	>=0
Sort	Sıralama düzeni.	-Bos-	Deyim
Source	Komutun kaynağı.	-Bos-	Deyim/Nesne
State	Recordset'in mevcut durumu.	AdStateClosed	Enum
Status	Kodların güncelleme durumu.	N/A	Enum

Recordset nesnesinin özelliklerinin belirtilmesinden önce recordset nesnesinin yaratılması gerekir. Recordset bir connection veya komut ile ilişkilendirilir. Ardından erişim, kilitleme gibi düzenlemeler yapılır.

Örnek Recordset nesnesi oluturulması ve verilere erişim;

```
Dim baglanti As New ADODB.Connection
Dim komut As New ADODB.Command
Dim kayıt As New ADODB.Recordset
Baglanti.CursorLocation=adUseServer
Baglanti.Open "ORNEK"

With komut
    .CommandText = "SELECT * FROM telefon"
    Set .ActiveConnection = baglanti
End With

With kayıt
    .CursorType = adOpenDynamic
    .LockType = adLockOptimistic
    .Open komut
End With

Set Datagrid1.DataSource = kayıt
```

Cursor Type Özelliği

CursorType özelliği recordset içinde hareketi kontrol eder.Olası işaretçi tipleri veri sağlayıcısı'nın desteğiyle mümkündür. Veri sağlayıcısı tarafından kullanılmayan bir işaretçi kullanıldığında ADO varsayım olarak en yakını kullanır.

Sabit	Deger	Açıklama
AdOpenForwardOnly	0	Sadece ileri
AdOpenKeyset	1	Güncel veri içerir.Geriye doğru hareket edilir.
AdOpenDynamic	2	Güncel veri içerir.Geriye doğru hareket edilebilir, Bookmark kullanılabilir.
AdOpenStatic	3	Veri Sabittir. Hareket geriye doğrudur.

LockType Özelliği

LockType özelliği bir recordseti kilitlemek için kullanılır.

adLockReadOnly	1	Veriler sadece okunur sekildedir degisemez.
adLockPessimistic	2	Uzerinde islem yapılan veriler satir bazinda kilitlenmektedir.Baskalarin erisimi engellenir.
adLockOptimistic	3	Uzerinde islem yapılan veriler kilitlenmez.Baskalarinin erisimi kontrol edilebilir

adLockOptimistic Batch	4	Degisiklikler toplu yapilir
------------------------	---	-----------------------------

Not: Pessimistic locking islemi Client-side 'ler tarafından kullanılmaz.

Bookmark Özelliği

Bookmark özelliği sayesinde bir recordset içindeki bir kayitin konumu belirlenir, kayit edilir. Daha sonra bu konuma geri dönebilir. Recordset açildiği anda her kayita tek bir bookmark atanir.

Örnek: Bulunan kayitin konumu saklanır.

```
Dim im as Variant
Private Sub bookmarkla_Click()
    Im=kayit.bookmark
End Sub

Private Sub Geridon_Click()
    Kayit.bookmark = im
End Sub
```

Not : Bazi cursor tipleri bookmark kullanimini desteklemez (Forward-only, Dynamic)

Eger kullanılan cursorun bookmark'i destekleyip desteklemediği öğrenilmek isteniyorsa aşağıdaki kodlar kullanılabilir.

```
Destek=kayit.Supports(adBookmark)  '(deger true veya
false olarak geri döner.)
Msgbox destek
```

Eof Özelliği

Kayit setinin sonunu gösterir. EOF özelliği kullanıcının en son kayidi geçmeye çalışmasında true degerini verir.

```
While not kayit.EOF
    'Kayit üzerindeki işlemler
    kayit.movenext
Wend
```

BOF Özelliği

Kayıt setinin basını isaret eder. Eğer isaretçi ilk kayıttan bir üste geçerse değeri true olur eğer recordset nesnesi boş ise EOF ve BOF true değerini alır.

Recordset Nesnesinin Metodları;

Recordset nesnesi üzerindeki çoğu işlem aşağıdaki metodlarla yapılabilir.

AddNew	Kayıt ekler.
Cancel	Yürürlükteki işlemi iptal eder.
CancelBatch	İslenecek kayıt kümesini iptal eder.
CancelUpdate	Mevcut kayıta yapılacak işlemleri iptal eder. (ekleme, degistirme)
Clone	Recordseti kopyalar.
Close	Recordseti kapatır.
Delete	Kayıt silmede kullanılır.
Find	Bir kaydı bulur.
GetRows	Bir kayıt kümesini iki boyutlu diziye kopyalar.
GetString	Kayıtları Text string olarak geri döndürür.
Move	Belirli bir kayıda gitme işlemi için kullanılır.
MoveNext	Bir sonraki kayıta gidilir.
MovePrevious	Bir önceki kayıta gidilir.
MoveFirst	İlk kayıta gidilir.
MoveLast	Son kayıta gidilir.
NextRecordset	Birleşik bir komut içindeki bir sonraki recordseti açar.
Open	Bir komutu isletir ve kursoru açar.
Requery	Bir Recordseti yeniden oluşturur, komutu yeniden isletir.
Resync	Önbelleğe alınan kayıtları yineler.
Save	Mevcut bir recordseti bir dosyaya kayıt eder. Bu dosya sonradan açılabilir.
Supports	Veri sağlayıcısı tarafından desteklenen cursor seçeneklerini belirtir.
Update	Yapılan değişiklikleri recordsete kaydeder.
UpdateBatch	Kaynak üzerindeki bir grup güncelleme işlemini yapar.

Örneğin aşağıdaki kodlar ile kayıt isimli recordset nesnesinde bir sonraki kayıda erişilebilir.

```
Private Sub sonraki_Click()  
    'Sonraki kayıta geçiliyor  
    Kayit.MoveNext  
    'Eof kontrolü yapılıyor  
    if kayit.EOF Then  
        beep  
        msgbox "Bu en son kayıttır"  
        kayit.MoveLast
```

```
End If  
End Sub
```

Önceki kayıta ulaşmak için ise aşağıdaki komutlar kullanılabilir.

```
Private Sub onceki_Click()  
    'Sonraki kayıta geçiliyor  
    Kayit.MovePrevious  
    'BOF kontrolü yapılıyor  
    If Kayit.BOF Then  
        beep  
        kayit.MoveNext  
    End If  
End Sub
```

Aşağıdaki örnekte ise Open methodu farklı bir şekilde kullanılmıştır.(SQL Sorgusu ile kayıtlar, SQL server'dan istenmiştir, çıktılar ise DBGrid'e yönlendirilmiştir.)

```
Dim kayit As New ADODB.Recordset  
Kayit.open "SELECT * FROM telefon", "driver={SQL  
Server};Server=BIM;uid=sa;pwd=;database=telefon", adOpen  
Keyset, adLockOptimistic  
Set DataGrid1.DataSource = kayit
```

MoveNext metodu bir sonraki kayıta, MovePrevious metodu bir önceki kayıta MoveLast metodu enson kayıta MoveFirst metodu ise ilk kayıta erismeyi sağlar.

Recordset üzerindeki yapılan işlemler (kayıt işlemi, silme işlemi, düzenleme işlemi) sonucunda recordset güncellenmelidir. Güncelleme işlemi Update İşlemi iki farklı şekilde yapılır.Anında (immediately) ve küme halinde (batched) gibi. Anında yapılan güncellemede yazılan veriler Update metodu çağrıldığında veriler üzerinde yapılan tüm değişiklikler anında uygulanır.(Örn Addnew ve Update)

Eğer veri sağlayıcı küme halindeki güncellemeyi destekliyorsa, birden fazla değişiklik yapıldıysa da bu değişiklikler önce bir cache bellekte

saklanır ve Sonra UpdateBatchmetodu ile yapılır.Güncelleme işlemi esnasında recordsetin status özelliği sayesinde durum öğrenilebilir.

Recordset ile Verilere Erişmek ve Görmek

Kayıtlara, verilere erişmek için recordsetin metodları kullanılır. Aşağıdaki örnekte ise SQL server üzerinde bir recordset nesnesinin Connection nesnesi ile oluşturulması ve doldurulması konu alınmıştır.

```
Dim kayit As New ADODB.Recordset
Dim baglanti As New ADODB.Connection
Baglanti.open "driver={SQL
Server};Server=BIM;uid=sa;pwd=;database=telefon"
Baglanti.open "SELECT * FROM telefon", baglanti,
adOpenKeyset, adLockOptimistic
Set DataGrid1.DataSource = kayit
```

Aşağıdaki örnekte Microsoft Access üzerinde; bir Recordset nesnesinin Connection nesnesi ile yaratılması ve doldurulması örneklenmektedir.

```
Dim kayit As New ADODB.Recordset
Dim baglanti As New ADODB.Connection
Baglanti.open "ORNEK" 'DSN ismidir
kayit.open "SELECT * FROM telefon", baglanti,
adOpenKeyset, adLockOptimistic
Set DataGrid1.DataSource= kayit
```

Recordset nesnesi kayıtlar üzerinde arama ve sıralama yapabilmek için özel methodlara sahiptir.Find methodu ile belirli bir kayıt bulunur, Sort methodu ile kayıtlar artan veya azalan değerlere göre sıralanırlar.Filter methodu ise belli bir değer içeren kayıtları bulmayı sağlar.

Sort

Sort özelliği recordset içindeki verilerin sıralanma düzenini belirler.Sıralama belirli bir alana göre yapılır. Sıralama ASC(artan) veya DESC (azalan) şeklinde yapılabilir.Eğer bir değer verilmez ise o zaman sort işlemi ASC olarak yapılır.

Örnek:Sıralama

```
kayit.Sort = "BORCU DESC,[ADI, SOYADI] ASC"
```

Recordseti tekrar eski haline getirmek için ise

```
kayit.Sort=""
```

Seklinde bir kod kullanılabilir.

Sıralama işlemi sonrasında verilerin tekrar kullanıcı bilgilendirme amaçlı kullanılan bileşenlere tekrar gönderilip güncelleme işleminin yapılması gerekir.

```
kayit.Sort=""  
Set Datagrid1.DataSource = kayit
```

Filter

Filter özelliği, recordset içindeki verileri bir kritere göre süzgeçten geçirilmesini sağlar. Mesela aranan kriter "ALI" olsun , kayıtlardan istenen alan "ALI" sözcüğüne göre filtrelenir ve "ALI" sözcüğüne sahip olan kayıtlar listelenir.

Örnek:

```
kayit.filter="ad='ALI' "
```

Filter methodundan sonra tüm kayıtları göstermek için "adFilterNone" kullanılabilir.

```
kayit.Filter = adFilterNone
```

Ayrıca filtreleme işleminde Or ve And kullanılabilir.

```
kayit.filter ="(ad='Ali' or ad='Veli')"
```

Find Methodu

Find methodu bir recordset üzerinde bir kolon içinde bir değer arar.

```
kayit.Find("ad='Nedim'")
```

Eğer aranacak kriter bir textbox ise aşağıdaki gibi bir yöntem kullanabiliriz.

```
kayit.Find "ad like ' " & text2.text& " ' "
```

DATA Environment

- **DATA Environment Designer'in Kullanilmasi**
- **Command ve Connection Nesneleri**
- **Komutların İşletilmesi**
- **Sorgulama İşlemleri**

Data Environment

Data Environment, database uygulamalarını geliştirmek için kullanılan bir yapıdır. Grafik ortamı ve programcıya sağladığı kolaylıklar sayesinde kolayca az kod kullanılarak database uygulamaları geliştirmeye olanak sağlar.

Yapı itibarı ile bir ADO Recordseti içerir. Tüm recordset özellikleri Data Environment vasıtasıyla kullanılır.

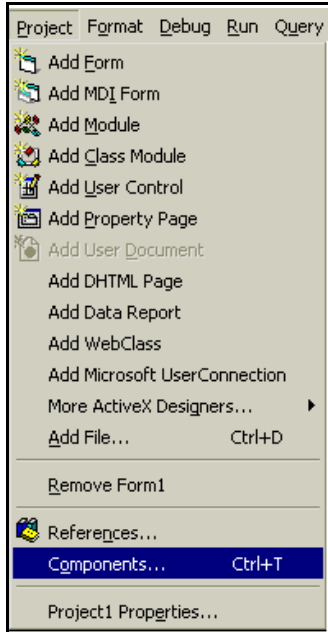
Data Environment Designer ise ActiveX designer'dır. Data Environment Designer sayesinde ADO Connection'ları ve Command'ları yaratılır. Nesneler bağlanır ve proje çalıştığında çıktıları kullanılır.

Data Environment Designer

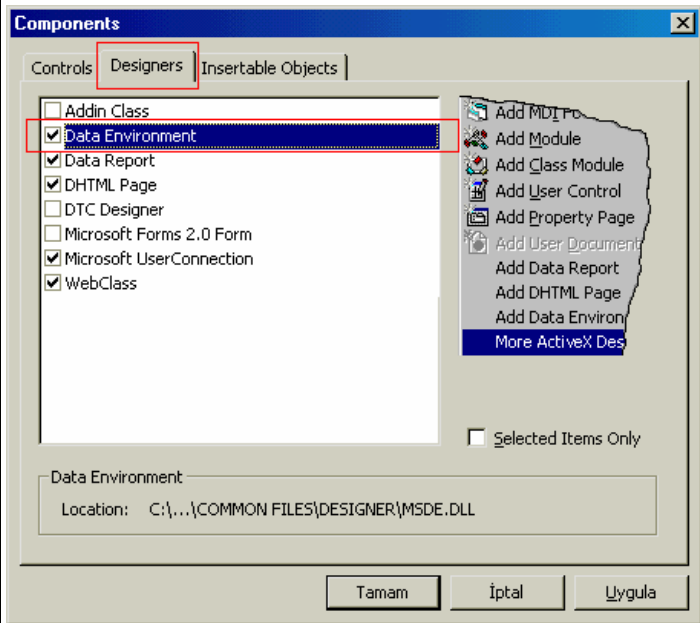
Data Environment Designer'in proje genelinde kullanılması için ilkönce projeye eklenmesi gerekir.

1- **Project** (bkz Resim1) menüsünden **Components** seçilir.

2- **Designers** sekmesi seçilir ve **Data Environment** seçeneği seçilir. (bkz Resim 2)



(Resim 1)

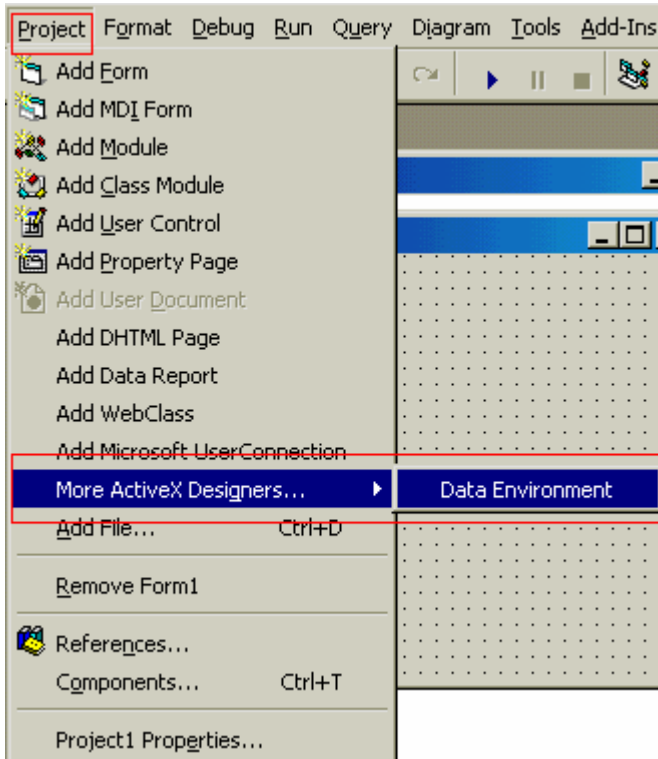


(Resim 2)

3- İşlem onaylanır ve **Data Environment** Projeye eklenmiştir.

Data Environment'i kullanmak için öncelikle proje bileşenlerine eklenmesi gerekir fakat bu işlemten sonra mevcut projede kullanabilmek için Data Environment Designer kullanılır, yani Data Environment Designer'in de projeye eklenmesi gerekmektedir.

Bunun için Project menüsünden **More ActiveX Designers...** seçeneği altındaki Data Environment seçeneğine tıklanılır ve Data Environment Designer ekrana gelir.(bkz Resim 3)



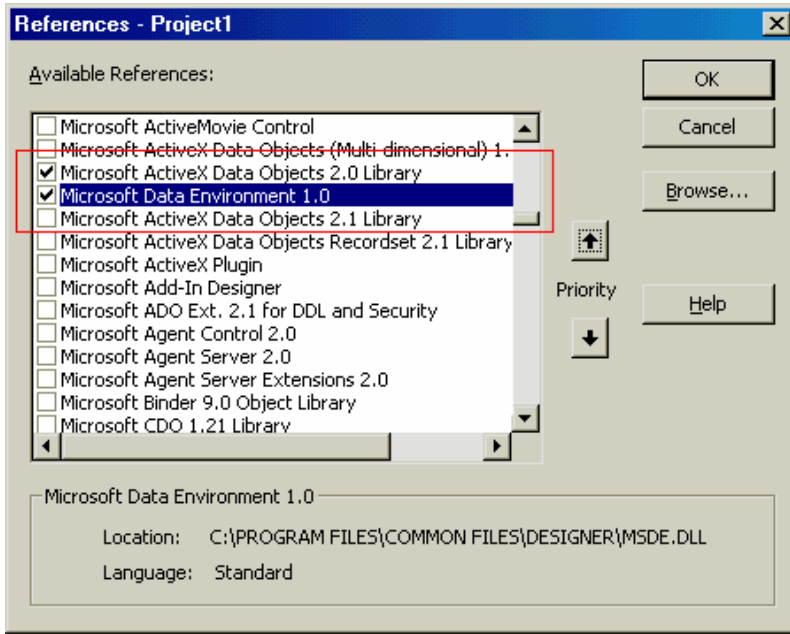
(Resim 3)

Bunun disinda Data Environment kullanimi için iki adet referansin da projeye dahil edilmesi gerekir.

1- Microsoft ActiveX Data Objects 2.0 Library.

2- Microsofy Data Environment Instance 1.0

(Not: Islem Project Menusunden References seçenegine tiklanilarak açilan diolog kutusundan yapilabilir)



(Bkz Resim 4)

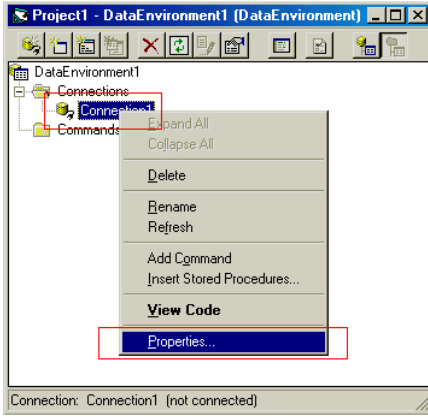
Command Ve Connection Nesneleri

Bir projede Data Environment kullaniliyorsa islemler bu D.E nesnesinin içinde barindirdigi Connection ve Command nesneleri ile yapılır.

Connection nesnesi herhangi bir providerin sagladigi veri kaynagidir,(varsayılan adi connection1 dir) Command ise o kaynak içerisindeki bir Stored Procedure , SQL sorgusu veya tablo olabilir.(varsayılan adi command1 dir)

Data Environment Designer açıldığında ayarlari yapılmamis Connection1 bağlantı nesnesini içerir.Ayarlari yapılip istenen veri saglayicısına bağlanılmak için varolan Data Environment Designer'ın içindeki varolan Connection1 nesnesinin üzerine sag tusla tıklayıp **Properties**(bkz Resim 5) seçenegine tıklamak yeterlidir. Açılan dialog penceresinde gerekli ayarlar yapılır.

Microsoft Visual Basic 6.0

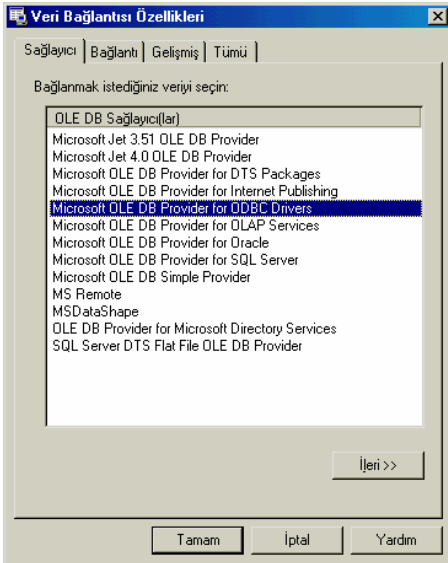


(Resim 5)

Örnek olarak Bir Data Environment- Connection nesnesi konfigure edilme İşlemi

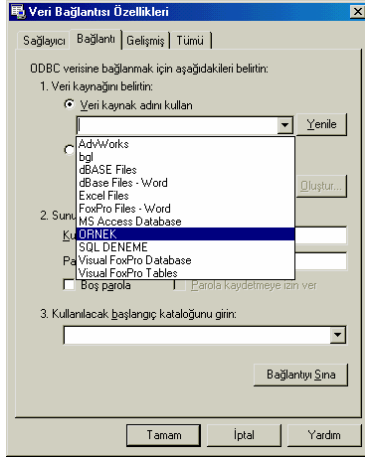
Yapılacak işlemler ADODC kontrolünün konfiurasyon işlemindeki işlemlere benzemektedir.Daha önce anlatıldığı gibi Connection nesnesine sağ tusta tiklanır ve **Properties** seçeneğine tiklanır.Açılan pencere aşağıdaki gibidir.

Bu pencereden kullanılacak olan Data Provider seçilir.(bkz Resim 6)



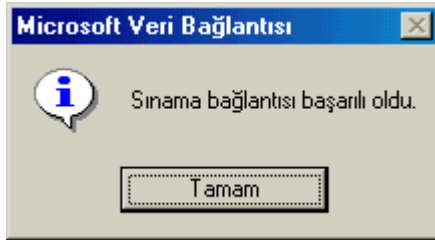
(Resim 6)

Eğer OLE DB Provider for ODBC Drivers seçeneği seçilir ve ileri (veya next) butonuna tiklanırsa işlemin bir sonraki adımına geçilir. Bu adımda veri kaynağı adı seçilir (ODBC DSN) (bkz Resim 7)



(Resim 7)

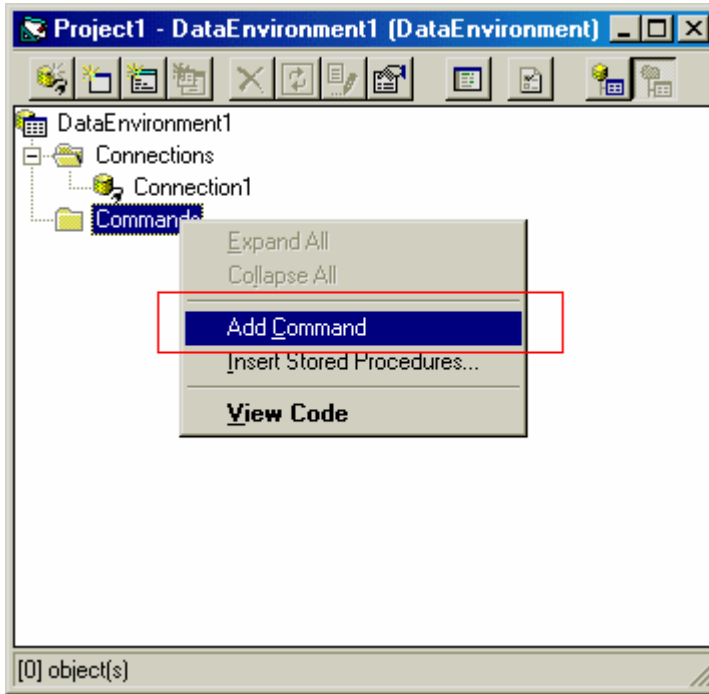
Bağlantıyı Sina (veya Test Connection) butonuna tiklanılarak bağlantıda sorun olup olmadığı kontrol edilebilir. (Resim 8)



(Resim 8)

Data Environment' te işlemler Connection'lar ve Command'lar ile yapılmaktadır. Yukarıda anlatılanlar sayesinde bir connection nesnesi bir veri sağlayıcısına bağlanır. Fakat işlem yapabilmek için bir de komut nesnesinin Data Environment Designer vasıtasıyla eklenmesi ve gerekli ayarlarının yapılması gerekir.

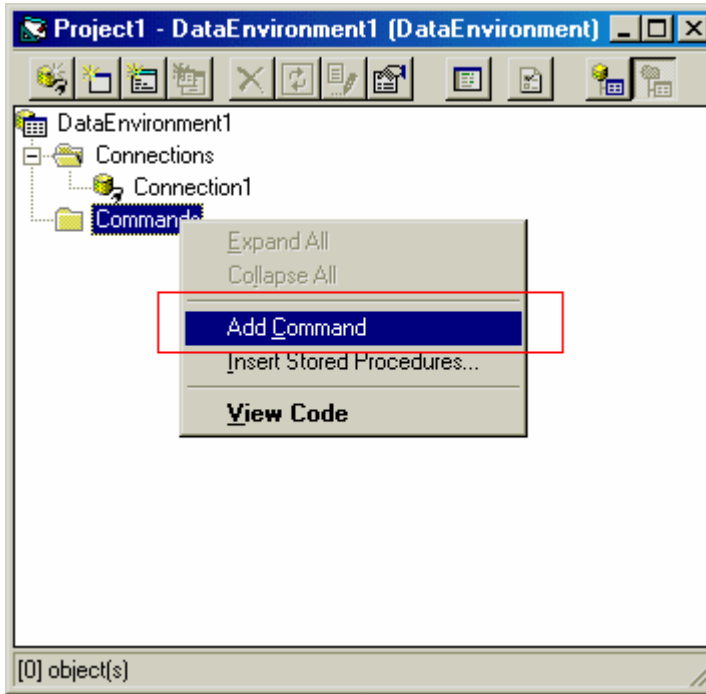
Bunun için de Data Environment Designer'da Connection nesnesinin altındaki kısımdaki **Command** seçeneğine sağ tusa tıklayıp **Add Command** seçeneğine tiklanması gerekir. (bkz Resim 10)



(Resim 10)

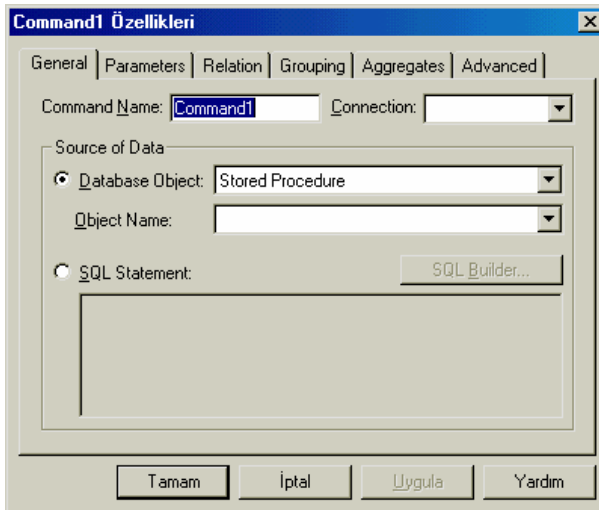
Bir önceki sayfada anlatılan işlemler yapıldıktan sonra Command1(komut nesnesinin varsayılan ismi Command1 dir denmisti) isminde yeni bir komut oluşturulur.Fakat komut nesnesinin de konfigure edilmesi gereklidir.

Bunun içinde Command1 isimli komut nesnesine sag tusla tiklanilarak açılan menüden **Properties** seçenigine tiklanmalıdır.(bkz Resim 11)



(Resim 11)

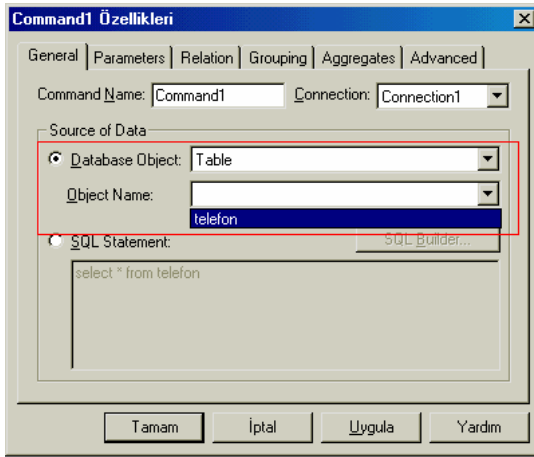
Yukarıdaki işlem adımları sonucunda ayarlamaların yapılacağı diyalog penceresi ekrana gelir.(bkz Resim 12)



(Resim 12)

Command1 Özellikleri (veya Command1 Properties) başlıklı pencerede command1 isimli komut nesnesinin tüm ayarları yapılabilir.

- Bu bileşenin çalışması için ilkönce bağlantı nesnesine bağlanması gereklidir. Bu işlem için **General** Sekmesindeki **Connection** liste kutusu kullanılır.(Connection liste kutusunda Data Environment Designer içindeki tüm bağlantı nesnelerini görmek mümkündür.)
- Sonra da Datasource object seçimi yapılır. Daha önceden de anlatıldığı gibi Database Object olarak Stored Procedure, SQL sorgusu veya Command1 nesnesinin bağlı bulunduğu veri kaynağının içindeki herhangi bir tablo olabilir.(bkz Resim 13)



(Resim 13)

Bu işlemler sonucunda Command1 nesnesine uygulanan her işlem direkt olarak bağlı bulunduğu bağlantının içindeki bağlı olduğu kaynağa uygulanır.

Ayrıca bir komut nesnesi oluşturulduğunda otomatik olarak recordset nesnesi yaratılır. Ve komut nesnesinin isminin önüne rs eki getirilerek bir varsayılan recordset nesnesi oluşturulur.

Örneklenen Command nesnesinin özellikleri;

Bağlantı nesnesi ismi : Connection1

Komut nesnesi ismi : Command1

Komut Objesinin ismi : telefon

Varsayılan Recordset ismi : rstelefon

Seklinde olmuştur

Komut Nesnesi Üzerinde Komutların Isletilmesi

Data Environment'te komutların isletilmesi iki şekilde olabilir

- Recordset oluşturma şekli ile.(recordset oluşturulur ve çıktılar kullanılabilir.)
- Update işleminin yapılması (Update yapılır ve herhangi bir çıktısı olmaz.)

Recordset Açma Metodu;

Aşağıdaki örnekte Microsoft Access veritabanı dosyası olan (c:\veritabanı.mdb – ODBC ayarlarından ORNEK DSN ' e sahip) ORNEK DSN ine bağlanılarak bilgiler form üzerindeki bir Datagrid kontrolüne yönlendirilmektedir.

```
Private Sub Form_Load()  
'Eğer bağlantı kapalıysa bağlantı açılır.  
DataEnvironment1.Connection1.State=adStateClosed Then  
DataEnvironment1.Connection1.Open  
'Eğer Kapalıysa Varsayılan Recordset Açılır.  
DataEnvironment1.rsCommand1.State=adStateClosed Then  
DataEnvironment1.rsCommand1.Open  
Set DataGrid1.DataSource = DataEnvironment1.rsCommand1  
End Sub
```

Recordset nesnesi sayesinde veriler üzerinde konum değiştirilebilir.

Örneğimizdeki yapıda herhangi bir işlem aralığına

Sonraki kayıt için DataEnvironment1.rsCommand1.MoveNext

Önceki kayıt için DataEnvironment1.rsCommand1.MovePrevious

İlk kayıt için DataEnvironment1.rsCommand1.MoveFirst

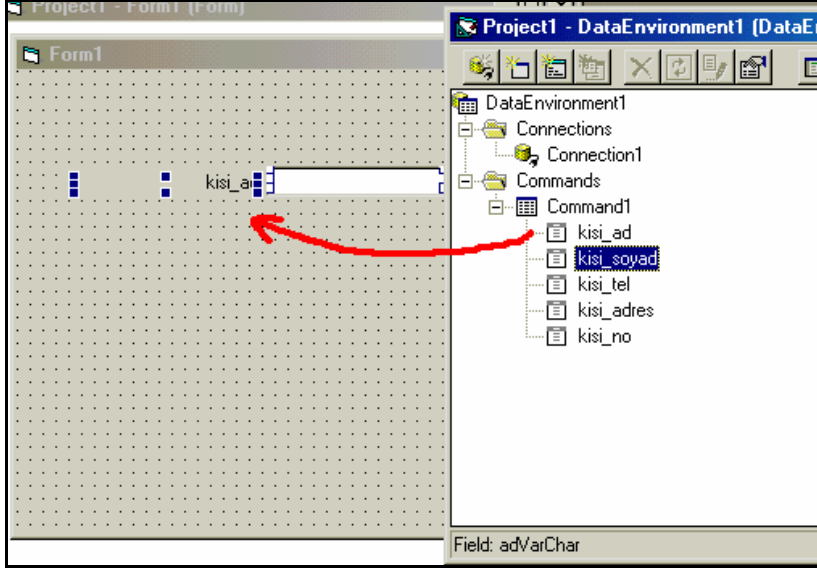
Son kayıt için DataEnvironment1.rsCommand1.MoveLast

Komutlarını eklememiz yeterli olacaktır.

Form Üzerindeki Nesnelerin Data Environment İlişkisi

Data Environment ile oluşturulan yapı gereği bir kaynaktaki verilere ulaşılabilir.Bu verilerin kullanıcıya sunulması şüphesiz form nesneleri ile

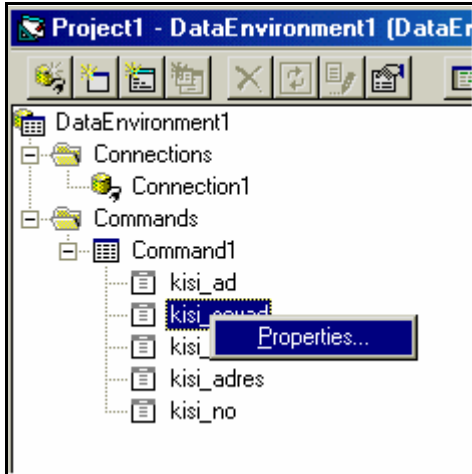
olmaktadır. Form nesnelerinin kaynak ile ilişkisinin olması için sürükleyip bırak tekniği kullanılır.



(Resim 14)

Resimde Data Environment içerisindeki Command1 nesnesinin içindeki kisi_ad alanına bağlı bir form nesnesi (Label-Textbox) oluşturulmuştur. (İşlemden sürükleyip bırak tekniği kullanılmıştır.)

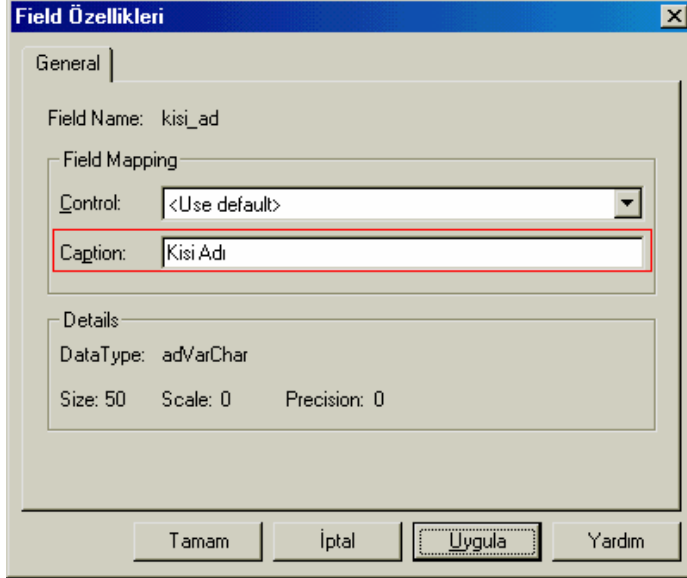
Eğer istenirse form nesnelerinin özellikleri Data Environment içindeki listelenen alanlara sağ tıklanarak açılan menüden **Properties** (bkz Resim 15) seçeneğine tıklanarak açılan diyalog penceresinden de yapılabilir.



(Resim 15)

Örneğin kisi_ad alanının Label nesnesinin Captionunu değiştirmek için

Data Environment içerisinde listelenen alanlardan kisi_ad seçeneğine sağ tusta tıklanır açılan menüden Properties'e tıklanır.(bkz Resim 16)



(Resim 16)

İstenen değişiklik yapıldıktan sonra da İşlem Tamam (OK) ile onaylanır ve bu işlem sonrasında sürükleyip bırak tekniği ile forma sürüklenen alan nesnesinin özelliği istenen şekilde oluşturulur.



(Değişiklikten önce)



(Değişiklikten Sonra)

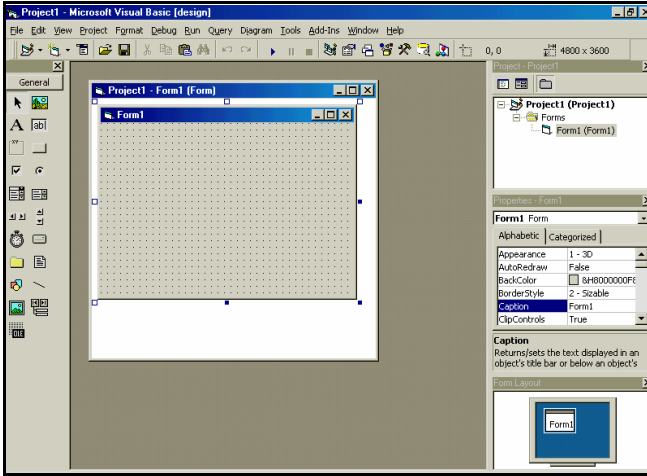
Örnek Uygulama

Örnek uygulama olarak ORNEK isimli DSN ' e bağlanmış olan bir Connection nesnesi kullanılmaktadır. Komut isimli Command nesnesi ORNEK DSN'i içindeki telefon tablosuna bağlıdır.

Adım Adım İşlemler

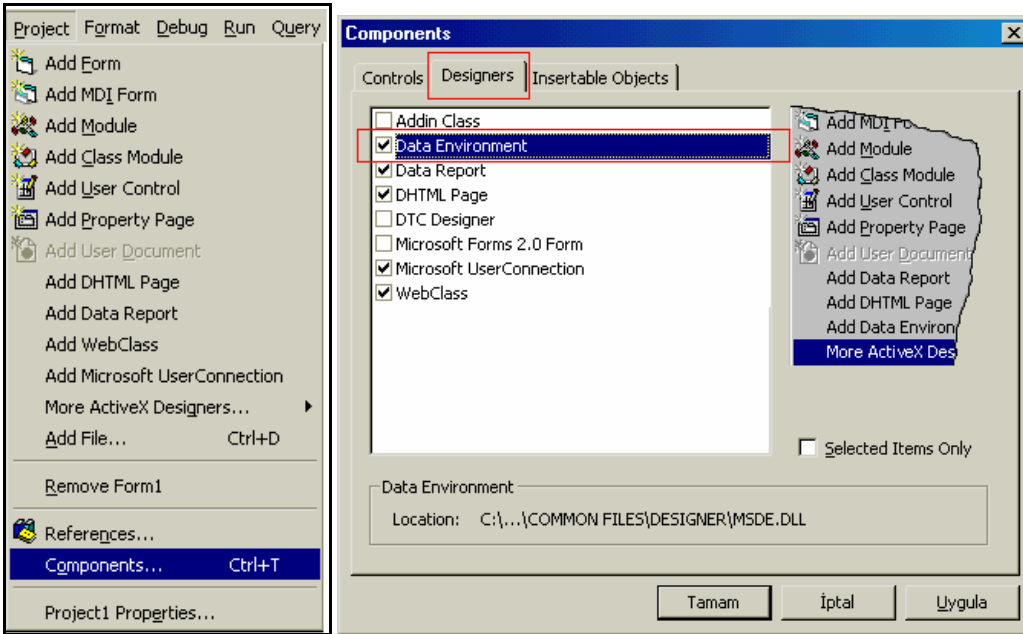
- Öncelikle Bos bir proje açıldı.(Resim 17)

Microsoft Visual Basic 6.0



(Resim 17)

- Projeye Data Environment Designer Eklendi

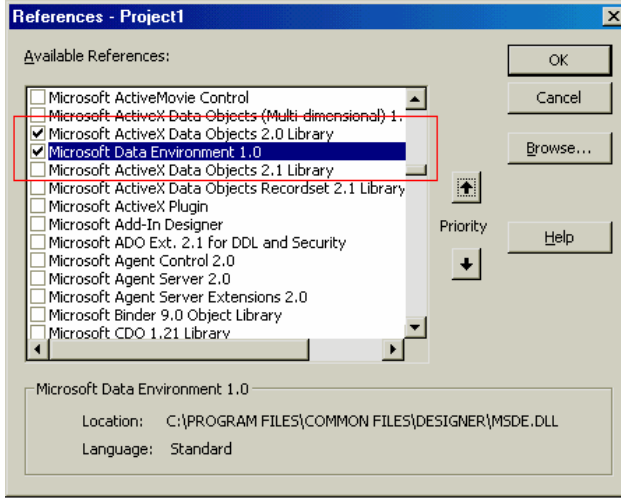


(Resim 18)

(Project menüsünden Components'e tiklanır açılan pencereden Designers sekmesinin altından Data Environment seçeneği işaretlenir, işlem onaylanır.)

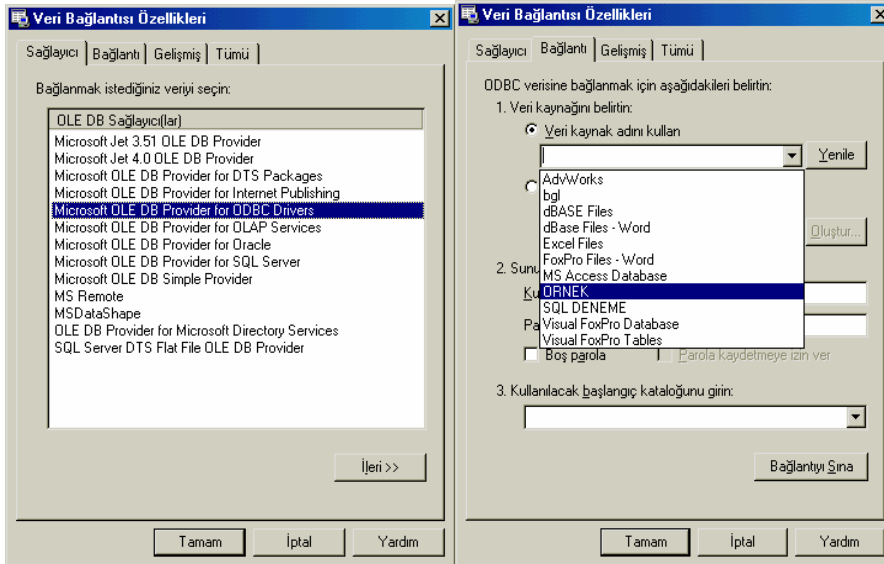
- Projeye gerekli referanslar eklendi;
(Microsoft ActiveX Data Object 2.0 Library ve Microsoft Data Environment 1.0)

(bkz Resim 19)



(Resim 19)

- Data Environment Designer'daki Connection1 nesnesi ODBC DSN'e bağlandı. (bkz Resim 20)

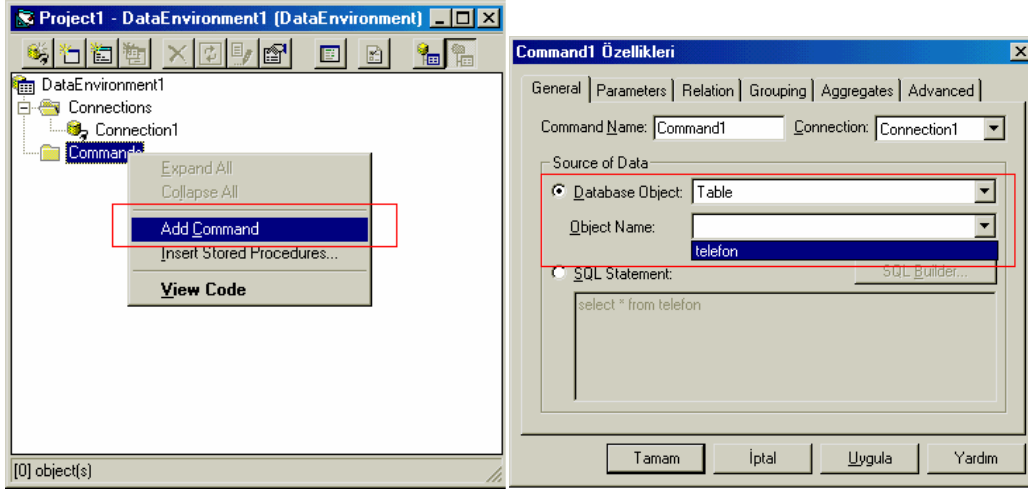


(Resim 20)

(Command1 nesnesinin özelliklerine girildi, sağlayıcı kısmından ODBC driveri seçildi ileri (next) butonuna tıklandı, sonrasında açılan pencereden ORNEK isimli DSN seçildi ve işlem onaylandı.)

Microsoft Visual Basic 6.0

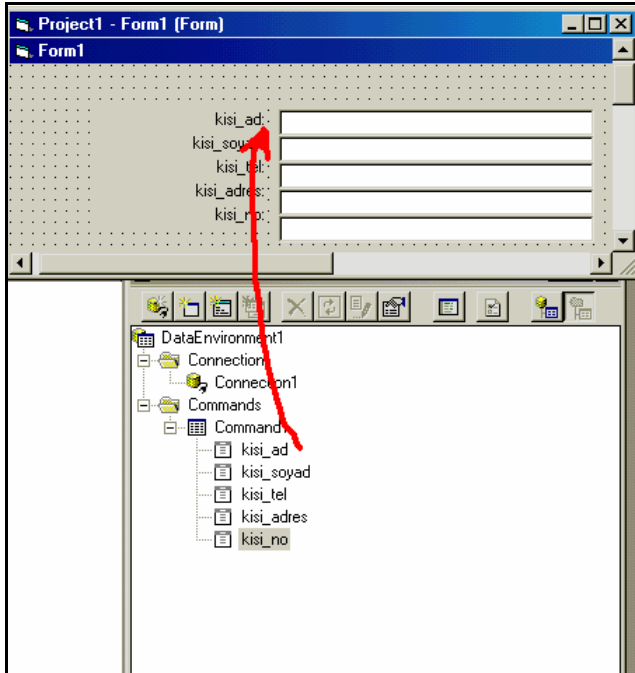
- Command nesnesi olusturuldu ve ayarlari yapildi. (bkz Resim 21)



(Resim21)

(Data Environment Designer içinde Command seçeneğine sag tusla tiklandı ve Add Command seçildi, açıklan pencerede Command ile ilgili ayarlamalar yapıldı (Ayarlamalar Database Object olarak table ve Object name olarak telefon değerlerinin kullanılacağı şekilde yapıldı.))

- Form elemanları üzerine alanlar eklendi (Sürükle Bırak Tekniği ile) (bkz Resim 22)



(Resim 22)

- Kayıtlar üzerinde gezinme için (ileri-geri) gerekli butonlar eklendi ve kodlari yazildi. (bkz Resim 23)

The screenshot shows a Windows-style application window titled "Data Environment Örneği". Inside the window, there are five text input fields arranged vertically, each with a label to its left: "kisi_ad:" containing "Nedim", "kisi_soyad:" containing "KARABULUT", "kisi_tel:" containing "02122122121", "kisi_adres:" containing "İstanbul", and "kisi_no:" containing "1". Below these fields, there are five buttons arranged horizontally: "Sonraki Kayıt", "Önceki Kayıt", "İlk Kayıt", "Son Kayıt", and "Çıkış". The "Çıkış" button is centered below the other four.

(Resim 23)

Kodlar;

```

Private Sub cikis_Click()
DataEnvironment1.Connection1.Close
End
End Sub
Private Sub ilk_kayit_Click()
If Not DataEnvironment1.rsCommand1.EOF Then
DataEnvironment1.rsCommand1.MoveFirst
End If
End Sub
Private Sub onceki_kayit_Click()
If Not DataEnvironment1.rsCommand1.BOF Then
DataEnvironment1.rsCommand1.MovePrevious
End If
End Sub
Private Sub son_kayit_Click()
If Not DataEnvironment1.rsCommand1.EOF Then
DataEnvironment1.rsCommand1.MoveLast
End If
End Sub
Private Sub sonraki_kayit_Click()
If Not DataEnvironment1.rsCommand1.EOF Then
DataEnvironment1.rsCommand1.MoveNext
End If
End Sub

```

Örnek Programlar Ve Açıklamaları

1-ADO Kontrolü ile yapılmış Örnek Uygulama

FORMLARIN AÇIKLAMASI

2-ActiveX Data Object ile yapılmış Örnek Uygulama

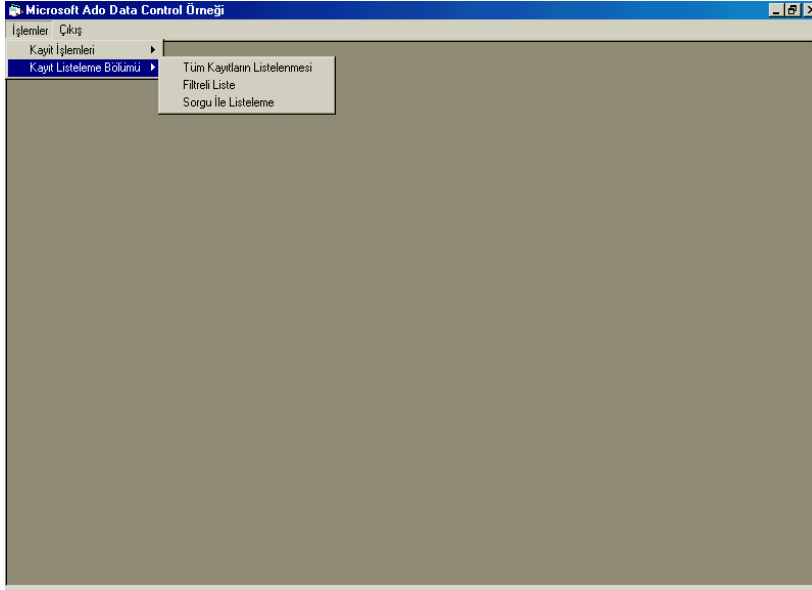
FORMLARIN AÇIKLAMASI

ADODC Önbilgi:	Ile	Yapılan	Örnek	Uygulama
-------------------	-----	---------	-------	----------

Programın amacı ADODC ile veri girişi, arama, silme, düzeltme, listeleme, filtreli listeleme gibi işlemleri örneklemektir. Basitçe telefon defteri işlevini görebilecek yapıdadır.

Programın çalışması için kitap ile verilen ek program CD sindeki ilgili klasör altında bulunan database.mdb dosyasının bir ODBC Sistem DSN i ile bağlanması gerekmektedir.DSN ismi “telefon_” olmalıdır.

Program biri MDI formu (bkz Resim 1) olmak üzere altı formdan oluşmaktadır.



(Resim 1)

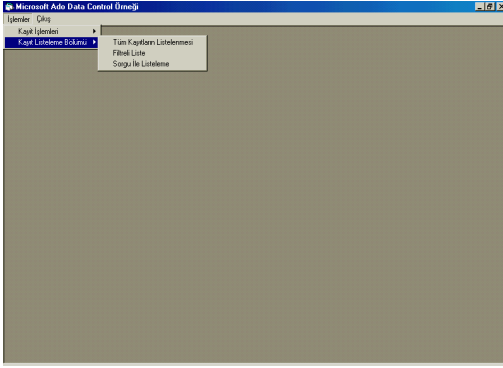
Bu formların basitçe kullanım amaçları

filtre_ile_liste	Filtreleme örneği
kayit_isl_1	Kayıt girme Örneği
kayit_isl_2	Genel kayıt işlemleri Örneği (Arama,Silme,Degistirme)
Liste1	Tüm kayıtların listelenmesi örneği
MDIForm1	Programın ana formu
sorgu_ile_liste	SQL sorgularının kullanılmasında dair örnek

MDIForm Açıklaması

Programın giriş formudur, kullanıcı menüler yardımı ile gerekli formları çalıştırabilir.Proje genelinde kullanılan Bağlantı tanımı Load yordamında yapılmıştır.(bkz Resim 2)

Microsoft Visual Basic 6.0



(Resim 2)

MDI formun kodlari Asagidaki gibidir;

```
Private Sub cik_Click()  
End  
'Program sonlandırılır.  
End Sub  
'Bu kismdaki kodlar formlarin aktiflesmesi amaçlidir.  
Private Sub fl_Click()  
filtre_ile_liste.Show  
End Sub  
  
Private Sub ke_Click()  
kayit_isl_1.Show  
End Sub  
  
Private Sub ksga_Click()  
kayit_isl_2.Show  
End Sub  
  
Private Sub ui_Click()  
unvan_isl.Show  
End Sub  
  
Private Sub MDIForm_Load()  
  
End Sub  
  
Private Sub s_i_l_Click()  
sorgu_ile_liste.Show  
End Sub  
  
Private Sub tkl_Click()  
Liste1.Show  
End Sub
```


KAYIT_ISL_1 Formunun Açıklaması

Kullanıcı bu form yardımı ile ad,soyad,telefon,adres,mail adresi gibi bilgileri Veritabanına ekleyebilir.

(KAYIT_ISL_1 formunun ekran görüntüsü)

KAYIT_ISL_1 Formunun kodlari asagidaki gibidir;

```
Private Sub temizle()  
'buradaki fonksyonun amaci asagidaki kodlari  
'çoğu kez tekrarlamamaktır  
'Bu kodlar çağrıldıklarında form elemanlarının  
'içi bosaltılır.  
ad.Text = ""  
soyad.Text = ""  
adres.Text = ""  
telefon.Text = ""  
mail.Text = ""
```

```
End Sub
```

```
Private Sub ad_KeyPress(KeyAscii As Integer)  
If KeyAscii = 13 Then  
soyad.SetFocus  
End If  
End Sub
```

```
Private Sub adres_Change()  
If KeyAscii = 13 Then  
telefon.SetFocus  
End If  
End Sub
```

```
Private Sub Command1_Click()  
'Ado nesnesi tazelenir  
Adodcl.Refresh
```

Microsoft Visual Basic 6.0

```
'Ado nesnesi yeni kayıt modunda açılır
Adodc1.Recordset.AddNew
'ado nesnesine bağlı olan alanların içi doldurulur.
Adodc1.Recordset("ad") = Trim(UCase(ad.Text))
Adodc1.Recordset("soyad") = Trim(UCase(soyad.Text))
Adodc1.Recordset("adres") = Trim(UCase(adres.Text))
Adodc1.Recordset("tel") = Trim(UCase(telefon.Text))
Adodc1.Recordset("mail") = Trim(mail.Text)
'Ado nesnesi güncellenir
Adodc1.Recordset.Update
'Ado nesnesi tazelenir
Adodc1.Refresh
'Form elemanlarının içi boşaltılır.
temizle
End Sub

Private Sub Command2_Click()
Unload Me
'Form Unload edilir.
End Sub

Private Sub Form_Load()
'Form yüklenirken form elemanlarının içi boşaltılır.
temizle
End Sub

Private Sub mail_KeyPress(KeyAscii As Integer)
'mail text'ine değer yazılıp enter'e basıldıktan sonra
'cursor Command1 (kayıt) butonuna konumlanır
If KeyAscii = 13 Then
Command1.SetFocus
End If
End Sub

Private Sub soyad_KeyPress(KeyAscii As Integer)
'soyad text'ine değer yazılıp enter'e basıldıktan sonra
'cursor adres textine konumlanır.
If KeyAscii = 13 Then
adres.SetFocus
End If
End Sub

Private Sub telefon_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Telefon textine bilgi girildikten sonra Enter'a basıldığında
'cursor mail text'ine konumlanacaktır.
'Programın akisini hızlandırmak amaçlı yapılmış yönlendirme cursor
islemidir
mail.SetFocus
```

```
End If
End Sub
```

KAYIT_ISL_2 Formunun Açıklaması

KAYIT_ISL_2 formu kullanıcının veriler üzerinde genel işlemleri yapmasını sağlar (Arama, Degistirme, Silme gibi).Kullanıcı KAYIT_ISL_2 formuna MDI form üzerinden ulaşabilir.

Proje çalışırken form aktif olduğunda kullanıcı text1 nesnesine bilgileri üzerinde işlem yapmak istediği kayitin ad text2 nesnesine soyad bilgilerini girer ve ara butonuna tıklar, arama işlemi Adodc1 nesnesinin CommandType özelliğinin adCmdText sekline dönüştürülmesiyle sql sorgusunu adodc nesnesine gönderir ve çıktı olarak gelen veriler form elemanlarına dağıtılır.Bu işlem sonrasında basta pasif halde olan silme ve degistirme butonları aktiflesir ve kullanıcı istediği işlemi yapar .

KAYIT_ISL_2 Formunun Kodları aşağıda açıklamaları ile incelenebilir;

```
Private Sub Command1_Click()
'Ado nesnesinin komut tipi Text olarak degistirilir
Adodc1.CommandType = adCmdText
'Adodc1 nesnesinin kayıt kaynağına SQL sorgusu yazılır
Adodc1.RecordSource = "SELECT * FROM TELEFON WHERE(AD= ' " &
Trim(UCase(Text1.Text)) & " ' AND SOYAD= ' " &
Trim(UCase(Text2.Text)) & " ')"
'Sorgu kayıt arama amaçlıdır
'form elemanlarının içi kayıt taki alan bilgileri ile dolar.
ad.Text = Adodc1.Recordset.Fields("ad")
soyad.Text = Adodc1.Recordset.Fields("soyad")
adres.Text = Adodc1.Recordset.Fields("adres")
tel.Text = Adodc1.Recordset.Fields("tel")
mail.Text = Adodc1.Recordset.Fields("mail")
'kayıt silme ve kayıt degistirme butonları aktif hale getirilir.
kayıt_degistir.Enabled = True
kayıt_sil.Enabled = True
End Sub

Private Sub Command4_Click()
Unload Me
'Form Unload edilir.
End Sub

Private Sub temizle()
'Form elemanlarını içi temizlenir.
ad.Text = ""
soyad.Text = ""
adres.Text = ""
tel.Text = ""
mail.Text = ""
```

Microsoft Visual Basic 6.0

End Sub

```
Private Sub Form_Activate()  
'Form Aktif hale geçince kursor text1 nesnesine odaklanır.  
Text1.SetFocus  
End Sub  
Private Sub Form_Load()  
'Form yüklendiğinde Adodc1 nesnesi yenilenir.  
Adodc1.Refresh  
End Sub
```

```
Private Sub kayit_degistir_Click()  
'Daha önceden sql sorgusu ile databaseden çekilen veriler  
'Adodc1 nesnesinin içerigindedir.Form elemanlarının datafield  
degerleri  
'adodc1 nesnesinin içinde barındırdığı tablonun alanlarına  
baglanmistir  
'Adodc update edilir ve degisiklikler form nesnelerinden alinip  
adodc1  
'recordsetine yani kaynaga yazilir  
Adodc1.Recordset.Update  
'kayit degistirme ve kayit silme butonlari pasif edilir  
kayit_degistir.Enabled = False  
kayit_sil.Enabled = False  
'Form temizlenir  
temizle  
End Sub
```

```
Private Sub kayit_sil_Click()  
'Adodc1 in içinde bulundurduğu kayit silinir.  
Adodc1.Recordset.Delete  
'Adodc1 güncellenir degisiklikler uygulanir.  
Adodc1.Recordset.Update  
'Form elemanlari temizlenir.  
temizle  
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)  
If KeyAscii = 13 Then  
'Text2 text kutusunda iken kullanıcı Enter'a basarsa  
'kursor text2 nesnesine odaklanır  
Text2.SetFocus  
End If  
End Sub
```

```
Private Sub Text2_KeyPress(KeyAscii As Integer)  
If KeyAscii = 13 Then
```

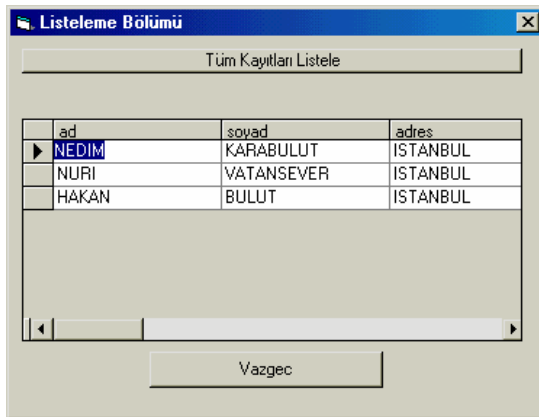
```

'kullanici text2 text kutusunda iken kullanıcı Enter'a basarsa
'kursor Command1 nesnesine odaklanır
Command1.SetFocus
End If
End Sub

```

Liste1 Formunun Açıklaması

Liste1 formu kullanıcının SELECT deyimi sayesinde veritabanından süzülen verileri datagrid nesnesine yönlendirmesi ile tüm kayıtların kullanıcı tarafından görülmesini sağlar.



(Liste1 formunun ekran görüntüsü)

Liste1 formunun kodları ve kod açıklamaları aşağıda incelenebilir.

```

Private Sub cikis_Click()
Unload Me
'Form Unload edilir.
End Sub

```

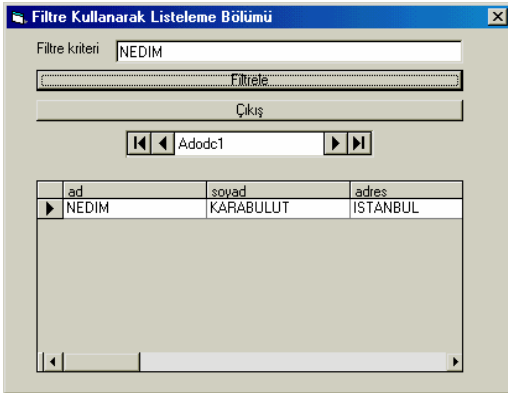
```

Private Sub Command1_Click()
'Adodcl nesnesinin recordsource ayarı yapılır
'Asagidaki SQL sorgusunun anlamı tüm kayıtlar dir.
Adodcl.RecordSource = "SELECT * FROM telefon"
'DataGrid1 nesnesi güncellenir.
Set DataGrid1.DataSource = Adodcl
End Sub

```

FILTRE İLE LİSTE Formunun Açıklaması

Filtre_ile_liste formunun amacı kullanıcının form üzerindeki data grid nesnesinde gösterilen tüm kayıtları filtre text nesnesine girilen metin kriterinde filtrelenmesini örneklemektir.



(filtre_ile_liste formunun ekran görüntüsü)

FILTRE_ILE_LISTE Formunun kodlari ve kod açıklamalari asagida incelenebilir.

```
Private Sub cikis_Click()  
'Form Unload edilir.  
Unload Me  
End Sub
```

```
Private Sub filtrele_Click()  
'Recordsetin Filter özelliginden yararlanilarak ad alanin  
'Text1'e yazilan metine göre filtrelenmesini saglar  
Adodc1.Recordset.Filter = "ad = '" & Text1.Text & "'" &  
'Degisikligi kullaniciya yansimasi için gridin  
'Datasource özelligi Adodc1 nesnesi ile bagdasir.  
Set Grid.DataSource = Adodc1  
End Sub
```

SORGU İLE LİSTE Formunun Açıklaması

Adodc nesnesinde SQL sorguları çalıştırılabilir.SORGU_ILE_LISTE formunun amacı adodc nesnesi ile sql sorgularının çalıştırılmasını örneklemektir.

(Formun Ekran Görüntüsü)

ActiveX Data Object ile Yapılan Örnek Uygulama Ön bilgi:

Programın amacı ActiveX Data Object ile veri girişi , veri silme, arama, veri düzenleme gibi işlemleri ve recordset nesnelerini, bağlantı nesnelerini çalışma prensibine işik tutmaktır.

Programın çalışması için kitap ile verilen ek program CD sindeki ilgili klasör altında bulunan "veritabani.mdb" dosyasının bir ODBC Sistem DSN i ile bağlanması gerekmektedir.DSN ismi "ACTIVEX" olmalıdır.

Program biri MDI formu (bkz Resim 1) olmak üzere altı formdan oluşmaktadır.

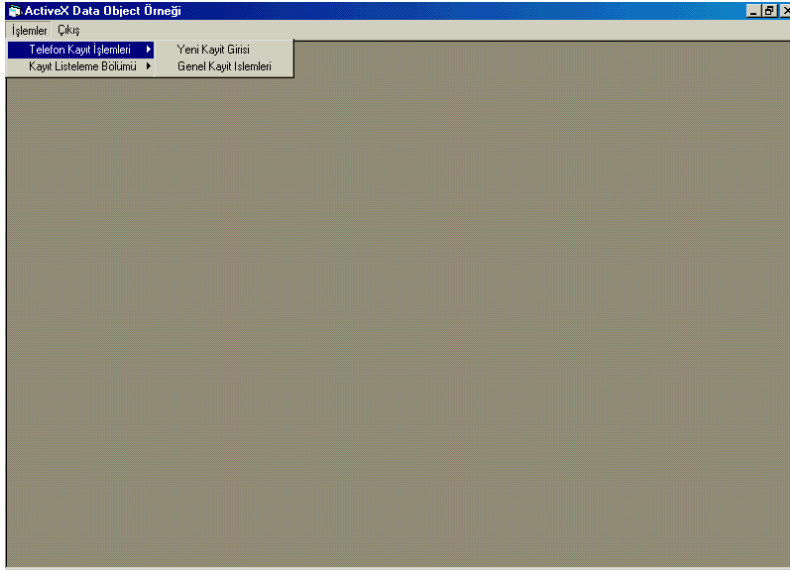
Bu formların basitçe kullanım amaçları

ANA_FORM	Programın ana formu
filtreli_liste	Filtreli listeleme formu
genel	Genel kayıt işlemleri Örneği (Arama,Silme,Degistirme)
Listele	Tüm kayıtların listelenmesi örneği
sorgu_ile_liste	Sorgunun listelemede kullanılmasına dair örnek
yeni_kayit	Veritabanına yeni kayıt girme amaçlı form

ANA Form Açıklaması

Microsoft Visual Basic 6.0

Ana form bir MDI formdur ve görevi kullanıcının diğer formlara ulaşmasını sağlamaktır.



(Formun Ekran Görüntüsü)

Formun kodları ve kodlarının açıklamaları aşağıda incelenebilir;

```
Private Sub cik_Click()  
End  
'program sonlandırılır.  
End Sub  
'Bu kısımdaki kodlar formların aktif hale geçmesi içindir.  
Private Sub fl_Click()  
filtreli_liste.Show  
  
End Sub  
  
Private Sub gki_Click()  
genel.Show  
  
End Sub  
  
Private Sub MDIForm_Load()  
'açılısta bağlantı connection nesnesi oluşturulur.  
'Burada dikkat edilmesi gereken bağlantının "bağlantı" anahtarı  
'ile "ACTIVEX" isimli sistem dsn yapısı ile ilişkilendirilmiş  
olduğudur.  
'uid = kullanıcı adını , pw ise şifreyi simgeler
```



```
baglanti.Open "ACTIVEX", uid = "", pw = ""
End Sub
```

```
Private Sub s_ile_liste_Click()
sorgu_ile_liste.Show
End Sub
```

```
Private Sub tk_l_Click()
Listele.Show
End Sub
```

```
Private Sub yeni_kayt_Click()
yeni_kayit.Show
End Sub
```

YENI_KAYIT Formu Açıklaması

Yeni_kayit formu kullanıcının yeni kayıt girilmesi amacıyla kullanabileceği bir formdur. Formda ilgili yordamlarda kayıt nesnesi açılır ve form elemanlarının içeriği kayıt nesnesinin alanlarına yazılır.

(Formun ekran görüntüsü)
Formun kodları ve kodların açıklamaları aşağıda incelenebilir.

```
Private Sub temizle()
'Prosedür form elemanlarını temizlemede kullanılır.
ad.Text = ""
soyad.Text = ""
adres.Text = ""
mail.Text = ""
telefon.Text = ""
```

```
End Sub
```

```
Private Sub ad_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Eğer kullanıcı ad text'inde iken enter tusuna basarsa
'kursor soyad text'ine konumlanır.
soyad.SetFocus
```

Microsoft Visual Basic 6.0

```
End If
End Sub
```

```
Private Sub adres_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Eger kullanıcı adres text'inde iken enter tusuna basarsa
'kursor mail text'ine konumlanır.
mail.SetFocus
End If
```

```
End Sub
```

```
Private Sub cikis_Click()
Unload Me
End Sub
```

```
Private Sub Form_Activate()
'Form aktif hale geldiginde kursor ad textbox'una konumlanır.
ad.SetFocus
End Sub
```

```
Private Sub Form_Load()
'Form yüklendiginde form bileşenleri temizlenir.
temizle
'komut command nesnesi oluşturulur
With komut
.CommandType = adCmdText
.CommandText = "SELECT * FROM telefon"
Set .ActiveConnection = baglanti
End With
```

```
'kayıt isimli recordset oluşturulur.
With kayıt
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With
```

```
End Sub
```

```
Private Sub kaydet_Click()
'kayıt isimli recordset Yeni kayıt ekleme modunda açılır
kayıt.AddNew
'form nesnelerinin içeriği alanlara yerleştirilir.
kayıt.Fields("ad") = Trim(UCase(ad.Text))
kayıt.Fields("soyad") = Trim(UCase(soyad.Text))
kayıt.Fields("adres") = Trim(UCase(adres.Text))
kayıt.Fields("mail") = Trim(mail.Text)
```

```
kayit.Fields("tel") = Trim(UCase(telefon.Text))
'kayit recordseti güncellenir.
kayit.Update
'Form elemanlari temizlenir.
temizle
'kullanici bilgilendirilir.
MsgBox "Kayit Tamamlandi"
End Sub

Private Sub mail_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Eger kullanıcı mail text'inde iken enter tusuna basarsa
'kursor telefon text'ine konumlanir.
telefon.SetFocus
End If

End Sub

Private Sub soyad_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Eger kullanıcı soyad text'inde iken enter tusuna basarsa
'kursor adres text'ine konumlanir.
adres.SetFocus
End If

End Sub

Private Sub telefon_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
'Eger kullanıcı telefon text'inde iken enter tusuna basarsa
'kursor kaydet nesnesine konumlanir.
kaydet.SetFocus
End If

End Sub
```

Genel Formu Açıklaması

Genel formu kullanıcının girilen kayıtlar arasından ad ve soyad bilgileri sayesinde aradığı kaydı bulmasını ve istediği kaydı değiştirmesi, silmesi gibi işlemleri yapmasını sağlar.

Microsoft Visual Basic 6.0

Genel Kayıt İşlemleri Bölümü

Ad: NEDİM
Soyad: KARABULUT
Adres: İSTANBUL
E-Mail Adresi: nedimkarabulut@hotmail.com
Telefon Numarası: 02122122122

Ara

Ad: nedim
Soyad: karabulut

Vazgeç
Kaydı değiştir
Kaydı Sil

(Formun ekran görüntüsü)

Kaydı sil ve Kaydı Degistir butonlari ilk olarak pasiftir , kullanıcı bir ad ve soyad bilgisi girer ve ara butonuna bastiginda eger kayıt bulursa bu iki buton aktif hale geçecektir.

Formun kodlari ve kodlariin açıklamaları aşağıda incelenebilir;

```
Private Sub temizle()  
'temizle prosedürünün amacı çağrıldığı anda  
'aşağıdaki form nesnelerinin içini boşaltmaktır.  
ad.Text = ""  
soyad.Text = ""  
adres.Text = ""  
mail.Text = ""  
telefon.Text = ""  
Text1.Text = ""  
Text2.Text = ""  
  
End Sub  
  
Private Sub ara_Click()  
  
'Eğer kayıt isimli recordset açıksa kapatır  
'Çünkü aşağıdaki ayarlar recordset oluşturulurken yapılmalıdır.  
If kayıt.State = 1 Then  
kayıt.Close  
End If  
  
'komut isimli Command nesnesi oluşturulur ve sorgu dizayn edilir.  
With komut  
.CommandType = adCmdText
```

```

.CommandText = "SELECT * FROM TELEFON WHERE (AD= ' " &
Trim(UCase(Text1.Text)) & " ' AND SOYAD= ' " &
Trim(UCase(Text2.Text)) & " ')"
'aktif baglanti seçilir.
Set .ActiveConnection = baglanti
End With

'kayit isimli recordset nesnesi olusturulur.
'Ve konfigure edilir.
With kayit
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With

'Eger kayit bulunamazsa form elemanlarinin içine
'recordset alanindaki bilgiler yazilmaz
'Eger kayit bulunamazsa ve form nesnelerinin içine
'alan bilgileri doldurulma istenirse hata olusur
If kayit.RecordCount <> 0 Then
ad.Text = kayit("ad")
soyad.Text = kayit("soyad")
adres.Text = kayit("adres")
telefon.Text = kayit("tel")
mail.Text = kayit("mail")
'Silme ve degistirme butonlari aktif edilir
sil.Enabled = True
degistir.Enabled = True
Else
'Form Elemanlari temizlenir.
temizle
End If
'kayit recordseti kapatilir.
kayit.Close

End Sub

Private Sub cikis_Click()
Unload Me
'Form unload edilir.
End Sub

Private Sub degistir_Click()
'komut isimli command nesnesi olusturulur ve konfigure edilir.
With komut
.CommandType = adCmdText
.CommandText = "SELECT * FROM TELEFON WHERE (AD= ' " &
Trim(UCase(Text1.Text)) & " ' AND SOYAD= ' " &
Trim(UCase(Text2.Text)) & " ')"

```

Microsoft Visual Basic 6.0

```
'baglanti ile komut nesnesi baglanir, yani aktif baglanti olarak
'mevcut baglanti secilir.
Set .ActiveConnection = baglanti
End With

'kayit isimli recordset tanimlanir.
With kayit
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With

'form nesnelerinin icerigindeki degistirilmis veriler
'recordset nesnesinin icerindeki alanlara esitlenir
kayit("ad") = Trim(UCase(ad.Text))
kayit("soyad") = Trim(UCase(soyad.Text))
kayit("adres") = Trim(UCase(adres.Text))
kayit("mail") = Trim(mail.Text)
kayit("tel") = Trim(UCase(telefon.Text))
'recordset guncellenerek degisiklikler kaydedilir.
kayit.Update
'Form elemanlari temizlenir.
temizle

'degistirme ve silme butonlari eskisi gibi pasifize edilir.
degistir.Enabled = False
sil.Enabled = False
'kayit recordseti kapatilir.
kayit.Close
End Sub

Private Sub Form_Activate()
'Form aktif hale gecince kursor text1'de konumlanir.
Text1.SetFocus
End Sub

Private Sub Form_Load()
'Form yuklendiginde form elemanlarin ici temizlenir.
temizle
End Sub

Private Sub sil_Click()
'komut isimli command nesnesi olusturulur ve konfigure edilir.
With komut
.CommandType = adCmdText
.CommandText = "SELECT * FROM TELEFON WHERE (AD=' " &
Trim(UCase(Text1.Text)) & " ' AND SOYAD=' " &
Trim(UCase(Text2.Text)) & " ')"
Set .ActiveConnection = baglanti
End With
'kayit isimli recordset nesnesi olusturulur.
```

```

With kayit
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With

'kayit isimli recordsetin içindeki kursorun aktif olduğu
'kayit silinir.
kayit.Delete
'kayit isimli recordset güncellenir.
kayit.Update
'Form elemanlari temizlenir.
temizle

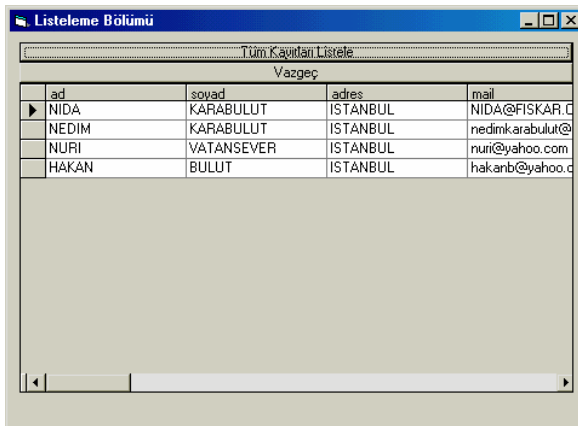
'recordset kapatilir
kayit.Close

'Degistirme ve Silme butonlari aktif hale getirilir.
degistir.Enabled = False
sil.Enabled = False
End Sub

```

Listele Formunun Açıklaması

Listele formu vasitasi ile kullanıcı database'deki tüm kayıtları dbgrid nesnesinde görüntüleyebilir.



(Formun Ekran Görüntüsü)

Formun kodları ve kodlarının açıklamaları aşağıda incelenebilir.

```

Private Sub temizle()
'form elemanlarının içeriği temizlenir.
filtre_.Text = ""

```

Microsoft Visual Basic 6.0

```
sql.Text = ""
```

```
End Sub
```

```
Private Sub cikis_Click()  
Unload Me  
'form geriyüklenir edilir.  
End Sub
```

```
Private Sub tum_kayitlar_Click()  
'Eger kayıt isimli recordset açıksa kapatır  
'Çünkü asagidaki ayarlar recordset olusturulurken yapılmalıdır.  
If kayıt.State = 1 Then  
kayıt.Close  
End If
```

```
'komut nesnesi olusturulur ve konfigure edilir  
With komut  
.CommandType = adCmdText  
.CommandText = "SELECT * FROM telefon"  
Set .ActiveConnection = baglanti  
End With
```

```
'kayıt nesnesi olusturulur ve komutla baglanır  
With kayıt  
.CursorType = adOpenStatic  
.LockType = adLockOptimistic  
.Open komut  
End With  
'grid nesnesi güncellenir  
Set Grid.DataSource = kayıt  
'grid tazelenir  
Grid.Refresh  
End Sub
```

SORGU İLE LİSTE Formunun Açıklaması

SQL sorgularının ado nesnesinde kullanımına örnek olması açısından tasarlanmıştır.Kullanıcı sorgu text kutusuna istediği sql sorgusunu yazar ve çalıştırabilir.

ad	soyad	adres	
NIDA	KARABULUT	ISTANBUL	
NEDİM	KARABULUT	ISTANBUL	
NURI	VATANSEVER	ISTANBUL	
HAKAN	BULUT	ISTANBUL	

(Formun Ekran Görüntüsü)

Formun kodlari ve kodlarinin incelemeleri asagida incelenebilir;

```
Private Sub Command1_Click()
'Eger kayıt isimli recordset açıksa kapatır
'Çünkü asagidaki ayarlar recordset oluşturulurken yapılmalıdır.
If kayıt.State = 1 Then
kayıt.Close
Set Grid.DataSource = kayıt
End If

'komut isimli command nesnesi oluşturulur ve ayarlanır.
With komut
.CommandType = adCmdText
.CommandText = Trim(sql.Text)
Set .ActiveConnection = baglanti
End With
'kayıt isimli recordset oluşturulur
With kayıt
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With
'grid nesnesi güncellenir.
Set Grid.DataSource = kayıt

End Sub
```

FILTRELİ_LISTE Formunun Açıklaması

Formun amacı recordset nesnesinin filter özelliğini örneklemektir. Kullanıcı form üzerinde filtre nesnesine gireceği değere göre (ad alanını etkiler) kayıtları filtrelenmiş şekilde listeleyebilir.

ad	soyad	adres	mail
NIDA	KARABULUT	İSTANBUL	NIDA@FISKAR.COM

Formun kodları ve kodların açıklamaları aşağıda incelenebilir;

```
Private Sub Command1_Click()  
Unload Me  
'Form unload edilir  
End Sub
```

```
Private Sub filtrele_Click()  
'Eğer kayıt isimli recordset açıksa kapatır  
'Çünkü aşağıdaki ayarlar recordset oluşturulurken yapılmalıdır.  
If kayıt.State = 1 Then  
kayıt.Close  
End If
```

```
'komut nesnesi oluşturulur ve konfigure edilir.  
With komut  
.CommandType = adCmdText
```

```

.CommandText = "SELECT * FROM telefon"
Set .ActiveConnection = baglanti
End With

'kayit isimli recordset olusturulur ve komut nesnesi ile
'baglanir.
With kayit
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut 'komut nesnesi ile recordset baglanir.
End With
'Girilen kriterlere göre sorgu olusturulur ve isletilir.
kayit.Filter = "ad='" & filtre_.Text & "'"
'grid güncellenir
Set Grid.DataSource = kayit
End Sub

Private Sub Form_Load()
'Eger kayit isimli recordset açıksa kapatir
'Çünkü asagidaki ayarlar recordset olusturulurken yapilmalidir.
If kayit.State = 1 Then
kayit.Close
End If

'Komut nesnesi ayarlanir
With komut
.CommandType = adCmdText
.CommandText = "SELECT * FROM telefon"
Set .ActiveConnection = baglanti
End With

'Kayit isimli recordset olusturulur.
With kayit
.CursorType = adOpenStatic
.LockType = adLockOptimistic
.Open komut
End With
'Grid nesnesi güncellenir
Set Grid.DataSource = kayit
'Grid nesnesi tazelenir.
Grid.Refresh

End Sub

```

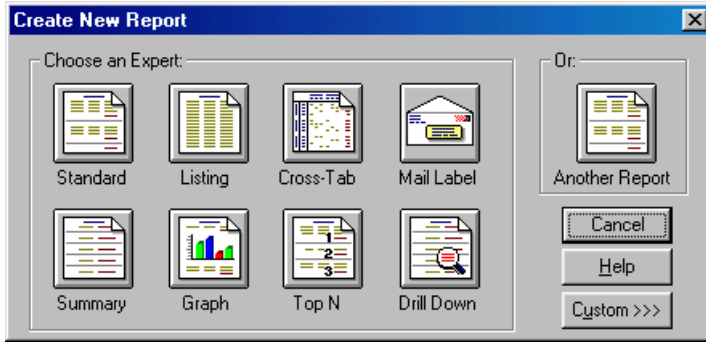
11-CYRISTAL REPORTS KULLANIMI

Crystal reports,Visual Basic ile beraber gelen bir rapor üretim programidir. Crystal reports ile farklı formatlardaki database'lerden ve SQL query'lerinden rapor almak mümkündür.

Crystal Reports Menüleri:

1- FILE:

New:



Report :Yeni bir rapor sayfası açar.

Cross-Tab :Çapraz kayıtlar için yeni bir rapor sayfası hazırlar.

Mail Label :Yeni bir zarf etiketi sayfası açar.Bu seçenekle kayıtlardaki isim ve adres gibi veriler kullanılarak posta etiketleri kolay bir şekilde hazırlanabilir.

Open:Önceden kaydedilmiş bir rapor dosyasını getirir.

Save:Aktif rapor dosyasını kaydeder.

Save As:Aktif rapor dosyasını değişik bir yere veya değişik bir isimle kaydeder.

Close:Aktif rapor dosyasını kapatır.

Print Preview:Aktif rapor dosyasının baskı ön izleme ekranında görünümünü sağlar.

Print

Printer:Aktif raporu yazıcıya gönderir.

Export:Baskı bir formatta kaydeder.

Report Definition:Rapor yapısını yazıcıya gönderir.

Printer Setup:Windows Printer Setup dialog kutusunu getirir.

Page Margins:Sayfa marj ayarları için kullanılır.

Options: Genel olarak veya çeşitli bölümlerin (fields,database)parametre ve formatlarını değiştirmek için kullanılır. Değiştirilen ayarlar daha sonra yaratılacak olan raporlar için de geçerlidir.

Exit :Crystal Reports' u sona erdirir.

2-EDIT

Cut:Seçili nesneyi siler.

Copy:Seçili nesneyi hafızaya alır.

Paste:Cut veya Copy edilen nesneyi bulunulan noktaya yapıştırır.

Paste Special:Paste işleminin istenen formatta yapılmasını sağlar.

Select Fields:Rapor nesnesinde bulunan alanları seçmek için kullanılır.

Formula:İçerdiği formül olan bir field'ın formülünü değiştirmek için kullanılır.

Text Field :İçerdiği formül olan bir field'ın text'ini değiştirmek için kullanılır.

Summary Field:İçerdiği özet bilgi olan bir field'ın özet bilgi tipini değiştirmek için kullanılır.

Browse Field Data:Seçili olan field'ın içeriğini görüntülemek için kullanılır.

Show/Hide Sections:İstenen kesimleri gizlemeye veya göstermeye yarar.

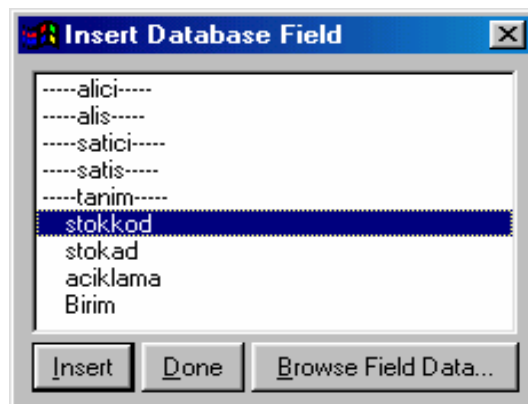
Group Section:Varolan bir grup kesiminin özelliklerini değiştirmek için kullanılır.

Delete Section:Varolan bir kesimi silmek için kullanılır.

Object:Daha önceden eklenmiş bir windows nesnesinin özelliklerini değiştirmeye yarar.

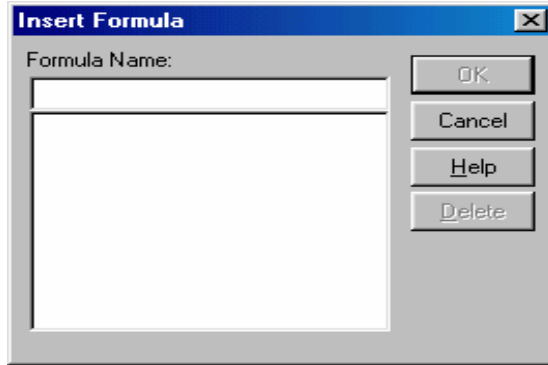
Links:Daha önceden yaratılmış olan database linklerinin özelliklerini değiştirmek için kullanılır.

3-INSERT



Database Field: Raporu bir database field eklemek için kullanılır.

Text Field : Raporu sabit bir bilgi eklemek için kullanılır.



Formula Field : Raporu içeriği formül olan bir bilgi eklemek için kullanılır.

Special Field

Page Number Field: Sayfa numarası

Record Number Field: Kayıt sayısı

Group Number Field: Grup sayısı

Print Date Field: Rapor baskı tarihi eklemek için kullanılır.

Subtotal: Aynı gruptaki field'ları içeren toplam veya ortalama eklemek için kullanılır.

Grandtotal: Tüm raporu içeren toplam veya ortalama eklemek için kullanılır.

Summary: Bir gruptaki özet bilgiyi elde etmek için kullanılır. Bu bilgi bir field'ın minimum-maximum, ortalama, toplam, kayıt sayısı gibi bilgiler olabilir.

Group Section: İstenen bir field'e göre bir grup yaratmak için kullanılır.

Line: Çizgiyi eklemek için kullanılır.

Box: Dörtgen eklemek için kullanılır.

Graphic: Grafik eklemek için kullanılır.

Object: Bir Windows nesnesi eklemek için kullanılır.

4-FORMAT

Font: Yazı tipi

Field: Bir alan

Border and Colors: Kenarlıklar ve renkler

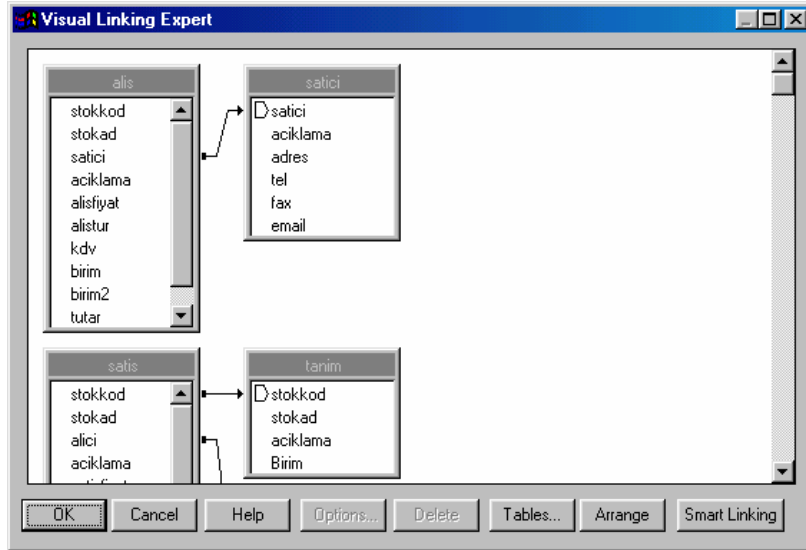
Graphic: Grafik

Line: Çizgi

Box:Dörtgen

Section:Kesimleri formatlamak için kullanılır.

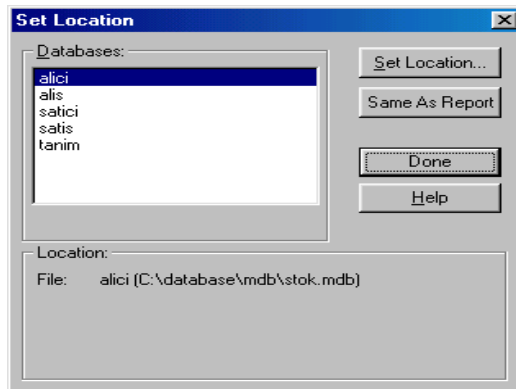
5-DATABASE



Visual Linking Expert:İki table'in aynı ortak field'leri üzerinde bir bağlantı yaratmak için kullanılır.Böylelikle ortak field yolu ile her iki table'daki dataya da tek field ile ulaşılabilir.

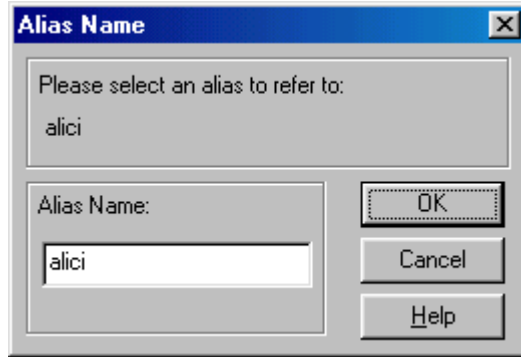
Add Database to Report:Rapora database eklemek için kullanılır.Birden fazla database 'den ortak rapor alınmak istendiğinde bu yol kullanılır.

Remove from Report:Rapordan bir database veya table çıkarmaya yarar.



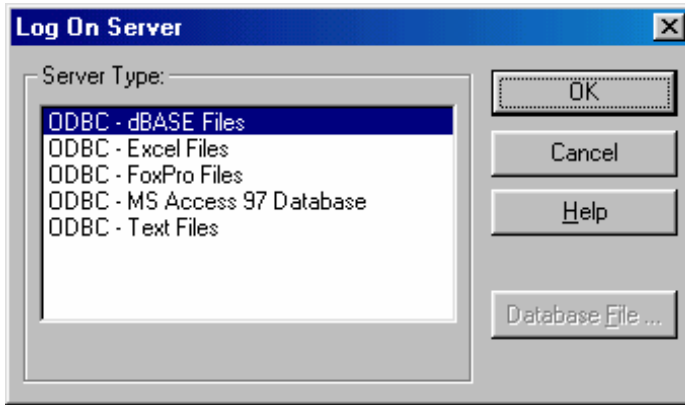
Set Location:Bir database veya table'in path'ini belirtmeye yarar.Fiziksel yer değişikliği olduğunda rapor database veya table nesnesinin yerini bulamayabilir.Bu durumda Set Location ile yer belirtilir.

Set Alias:Bir database veya table'a takma ad vermeye yarar.Takma adlar kolay kullanım,hatirlama,v.b.için kullanılabilir.



Verify Database:Database'in yapısında bir degisiklik oldugunda(field ekleme,field'in ismini degistirme)Crystal Reports'un bu degisikligi rapora yansitabilmesi için kullanılır.

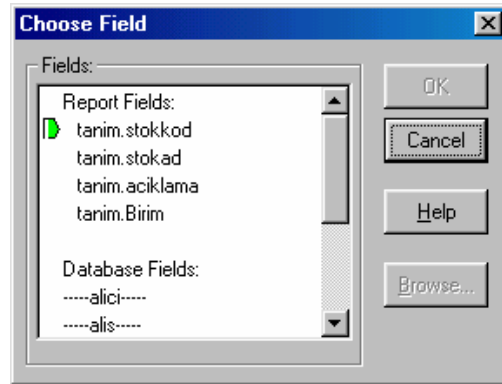
Verify On Every Print:Bir önceki her printtan önce isleme sokar.Isaretili olarak kullanım içi,isaretsiz olarak kullanım disidir.



Log On Server:Bir SQL Server database'ini kullanima sokar.

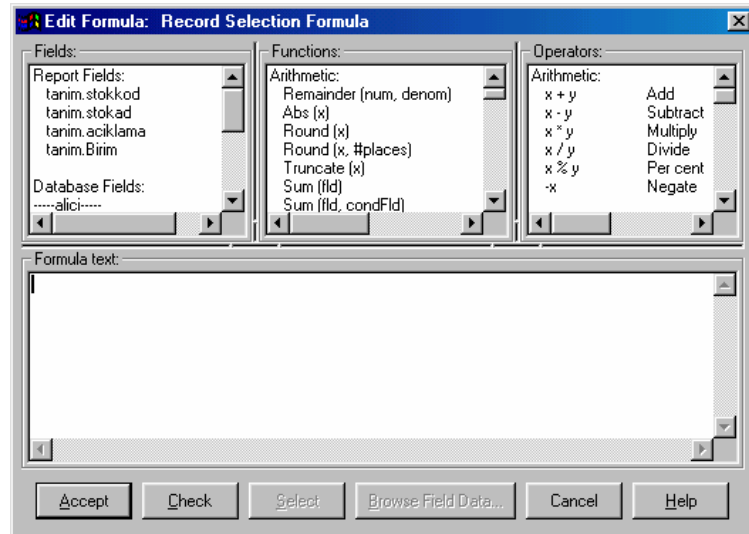
Log Off Server:Bir SQL Server database'ini kullanım disi birakir.

Show SQL Query:Kullanimda olan SQL server Database'ine bir SQL query göndermek için kullanılır.



6-REPORT

Select Records Expert:Seçili field'ların istenen kısımlarını getirmek için kullanılır.



Edit Record Selection Formula:Kayıtların hepsi istenmiyorsa istenen kısmı için seçim kriteri yaratmak için kullanılır.

Record Sort Order:Kayıtların hangi field'a göre a'dan z'ye veya z'den a'ya sıralanacağı işlemidir.

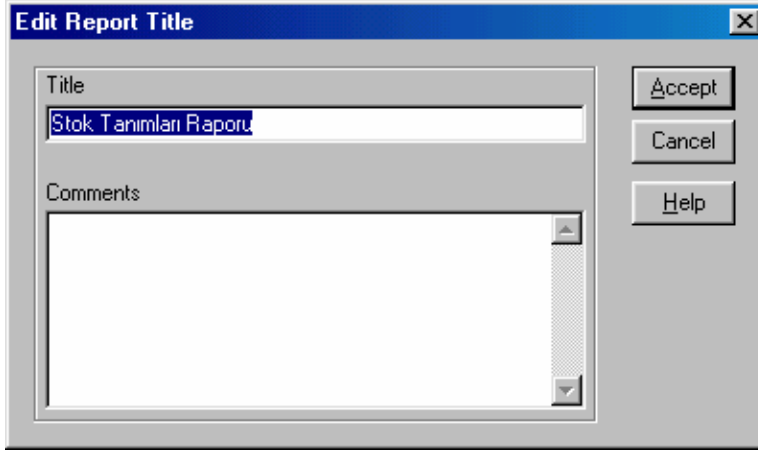
Select Groups:Select Records'daki işlemi grup için uygular.

Edit Group Selection Formula:Edit Record Selection Formula'daki işlemi grup için uygular.

Group Sort Order:Record Sort Order'daki işlemi grup için uygular.

Refresh Report Data:Database'e tekrar sorgu göndererek verilerin tazelmesini sağlar.

Save Data With Closed Report:Seçimlik bir komuttur.Seçili olduğunda program çıkışında otomatik olarak rapor datası kaybolur.Seçili olmadığında data kaybolmayarak,rapor tekrara açıldığında database'den gelir.



Report Title:Rapor başlığını girmek için kullanılır.

Set Print Date:Rapor baskı tarihini ayarlamak için kullanılır.

7-WINDOW

Tile Vertically:Pencereleri dikey dösemek için kullanılır.

Tile Horizontally:Pencereleri yatay dösemek için kullanılır.

Cascade:Pencereleri basamaklamak için kullanılır.

Arrange Icons:İkon halindeki pencereleri düzenlemek için kullanılır.

Close All:Tüm pencereleri kapatmak için kullanılır.

8-HELP

Contents:Help konularını ekrana getirir.

Search:Bir help konusunu aramak için kullanılır.

Using Help:Help in nasıl kullanılacağını anlatır.

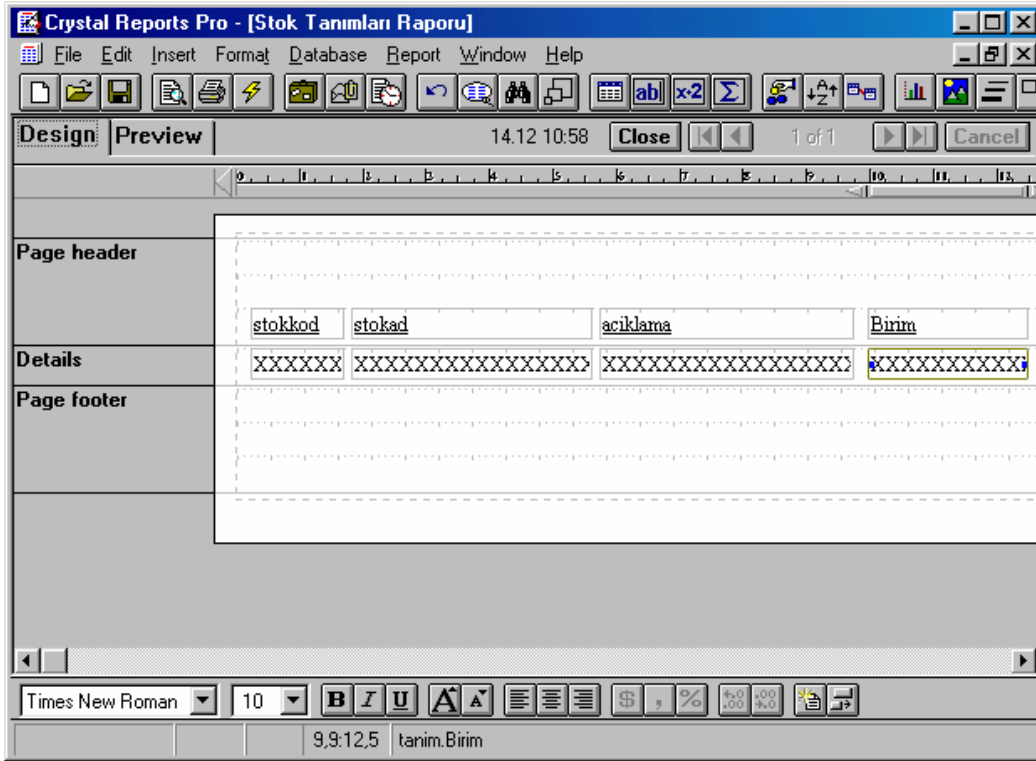
Register/Chance Adress:Kisisel kayıt adresi için kullanılır.

Technical Support Requeset:Teknik destek istemek için istek formu doldurma ekranını getirir.

About Crystal Reports for Visual Basic:Crystal Reports hakkında sürüm,üretici,firma,v.b.bilgileri verir.

CRYSTAL REPORTS İLE ÖRNEK RAPOR HAZIRLAMA

Bir rapora başlamadan önce raporun kaynak database ve table'leri gözden geçirilerek rapor hazırlığı yapılır.yeni bir rapor için File menüsünden New Report komutu verilir. Daha sonra gelen ekranda hangi database'in kaynak olarak kullanacağı belirtilir. Biz bu örnekte stok.mdb adlı bir access database ve bu database'e ait tanım adlı bir table'i kullandık. Kullanılacak table'i üzerine çift tiklandığında table'a ait field'lar listesi gelecektir. Burada kullanılmak istenen field'lar sürükleyip bırak yöntemi ile rapor sayfasını details kısmında istenen yere bırakılarak rapora konulmuş olur.Rapor sayfasında başlangıçta 3 ana grup vardır: Report Header,Details,Report Footer.Report Header, sayfanın başına 1 kere yazılacak sabit bilgileri içerir. Örneğin field başlıkları gibi.Report Footer da içerik olarak report Header ile aynıdır,farkı,bilgileri sayfa sonuna yazmasıdır.Details kısmı ise, dataların konulduğu içeriği değişen ve data geldikçe genişleyen bir kısımdır. Database field'lar bu kısma konulur. Tanım table'ında 4 field bulunmakta ve bunların hepsi raporda yerini almaktadır. Gereklik halinde daha önce bahsedilen komutlarla istenen field'la ilgili olarak bir grup kısmı yaratılabilir. Field'lar detail'e konulduktan sonra Format menüsünden istenilen sekile getirilebilir. Baskı önizleme için File menüsünden Print Preview komutu verilir. Yapılan raporu kaydetmek için ise File menüsünden Save komutu kullanılır. Crystal Reports'a yapılan bir raporu Visual Basic'de kullanmak için Custom Controls'den Crystal Report nesnesi VB toolbar'ına eklenir.Bu nesne aracılığıyla rapor çağrılır.



DOSYA İSLEMLERİ

Bu bölümde dosyala kayıt yapma, yapılan bu kayıtları düzeltme, silme yazdırma gibi konuları inceleyeceğiz. Bunu bir örnekle açıklayabiliriz. Örneğin bu, kütüphane kayıtlarını tutan bir program olabilir. Programda her bir işlem için ayrı bir form açıp, daha sonra bunları birleştirelim.

Dosyaya kayıt yazma

Bir kayıta kitap numarası, kitap adı, raf numarası, konu, yazar ismi ve yayın evi bilgileri olsun. Bu bilgilerin her birine alan denir. Form1 de her alan için Label ve textbox lar command buttonlar şeklindeki gibi hazırlanır. label'ların her birinin, properties penceresindeki caption alanlarının her biri yazılır. Textbox'ların ise yine properties penceresindeki text'lerin içi boşaltılır.

The image shows a Visual Basic form titled "KAYIT". It contains six text boxes for data entry, each with a label to its left: "KİTAP NUMARASI", "KİTAP ADI", "RAF NUMARASI", "KONU", "YAZAR ADI", and "YAYIN EVİ". Below the text boxes, there are three buttons: "KAYDET", "VAZGEÇ", and "ÇIKIŞ". The form has a standard Windows 95/98 style with a title bar and window controls.

Formdaki textbox'lara girilen bilgilerin her seferinde hard diske yazılması için, bilgiler random tipli dosya açıldıktan sonra Field deyimi ile belirlenip, Record tipli değişken dosyaya Put deyimi ile yazdırılır.

Bir visual basic projesinde bas uzantılı Modüle dosyasında bulunabilir. Bu dosya program kodları içerir. Projeye modül dosyası eklemek için, Project menüsündeki Add Modüle komutu kullanılır. Ekran Add Modüle dialog kutusu gelir. Burada modüle seçeneği seçilip, aç tiklanırsa ekrana aşağıdaki gibi bir pencere gelir.

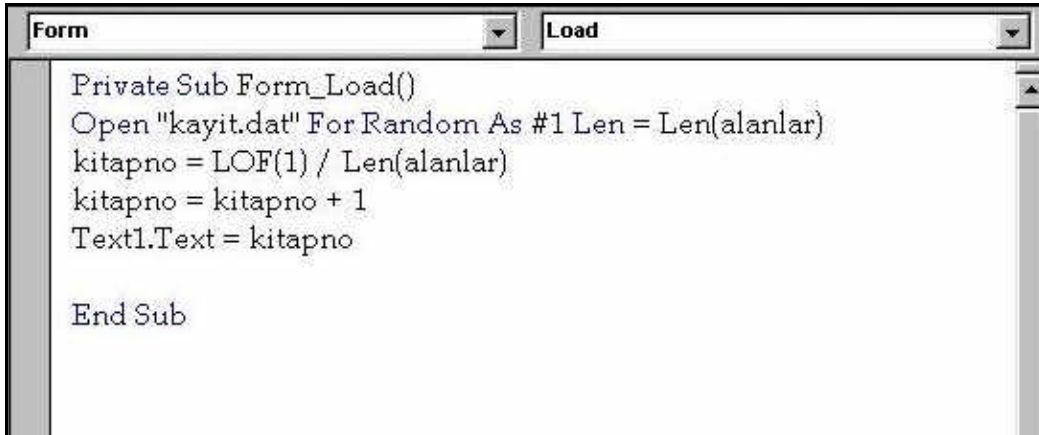
The image shows the "Declarations" window in Visual Basic. It has two tabs: "(General)" and "(Declarations)". The "(Declarations)" tab is selected, displaying the following code:

```

Type kayitbilgisi
rafno As String * 20
kadi As String * 30
kno As String * 5000
konu As String * 30
yazar As String * 30
yevi As String * 30
End Type
Global alanlar As kayitbilgisi

```

Bu pencerede Object liste kutusunda General, Prog liste kutusunda Declarations seçeneği seçili iken gerekli tanımlamalar yapılır. Tanımlama satırları Type deyimi ile başlar, End type deyimi ile biter. Değişkenler tanımlanırken uzunlukları da belirtilir. Bu tanımların projenin diğer form ve modüllerinde kullanılması için Global deyimi ile tanımlanması gerekir. Burada kayıt tipli değişken adı "kayitbilgisi", global özellikli değişken adı ise "alanlar" olarak seçilmiştir. Burada tekrar formun kod sayfasına dönelim. Kod sayfasında ilk yapılacak iş bilgi kaydedilecek olan dosyayı Open deyimi ile açmaktır.

The image shows a screenshot of the Visual Basic code editor. At the top, there are two dropdown menus: the first is labeled 'Form' and the second is labeled 'Load'. Below these, the code for the 'Form_Load' event procedure is displayed. The code is as follows:

```
Private Sub Form_Load()  
Open "kayit.dat" For Random As #1 Len = Len(alanlar)  
kitapno = LOF(1) / Len(alanlar)  
kitapno = kitapno + 1  
Text1.Text = kitapno  
  
End Sub
```

Sekilde gördüğümüz gibi open deyimi ile açılan dosyaya "kayit.dat" ismi veriliyor. Takibinde "for random" ile dosyanın rastgele erişimli olduğu dosyanın numarası ve byte olarak tuttuğu yer belirtiliyor. Dosya numarası önüne "#" karakteri konularak yazılır. Ayrıca dosya numarası proje içinde dosyanın ismini temsil eden sayısal değerdir.

Random dosyalara kayıt yazarken her kayda bir kayıt numarası verilir. O ana kadar dosyada kaç kayıt olduğunu bulmak için, dosyanın byte olarak uzunluğu, bir kayıt uzunluğuna bölünür. Her kayıt işleminden sonra, kitap numarası bir artırıldıktan sonra text1 e yazdırılır. Yukardaki şekilde formun aktive olduğunda yazılmış ilk satır, text1 e kitap numarası yazıldıktan sonra ikinci satırda kursorün zaten yazılmış olan kitap numarası isimli textbox a değil, "kitapadi" isimli ikinci textbox a gitmesini sağlar.

KAYIT

KITAP NUMARASI

KITAP ADI

RAF NUMARASI

KONU

YAZAR ADI

YAYIN EVI

KAYDET VAZGEÇ ÇIKIŞ

Form kapatılırken open deyimi ile açılan dosyanın da kapatılması gerekir. Formun unload kısmına close #1 şeklinde yazılarak kapatılır.

```

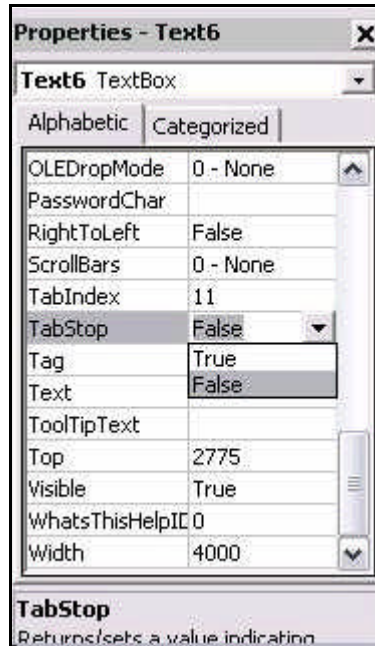
(General) (Declarations)

Private Sub KAYDET_Click()
    alanlar.kadi = Text2.Text
    alanlar.rafno = Text3.Text
    alanlar.konu = Text4.Text
    alanlar.yazar = Text5.Text
    alanlar.yevi = Text6.Text
    cevap = MsgBox("kayit yapilsinmi", 4)
    If cevap = 6 Then
        Put #1, kitapno, alanlar
        kitapno = kitapno + 1
        Text1.Text = kitapno
    End If
End Sub

```

Kodların ilk satırlarında textbox a girilen bilgiler teker teker alanlara aktarılıyor. Kaydı dosyaya yazdırmadan önce msgbox ile ekrana onay kutusu çıkartırız. Cevap degiskenine kullanıcı evet düğmesine tıklama yaparsa 6, hayir düğmesine tıklama yaparsa 7 değeri aktarilir. Bundan sonra if kontrolü ile evet tiklandiginda yapılacak işlemler sıralanıyor.önce put deyimiile alanlar içeriği dosyaya yazdırılıyor. Kitap no bir artırılıyor. Bu işlemden sonra textbox larin içeriğine teker teker bosluk değeri aktarilir.

Bu formda ilk textbox'a arayacağımız kitap numarasını girdikten sonra girilen kitap numarasına ait bilgiler, diğer textbox'larda görüntülenecek. Yani bunlara dışardan bilgi girişi olmayacak bunun için textbox'ların tabstop özelliği false olmalı.



Dosyadan kayıt okuma

Program çalıştığında, Text1 e kitap numarası yazılıp, Arama düğmesine basıldığında yazılan kitap numarasına ait bilgileri görüntüleyecek. bunun için kod sayfasında arama click yordamında GET deyimi ile, girilen kitap numarasına ait bilgileri dosyadan okutmak gerekir.

```

Form Load

Private Sub ARAMA_Click()
    kitapno = Text1.Text
    Get #1, kitapno, alanlar
    Text2.Text = alanlar.kadi
    Text3.Text = alanlar.rafno
    Text4.Text = alanlar.konu
    Text5.Text = alanlar.yazar
    Text6.Text = alanlar.yevi
End Sub

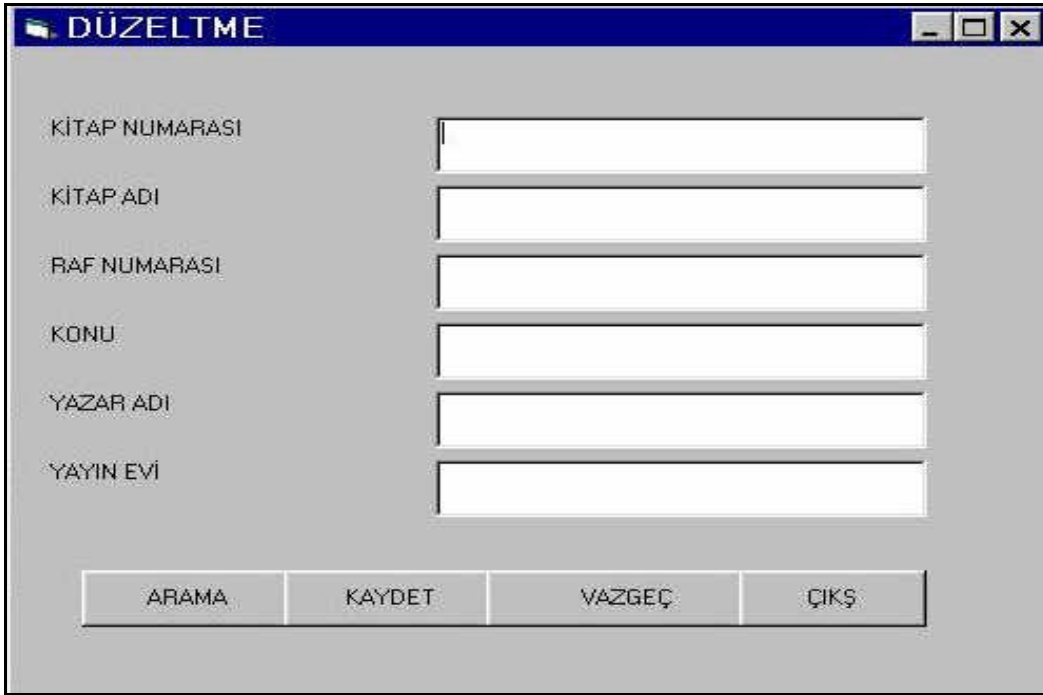
```

Formun load olayına;
open "kayit.dat" for random as #1 len=len (alanlar) yaz

Kayit düzeltme

Kitap numarası istendikten sonra dosyadan okutup ekrana yazdıracağız. Fakat kullanıcı bu sefer textbox'lara müdahale edebilecek.

Formdaki arama düğmesinin işlevi, kitap numarası girildikten sonra, tıklandığında kitap numarasına ait bilgilerin ekrana yazılmasını sağlamaktır. Form üzerinde istenilen değişiklikler yapıldıktan sonra kaydet düğmesi ile dosyaya yazdırılır.



The screenshot shows a Visual Basic form titled "DÜZELTME". The form has a light gray background and a blue title bar. It contains six text boxes for input, each preceded by a label: "KİTAP NUMARASI", "KİTAP ADI", "RAF NUMARASI", "KONU", "YAZAR ADI", and "YAYIN EVİ". At the bottom, there are four buttons: "ARAMA", "KAYDET", "VAZGEÇ", and "ÇIKIŞ".

Dosyadan kayıt silme

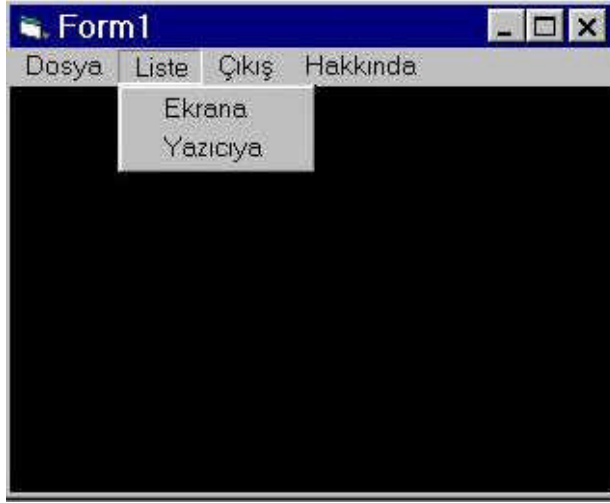
İlk olarak kullanıcıdan kitap numarası istenir ve arama mantığıyla görüntülenir. Formu silmek için textbox'ları boşluklara eşitleyip alanlara atarız.

Menü hazırlama

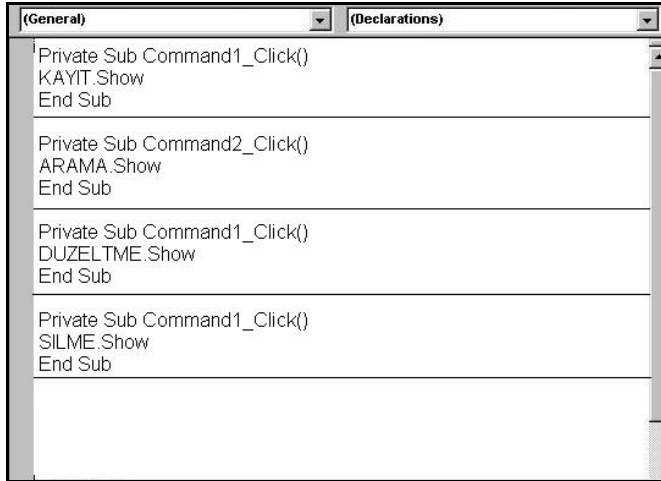
Program çalıştığında hazırladığımız formları düzenli bir sıra ile gönderilmek için, bir menü hazırlamamız gerekiyor. Bunu yeni bir form üzerine her form için düğme koyarak yapabileceğimiz gibi, mdi form oluşturmaya da yapabiliriz. MDIForm, project menüsünün ADD MDIForm seçeneğinden açılır. Ekrana ADD MDIForm dialog kutusu gelir.

Microsoft Visual Basic 6.0

Menüleri girdikten sonra ok deriz ve mdiform'un ekran görüntüsü aşağıdaki gibidir.



Alt menüleri hazırlarken, next'en önce sol ok tusuna basılmalı. Bu ok ile kayıt artık dosya menüsünün bir alt menüsü. Eğer hazırlayacağımız formlar başka bir proje içinde olsaydı, onları bu forma dahil etmek için, Project menüsünün Add File seçeneğinden yararlanacaktık. Şu halde ekleme işini formun kod sayfasına yazacağız.



```
Private Sub form_Activate()  
If text1.text = 0  
Label11.left = 9480  
End if  
End Sub  
  
Private Sub form_Unload(Cancel As Integer)  
Unload CARI  
Unload CRAPOR  
Unload KUR  
Unload SRAPOR  
Unload STOK  
Unload THSTDY  
End Sub  
  
Private Sub Sscommand1_Click()  
Text1.text=1  
CARI.Show  
End Sub  
  
Private Sub Sscommand10_Click()  
Unload Me  
End Sub  
  
Private Sub Sscommand2_Click()  
Text1.text=1  
Stok.Show  
End Sub  
  
Private Sub Sscommand3_Click()  
End Sub  
  
Private Sub Sscommand4_Click()  
Text1.text=1  
SRAPOR.Show  
End Sub  
  
Private Sub Sscommand5_Click()  
Text1.text=1  
KUR.Show  
End Sub  
  
Private Sub Sscommand6_Click()  
Text1.text=1  
THSTDY.Show  
End Sub  
  
Private Sub Sscommand7_Click()  
Text1.text=1
```

Microsoft Visual Basic 6.0

```
CRAPOR.Show
End Sub
Private Sub Sscommand8_Click()
Text1.text=1
HAKKINDA.Show
End Sub

Private Sub Text1_Change()
If Text1.Text = 1 then
Labell11.Left = 7440 : BEEP
End Sub

Private Sub Timer1_Timer()
If Text1.Text = 0 Then
Labell11.Left = Text2.Text
Text2.Text = Val(Text2.Text) - 120
If Text2.Text = 7320 Then
Text1.Text = 1
Timer1.Enable = False
End If
Enf If
End Sub
```

CARI.FRM

```
Private Sub Combol_Change()
Text5.Set Focus
End Sub

Private Sub Combol_Keypress(KeyAscii As Integer)
If keyascii = 13 Then
Text5.Setfocus
End Sub

Private Sub Combol_Click()
Unload Me
End Sub

Private Sub Combo2_Click()
On Error Goto 50
If Text1.Text = "" Or Text1.Text = " " Or Text2.Text = "" Or
Text2.Text = " " Then
Msgbox "Ürün kodu adini tam olarak girmelisiniz.",vbOKonly,"Text
ile For Windows" : Goto 100
If Text3.Text = "" Then Text3.Text = " "
If Text4.Text = "" Then TAR$ = DATE$
GUN$ = Mid$(TAR$, 4, 2)
```

```

AY$ = Left$(TAR$, 2)
Yil$ = Right$(TAR$, 4)
Text4.Text = GUN$ + "/" + AY$ + "/" + YIL$
End If
If Text5.Text = "" Then Text8.Text = " "
If Text6.Text = "" Then Text9.Text = " "
If Text7.Text = "" Then Text10.Text = " "
If Combol.Text = "" Then Combol.Text = "TL"

```

```

Datal.Recordset.Edit
Datal.Recordset("CariKodu") = Text1.Text
Datal.Recordset("Cariadi") = Text2.Text
Datal.Recordset("ilgili") = Text3.Text
Datal.Recordset("Acilistarihi") = Text4.Text
Datal.Recordset("kuru") = Combol.Text
Datal.Recordset("adresi") = Text5.Text
Datal.Recordset("ili") = Text6.Text
Datal.Recordset("VDVN")= Text7.Text
Datal.Recordset("Tel1")=MaskedTextBox1.Text
Datal.Recordset("Tel2")=MaskedTextBox2.Text
Datal.Recordset("Faks") = MaskedTextBox3.Text
Datal.Recordset.Update
Datal.Refresh
List1.RemoveItem(Text200.Text)
List1.AddItem (Text1.Text & String(15 -
Len(Text1.Text), "_")&Text2.Text)

```

```

Private Sub Command3_Click
On Error Goto 50
Cevap=Msgbox("Bilileriniz silinsinmi",52,"Texttile For Windows")
If cevap = 6 Then
Datal.Recordset.Delete
Datal.Recordset.MoveFirst
a% =List1.Listcount - 1
For i% 0 to a%
List1.Selected(i%) = True
20
Next i%

```

```

Private Sub Command4_Click
On Error Goto 1000
For i% = 0 To List1.Listcount -1
If Text1.Text & String(10 - Len(Text1.Text), "_") = Rtrim
(Left(List1.Text,10))Then
Msgbox "Ayni kodda baska bir kayit tespit edildi",vbOKOnly
Text1.Setfocus
Goto 1001
End If
Next i%

```

Microsoft Visual Basic 6.0

```
If Text1.Text = "" Or Text1.Text = " " Or Text2.Text = "" Or  
Text2.Text = " " Then  
Msgbox "Ürün kodu adini tam olarak girmelisiniz.",vbOKonly,"Text  
ile For Windows" : Goto 1000  
If Text3.Text = "" Then Text3.Text = " "  
If Text4.Text = "" Then TAR$ = DATE$  
GUN$ = Mid$(TAR$, 4, 2)  
AY$ = Left$(TAR$, 2)  
Yil$ = Right$(TAR$, 4)  
Zaman% = Asc("2")*Asc("(")  
If yil$ = Zaman% Then Error Number = 1 : Goto 1000  
Text4.Text = GUN$ + "/" + AY$ + "/" + YIL$  
End If  
If Text5.Text = "" Then Text8.Text = " "  
If Text6.Text = "" Then Text9.Text = " "  
If Text7.Text = "" Then Text10.Text = " "  
If Combol.Text = "" Then Combol.Text = "TL"  
  
Data1.Recordset.AddNew  
Data1.Recordset("CariKodu") = Text1.Text  
Data1.Recordset("Cariadi") = Text2.Text  
Data1.Recordset("ilgili") = Text3.Text  
Data1.Recordset("Acilistarihi") = Text4.Text  
Data1.Recordset("kuru") = Combol.Text  
Data1.Recordset("adresi") = Text5.Text  
Data1.Recordset("ili") = Text6.Text  
Data1.Recordset("VDVN")= Text7.Text  
Data1.Recordset("Tel1")=MaskedTextBox1.Text  
Data1.Recordset("Tel2")=MaskedTextBox2.Text  
Data1.Recordset("Faks") = MaskedTextBox3.Text  
Data1.Recordset("ToplamBorc") = "0"  
Data1.Recordset("ToplamAlacak") = "0"  
Data1.Recordset.Update  
Data1.Refresh  
  
List1.AddItem (Text1.Text & String(15 -  
Len(Text1.Text), "_")&Text2.Text)  
  
Private Sub Command6_Click  
Text1.Text = ""  
Text2.Text = ""  
Text3.Text = ""  
Text4.Text = ""  
Combol.Text = "TL"  
Text5.Text = ""  
Text6.Text = ""  
Text7.Text = ""  
MaskedTextBox1.Text = "(____) ____ _ _"
```



```

MaskedTextBox2.Text = "(____) ____ _ _"
MaskedTextBox3.Text = "(____) ____ _ _"

Text1.Setfocus
End Sub

Private Sub Form_Activate()

Anamenu.Hide
Data1.Recordset.MoveFirst

List1.AddItem (Data1.Recordset("Carikodu") & String(15 - Len
(Data1.Recordset("caricodu")), "_") & (Data1.Recordset("cariadi")))
10:
Data1.Recordset.MoveNext
If Data1.Recordset.EOF Then Goto 20

List1.AddItem (Data1.Recordset("Carikodu") & String(15 - Len
(Data1.Recordset("caricodu")), "_") & (Data1.Recordset("cariadi")))

If Data1.Recordset.EOF Then Goto 20 Else Goto 10
End Sub

Private Sub Text3_Keypress(KeyAscii AS Integer)
If KeyAscii = 13 Then

TAR$ = DATE$
GUN$ = Mid$(TAR$, 4, 2)
AY$ = Left$(TAR$, 2)
Yil$ = Right$(TAR$, 4)
Zaman% = Asc("2")*Asc("(")
If yil$ = Zaman% Then End If
Text4.Text = Gun$ + "/" + ay$ + "/" + yil$
Text4.Setfocus
End If
End Sub

Private Sub Text3_Keypress(KeyAscii AS Integer)

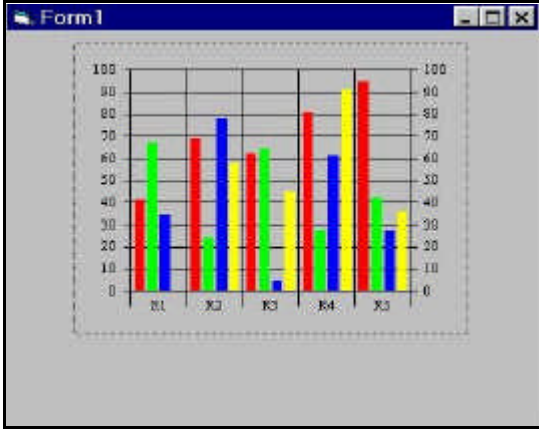
TAR$ = DATE$
GUN$ = Mid$(TAR$, 4, 2)
AY$ = Left$(TAR$, 2)
Yil$ = Right$(TAR$, 4)
Zaman% = Asc("2")*Asc("(")
If yil$ = Zaman% Then Err.Number = 1 : End
Text4.Text = Gun$ + "/" + ay$ + "/" + yil$
Text4.Setfocus

End Sub

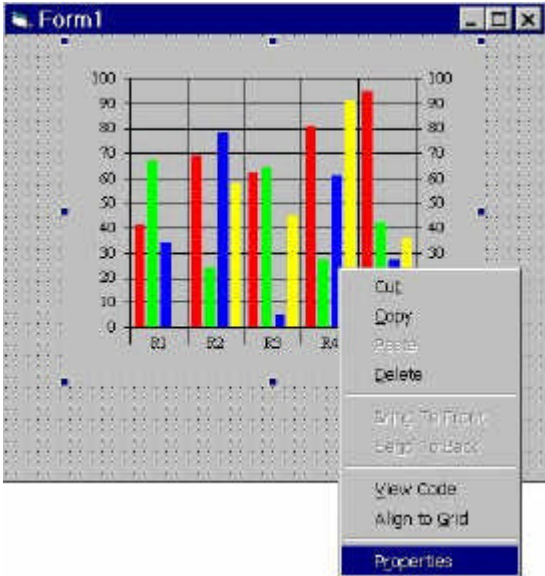
```

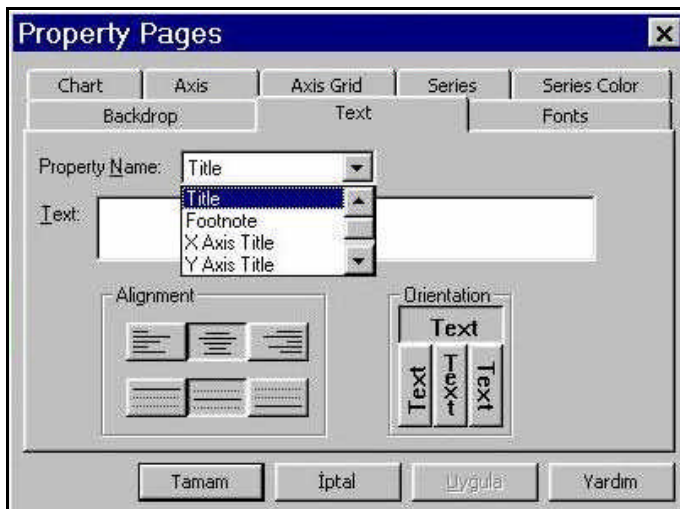
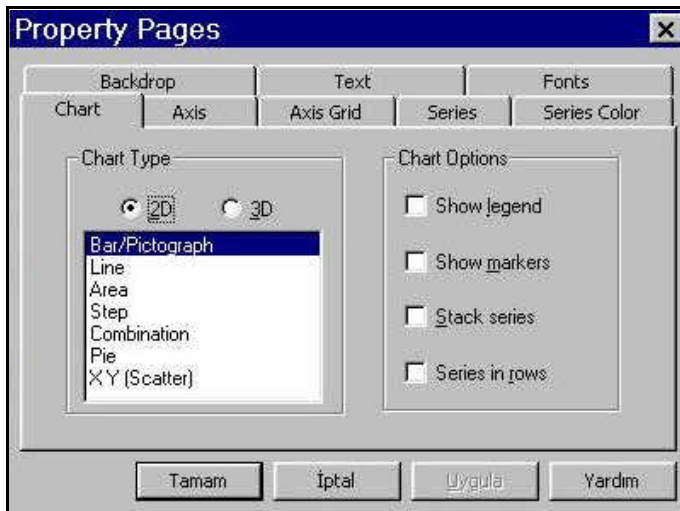
12-VB'de GRAFIK KULLANIMI

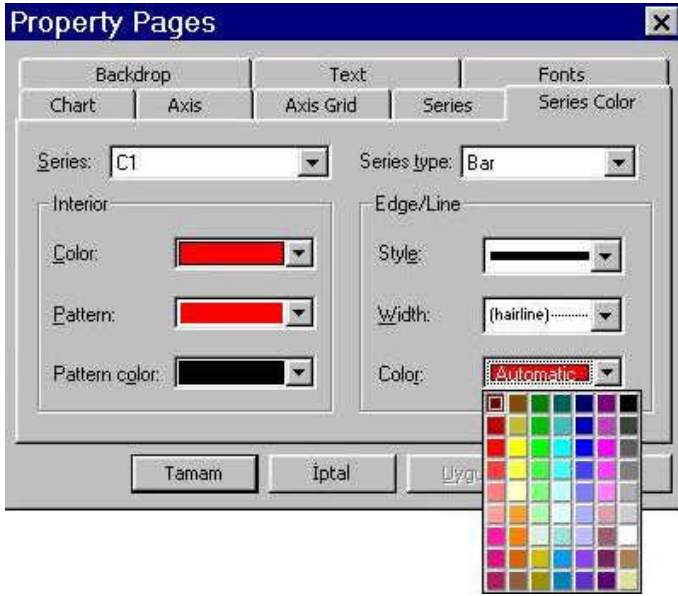
Bu kontrol elamani kullanarak veri grafikleri çizilebilir. Keza kopyalama ve yapistirma kaydetme ve yazdirma gibi islemler de gerceklestirilebilir. Verilerin grafiksel olarak çizimi çok sayuda grafik sitili tarafından desteklenir.



Grafik arayüzünü VB de Chart kontrolü salar. Chart kontrolünün bir çok özelligini tasarim aninda görerek degistirmemiz mümkündür. Bunun için form üzerindeki Chart kontrolüne sag tusla tiklanir, açilan pencereden Properties komutu seçilir. Böylece asagidaki açilan pencerelerden istenildigi gibi özellikler degistirilebilir.







Bu kontrolü kullanabilmek için Project-Components menüleri ile açılan pencereden "Microsoft Char Control" seçeneği işaretlenir veya Browse düğmesi ile MSCHRT20.OCX dosyası bulunup projeye eklenir.

Chart kontrolü ile bir grafik oluşturmak için yapılması gereken işlemler şöyle sıralanabilir.

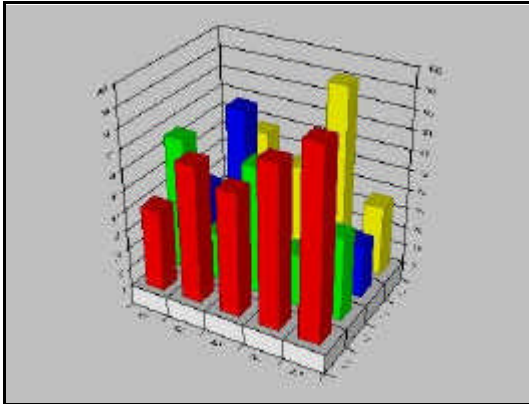
- ChartType ile grafiğin tipini belirlemek.
- Kontrol üzerinde kaç ayrı grafiğin gösterileceğini ColumnCount ile belirlemek.
- Her bir grafiğe ait kaç veri olduğunu RowCount özelliği ile belirlemek.
- Data özelliği ile her bir noktanın değerini belirlemek.

Chart kontrolü ile bir grafiğin oluşturulmasını bu adımlarla özetleyebiliriz.

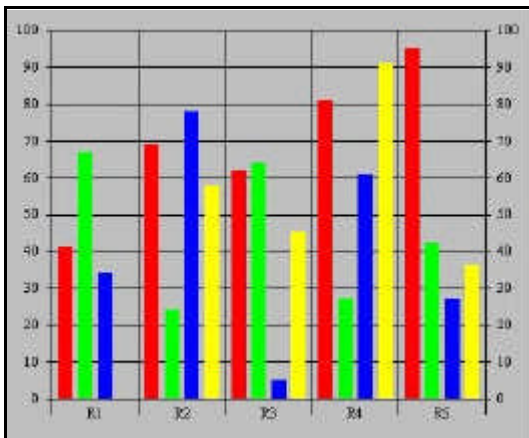
Chart Özellikleri (Properties)

ChartType

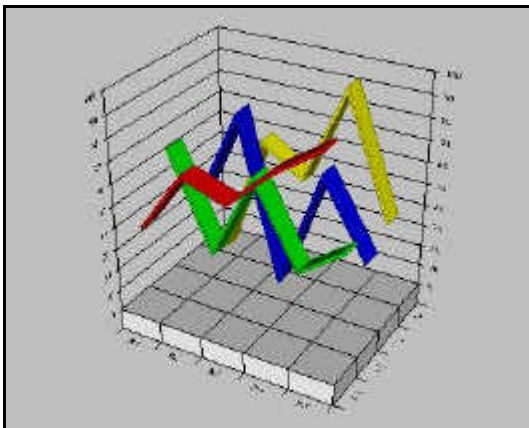
Grafik tipini belirlemek için bu özellik kullanılır. Alacağı değerler ve isimleri aşağıda belirtilmiştir.



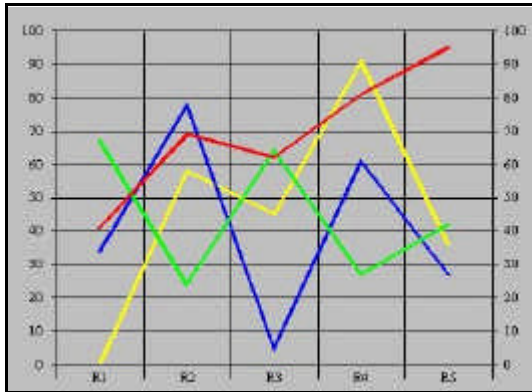
Charttype=0,VtVhChartType2dBar



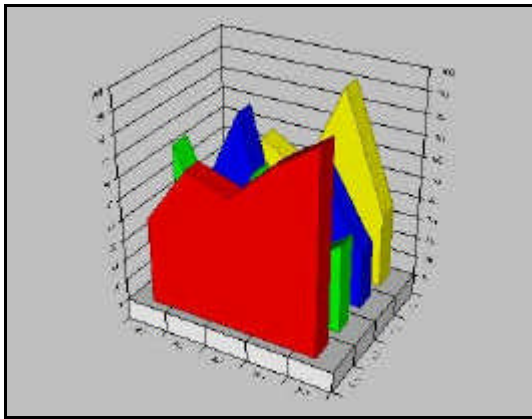
Charttype=1,VtVhChartType2dBar



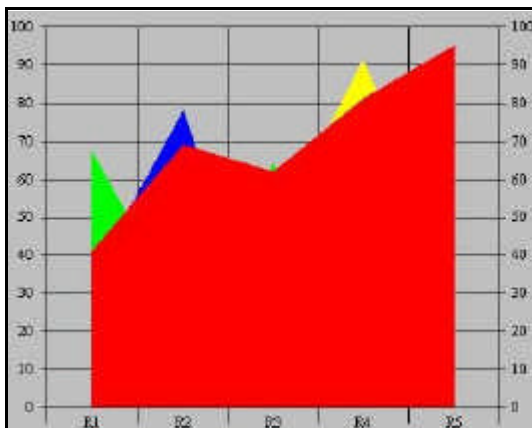
Charttype=2,VtVhChartType3dLine



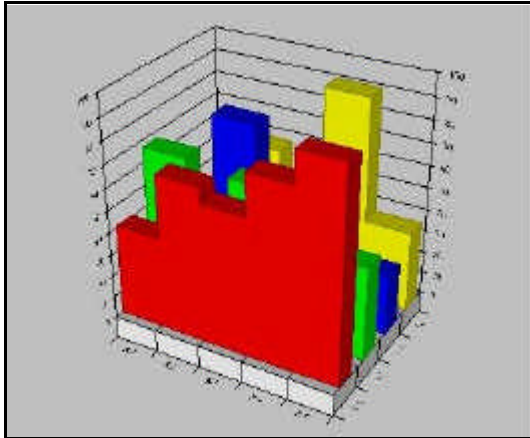
Charttype=3,VtVhChartType2dLine



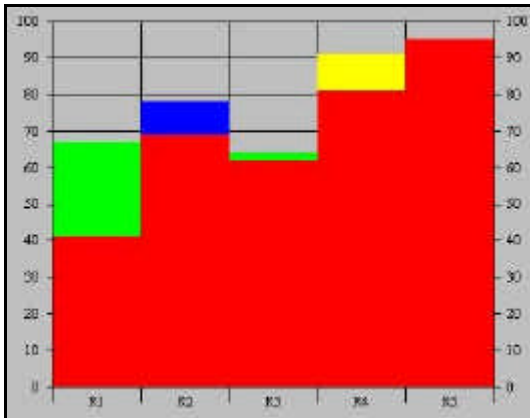
Charttype=4,VtVhChartType3dArea



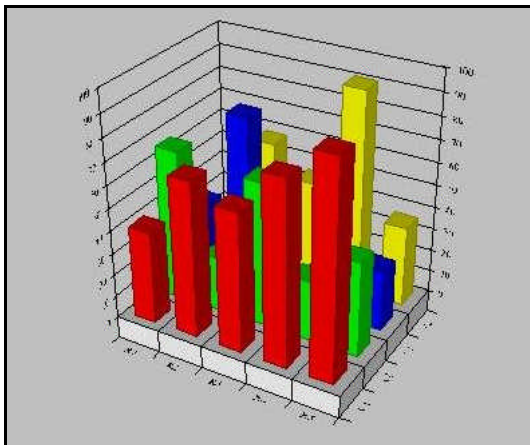
Charttype=5,VtVhChartType2dArea



Charttype=6,VtVhChartType3dStep

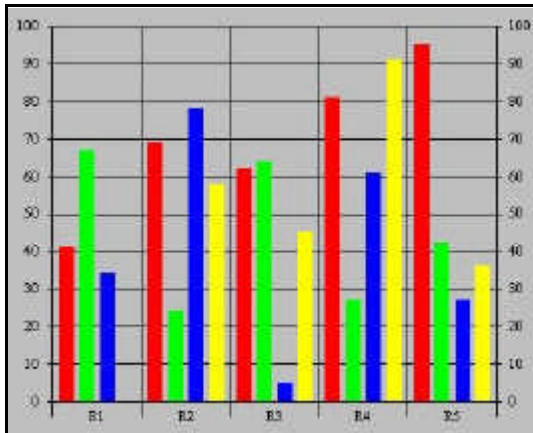


Charttype=7,VtVhChartType2dStep

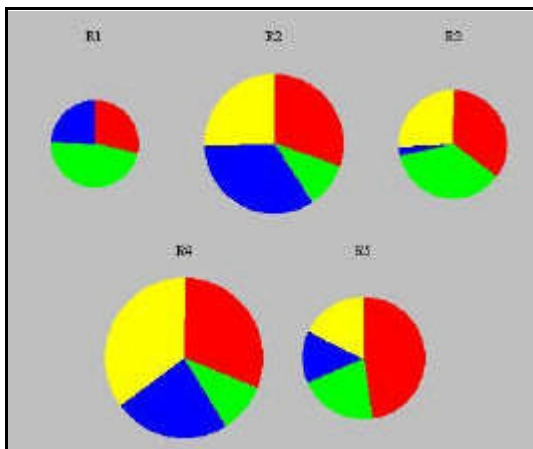


Charttype=8,VtVhChartTypeCombination

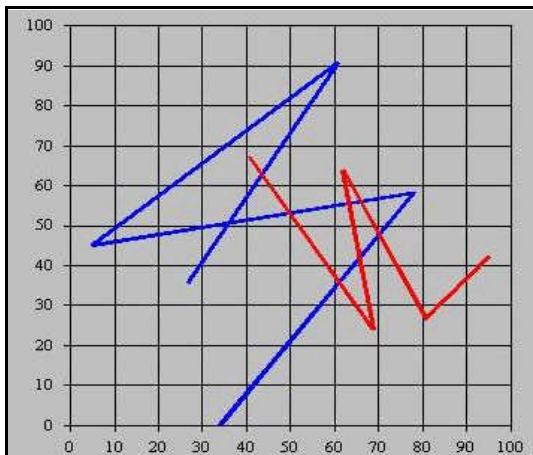
Microsoft Visual Basic 6.0



Charttype=9,VtVhChartType2dCombination



Charttype=14,VtVhChartType2dPie



Charttype=16,VtVhChartType2dXY

ColumnCount

Birkaç farklı grafik aynı anda bir grafik kontrolü üzerinde gösterilebilir. Bu özelliğe verilecek değer ile kaç türlü grafiğin gösterileceği belirtilebilir. Bu özellik ile belirlenen her grafik için RowCount özelliği ile belirlenen sayıda ayrı değerlerin atanması gerekir.

RowCount

Her bir grafiğin nokta sayısı bu özellik aracılığıyla belirlenir. Bu özelliğe yapılan atama ile veri eksenini okadar parçaya bölünür.

Row,Column

Üzerinde işlem yapılacak nokta bu özellik ile belirlenir.

Data

Row ve Column özellikleri ile belirlenen noktanın değeri bu özellik ile belirlenir.

ChartData

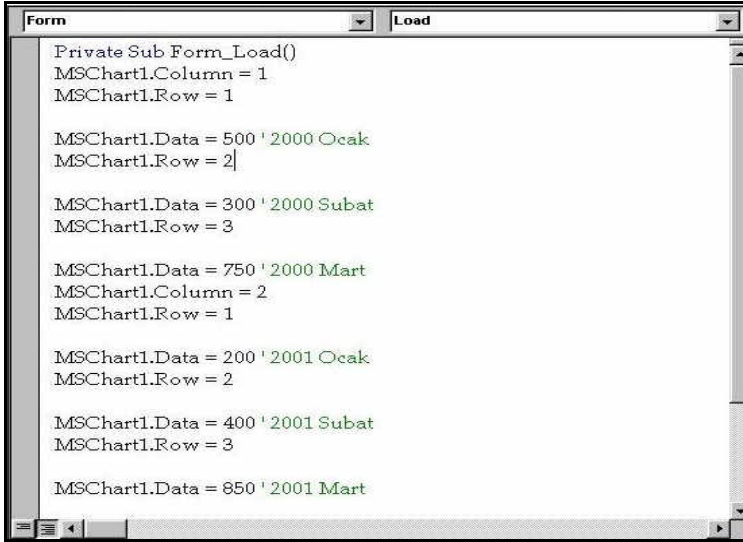
Grafik üzerinde bir noktanın değerini belirlemek için Data özelliğini kullanıyorduk.

Ayrıca yukarıdaki örnekte gördüğümüz gibi toplu değer atamada data çok etkili bir yöntem değildir. Bunun yerini ChartData daha avantajlıdır.

Tanımlanacak dizi iki boyutlu olmalıdır. Dizinin ikinci boyutu hangi grafik setinin kullanılacağı, birinci boyutu grafik üzerindeki veriyi gösterir.

Örneğimizde iki farklı yıla ait 3 ayın grafiğini çizdire biliyorduk. Budurumda tanımlayacağımız dizi x(3,2) olmalıdır. Yani üç noktadan oluşan iki farklı grafik

Ayrıca VB'de diziler ilk elmandan başlamaktadırlar. Chart kontrolünde ise Column ve Row numarası birden başlar. Bu yüzden diziyi x(1 To 3,1 To 2) şeklinde olmalıdır.



ColumnLabel, RowLabel

Grafikte her verinin neyi temsil ettiğini belirten metin bu özelliklerle belirlenir.

Column veya Row özelliği ile değiştirilmek istenen nokta seçildikten sonra bu iki özellikle o iki noktanın özellikleri belirtilebilir.

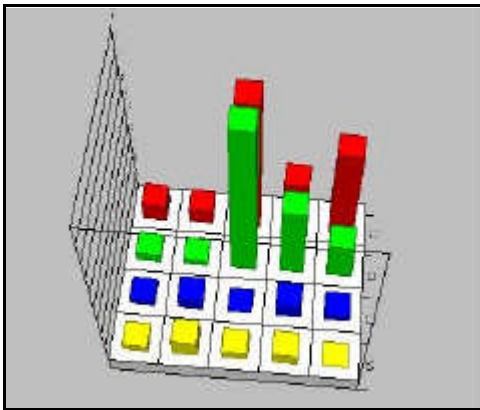
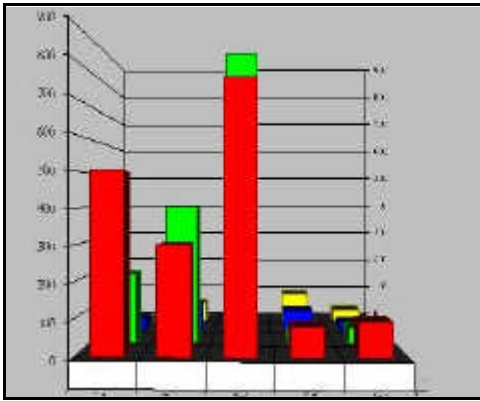
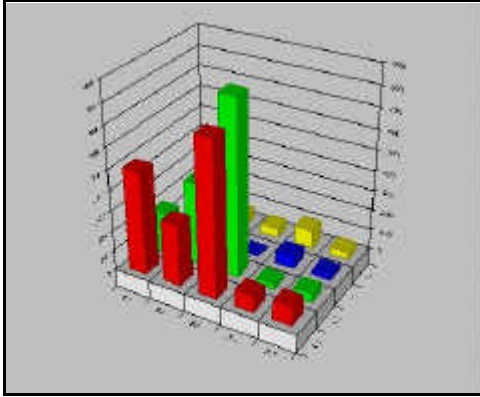
ShowLegend

Kart üzerinde birden fazla grafik gösterilecekse, hangi rengin hangi grafiğe ait olduğunu belirten göstergelerin gösterilmesi için bu özellik true yapılmalıdır.

AllowDynamicRotation

Bu özelliğin default değeri true'dir hiçbir koda gerek kalmadan 3 boyutlu grafikleri döndürebilir. Bu özellik false yapılırsa döndürme işlemi için kullanıcı ekstradan kodlar yazması gerekir.

Bu özellik true iken kullanıcı ctrl tusunu basılı tutarak grafiği fare yardımıyla döndürebilir.



AutoIncrement

Data özelliği ile grafik üzerindeki noktaların değerlerini belirlerken önce Column ve Row özelliği ile hangi noktanın değerini vereceğimizi belirtiyorduk. İstenirse bu özelliği true verilerek her data atamasından sonra yeni noktayı belirlemeye gerek kalmadan bir sonraki noktaya geçmesini sağlayabiliriz.

TitleText, FootnoteText

Grafigin üstüne ve altına yazilmasi istenen metinler bu iki özellikle belirlenir.

Title, Footnote

Grafigin üstüne ve altına yazilacak yazilari Titletext ve FootnoteText özellikleri ile belirliyebiliyorduk. Bunlara ait özellikleri bu iki özelliğin alt özelliği ile belirtiyoruz.

VtFont

Metnin yazı tipi bu özellikle belirlenir

```
MsChart1.Title.VtFont.Name="Times New Roman Tur"
```

Location

Metnin yeri bu özelliğin alt özellikleri sayesinde belirtilir.

Location.Visible

Bu özelliğe False verilerek metin gizlenebilir.

Location.LocationType

Bu özelliğe aşağıdaki değerlerden biri verilerek metnin yazılacağı konum belirlenir.

VtChLocationTypeTop : Üst

VtChLocationTypeTopLeft : Sol üst

VtChLocationTypeTopRight : Sağ üst

VtChLocationTypeLeft : Sol

VtChLocationTypeRight : Sağ

VtChLocationTypeBottom : Alt

VtChLocationTypeBottomLeft : Alt sol

VtChLocationTypeBottomRight : Alt sağ

VtChLocationTypeCustom : Rect özelliği ile belirlenen koordinat

TextLayout

Metnin yerleşimi bu özelliğin alt özelliği ile belirlenir.

TextLayout.Orientation

Metnin yatay mı yoksa dikey mi olacağı bu özellik ile belirlenir. Alacağı değerler ve anlamları şunlardır.

VtOrientationHorizontal : Metin yatay olarak yazılır.

VtOrientationVertical : Metin dikey olarak yazılır.

VtOrientationUp : Metin aşağıdan yukarıya doğru yazılır.

VtOrientationDown : Metin yukarıdan aşağıya doğru yazılır.

TextLayout.HorzAligment

Metnin yatay yerleşimi bu özelliğe verilecek değerlerle belirlenir.

VtHorizontalAligmentLeft : Sola

VtHorizontalAligmentRight : Sağa

VtHorizontalAligmentCenter : Ortaya

TextLayout.VtVerticalAligment

Metnin dikey yerlesimi bu özellige verilecek degerler ile belirlenir.

VtVerticalAligmentTop : Üste

VtVerticalAligmentBottom : Alta

VtVerticalAligmentCenter : Ortaya

Chart Metodlari (Methods)

EditCopy

Bu özellik kullanılarak grafik panoya kopyalanabilir.

' MSChart1.EditCopy

EditPaste

Bu metodla panodaki bilgi grafige alınabilir.

13-MICROSOFT OFFICE'E BAGLANMAK

A. Bir İşletme Bilgi Sistemi Olusturmak

Tutarlı ve kullanımı kolay bir arabirim aracılığı ile işe ilişkin önemli bilgiler sağlayan uygulamalara İşletme Bilgi Sistemleri adı verilir. İşletme Bilgi Sistemleri genellikle Yönetim Bilgi Servisi bölümlerince ya da veri tabanı uzmanlarınca hazırlanır (bu kişiler Oracle ve SQL Server gibi veri tabanı programlarından bilgi toplama ve biçimlendirme konularında bilgi sahibi olmalıdır). İşletme Bilgi Servisi uygulamalarını kullananlar ise genellikle veri tabanı yönetimi konusunda fazla deneyim sahibi olmayanlardır.

B. İşletme Bilgi Sistemlerinin Kullanım Alanları

İyi hazırlanmış İşletme Bilgi Sistemleri, güçlü iş yönetim araçları oluşturmak için bir dizi uygulamanın çeşitli özellik ve verilerini birleştirir. Kurumunuz ve işletmeniz için özelleştirilmiş çözümler hazırlamak için Visual Basic OLE Denetimi ve elinizdeki Office belgeleri kullanılır. Bu kullanımlara örnek verilecek olursa:

B.1. Sipariş girme sistemi: Kullanıcılar, Excel kullanarak veri giriş formlarını ve gelen siparişleri görüntüleyebilir. Ayrıca Word aracılığı ile satış raporları hazırlayabilirler.

B.2. İnsan kaynakları yönetimi: Kullanıcılar bir Access veri tabanındaki kayıtları ve Word belgelerinde saklanan personel görüşmelerini yönetebilir. Aynı zamanda Paint ile fotoğraf düzenleyebilir.

B.3. Mali analiz aracı: Kullanıcılar Excel dosyalarında saklanan şirket kayıt ve defterlerini, ağ hizmetlerinden indirilen yatırım bilgilerini inceleyebilirler.

B.4. Envanter yönetim bilgi sistemi: Üst düzey yönetim ve satış personeli, Oracle ya da SQL sunucularında depolanmış envanter bilgilerinden yararlanarak üretim ve fiyatlandırma kararlarını daha iyi bir şekilde alabilirler.

B.5. Yönetim bilgi sistemi: Üst yönetim, Microsoft Outlook ile elektronik posta servislerini kullanabilir. Word belgelerine depolanan şirket bilgilerine girebilir ve Access'te özelleştirilmiş veri tabanı sorgularını çalıştırabilir.

B.6. Proje yönetim sistemi: Kullanıcılar Microsoft Project'i kullanarak takvim bilgilerini yönetebilir. Excel aracılığı ile pazarlama tahminlerini

izleyebilir, Word ile özelleştirilmiş durum raporları hazırlayabilir ve Microsoft Power Point ile de elektronik sunular hazırlayabilir.

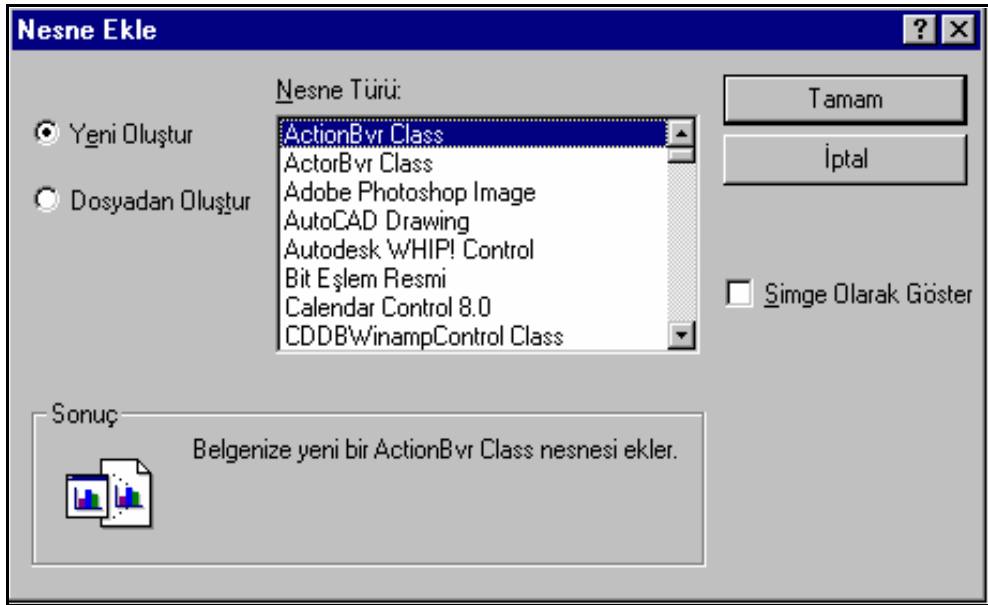
Bit Microsoft Access veri tabanındaki kayıtları, Paint'te hazırlanmış personel fotoğraflarını ve bir Excel çalışma sayfası ve grafiğindeki satış verilerini gösteren ve basit bir İşletme Bilgi Sistemi hazırlamak için Visual Basic OLE denetimi kullanılır.

C. OLE (Nesne Bağlama ve Gömme Elemanı)



Bu kontrol elemanı aracılığı ile OLE'yi destekleyen tüm programlar arasında bağlantı kurmak, bu bağlantı sonucunda istenilen veri transferi yapmak mümkündür. Dolayısıyla bir programa eklenmesi gereken bazı bölümleri eklemek yerine bu işi tam anlamıyla yapan programlara bağlantı kurmak suretiyle kolayca halledilebilir.

OLE nesnesini form üzerine aldığınızda standart OLE diyalog kutusu karşımıza otomatik olarak çıkar:



Bu pencerede OLE'yi destekleyen programlar yer almaktadır. Bunlardan herhangi biri seçildiği zaman o program ile direkt bağlantı kurularak programın kaynağına ulaşılır. Bu listedeki programlardan biri seçilerek o

uygulama ile baglanti saglanacagi gibi **Dosyadan Yarat** düğmesi kullanarak ta daha önce kaydedilmiş dosya ile baglanti kurulabilir.

OLE işlemlerinde bir nesne iki türlü kullanılabilir. Bunlar Linked (baglantili) ve Embedded (gömülü) olarak adlandırılır. Linked olarak kurulan baglantilarda uygulama ile Nesne arasında paylasilacak veri alinir. Embedded olarak kurulan iliskilerde ise Nesnenin paylasilacak verisi ile birlikte onu açip düzenleyecek kodda alinir. Linked olarak kurulan baglantilarda sadece veri alindigi için daha az yer kaplar ancak bu tip baglantilarla olusturulmus dosyalarin baska bilgisayara götürülmesi durumunda orada çalışacagi garanti edilmez. Çünkü baglantinin kurulduğu program o bilgisayarda bulunmayabilir. Embedded olarak iliski kurulduğunda ise veri ile birlikte programi düzenleyecek kodda nesneye dahil olduğu için bu tip dosyalar rahatlıkla baska bilgisayara götürölüp çalıştırılabilir. Embedded olarak kaydedilen OLE dosyaları Linked olanlara göre daha çok yer kaplar.

D. OLE Teknolojisinin Gelismisi

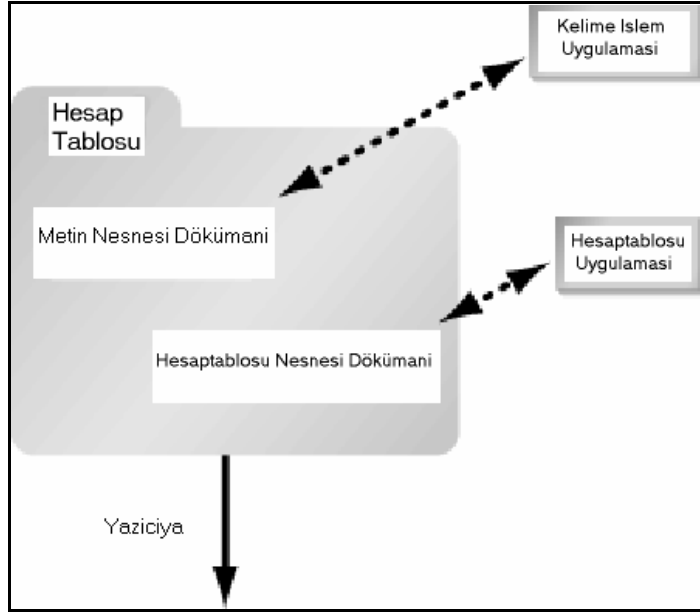
OLE DB (object linking and embedding), Microsoft'un yeni jenerasyon veri erisimi tanimidir. Bu teknolojinin amaci uygulamaların genis bir çeşitlilikteki verilere servis saglayicilar yoluyla erismesinin saglanmasidir.

D.1. DDE Teknolojisi (Dynamic Data Exchange)

DDE (Dynamic Data Exchange), uygulamaların birbiriyle haberlesmesini saglama yolunda ilk adimi olusturan teknolojiydi. Bu yöntemle uygulamalar, veri alisverisi yapabiliyor, bir uygulama diger bir uygulamadaki komutu çalıştırabiliyordu. Fakat bu sinirli, yavas ve kullanmasi zor bir teknolojiydi.

D.2. OLE 1.0

OLE 1.0 (object linking and embedding), veri bagimli bilgisayar uygulamalarında çok önemli bir kavramin geliştirilmesinde ilk adimi olusturdu. Bu kavram bir dokümanın farklı tiplerde nesneler içerebilmesi fikriydi. Nesnelerin doküman içine gömölü veya bulunduđu yere bir baglanti saglanarak dokümanla bütünlüğü saglanıyordu.



Bir Word dokümanı içine gömülü Excel dokümanı

Bu dokümanda bir metin ve bir hesap tablosu nesnesi bulunmaktadır. Bu doküman bir kelime işlemcisi kullanarak açıldığında her iki nesne de görülebilir ve eğer hesap tablosu nesnesi üzerinde işlem yapılmak istenirse üzerine çift tıklandığında hesap tablosu uygulaması açılarak hesap tablosu üzerinde işlem yapılması sağlanır. Donanım kısıtlamaları bu teknolojinin başarısız olmasının sebebiydi.

D.3. OLE 2.0

Bu teknoloji belki de anılması en zor ve karmaşık yazılım teknolojilerinden birisidir. Fakat bu karışıklık Visual Basic gibi uygulama programları tarafından gizlendiğinden kullanımı için bu teknolojiyi tamamen anlamak şart değildir.

OLE teknolojisi aslında birbiriyle pek az ilişkisi bulunan bir grup teknolojinin birleşimidir. Bunların tek ortak yanı nesneye dayalı olarak çalışmalarıdır. Bu nesneler özel olarak COM (component object model) olarak bilinir.

D.4. COM Teknolojisi (component object model)

COM (component object model)'un genel özellikleri şunlardır: Nesnelere erişme ve üzerinde işlem yapmada genel bir yöntemdir. Nesnelerin, kullanımdaysa izlenmesini, kullanımda değilse silinmesini sağlar. Standart bir hata raporlama mekanizmasıdır. Uygulamaların nesne değişimini sağlayan bir mekanizmadır.

Nesnelerin tanınmasını ve bunları işleyecek uygulamalar ile ilişkilendirilmesini sağlar. Daha önce ele aldığımız örnekte kelime işlemcisinin hesap tablosu

nesnesini kolaylıkla gösterebilmesi hesap tablosu nesnesinin bir COM nesnesi olmasındandır. COM nesnesi, nesnelerin kendilerini görüntüleyebilmeleri için bir dizi fonksiyonu destekler. Kelime işlem dokümanı hesap tablosu için bilgi içerdiği gibi doküman aynı zamanda sistem için de nesnenin bir hesap tablosu olduğuna dair bilgi tasir. Yani dokümanda nesneyi görüntüleyen asil fonksiyonlar bulunmaz. Bu fonksiyonlar bir uygulama içinde ya da .dll içinde tutulur. Burada kelime işlem uygulaması çizim fonksiyonlarını içeren hesap tablosu uygulamasını bulmak için COM mekanizmasını kullanır. Hata olustugunda da yine bir COM standardi devreye girerek her iki uygulamanın da silinmesini sağlayacaktır.

Windows'un su an içerdiği DCOM (distributed common object model) COM'un genişletilmiş bir seklidir. DCOM nesneleri OLE vasıtasıyla ağ üzerinde değişik sistemlerde çalışan uygulamalar arasında da transfer edilebilir.

D.5. UUID Yapısı (universally unique identifier)

UUID (universally unique identifier) veya GUID (globally unique identifier) veya CLSID (class identifier)

OLE'nin temel ihtiyaçlarından birisi de nesneleri tanımlama gereksinimidir. Bir uygulama birçok farklı çeşit nesneyi içeren bir doküman ile çalışıyorsa bu nesneleri doğru olarak tanımlayabilmelidir ki sistem bu nesneyle işlem yapabilecek uygulamayı tanımlayabilsin. Bunu sağlayabilmek için COM her çeşit nesneye 16 baytelik bir değer atar.

örnek 2 - sistem registry'sindeki bir GUID numarası {970EDBA1-111C-11d0-92B0-00AA0036005A}

Bu GUID numaraları, evrensel ve sadece atandığı nesneye özeldir. Numaraları olusturmak için network kartının adresinden yararlanılır (eger bir network kartı yoksa bu GUID olusturucu programca olusturulur). Visual Basic 5.0, GUID numaralarını otomatik olarak sizin için olusturmaktadır ve olusturulan nesneler de bu numaralarca tanımlıdır.

D.6. Yapılandırılmış Doküman Saklama Yöntemi (OLE Structured Storage)

Farklı yapıdaki nesneleri içeren OLE dokümanlarının saklanması için kullanılan yöntem OLE Structured Storage yöntemidir. Bu yöntemde bir doküman bir dizi hiyerarsik yapıdaki storage ve dizilere ayrılır. Burada storage kabaca klasör yapısına, dizi de dosyaya karşılık gelmektedir.


D.7. OLE Özdevinim Yapısı (OLE Automation)

OLE automation, bir uygulamanın diğer bir uygulamadaki komutları çalıştırmasını sağlayan yapıdır. OLE automation sadece fonksiyonların çağırılmasını sağlamakla kalmaz, aynı zamanda çalışma anında fonksiyonların hangi nesneleri kullanabileceğini ve hangi parametreleri alabileceğini belirler. OLE automation anlayışı aynı zamanda ActiveX teknolojisinin de temelini olusturan bir basamak olusturmıştır.

E. OLE Denetimini Kullanmak

Visual Basic programlarına OLE desteğine sahip nesneler eklemek için OLE denetimi kullanılır. Erisilecek OLE nesneleri, sisteminize yüklenmiş olan Windows uygulamalarına bağlıdır. OLE desteğine bağlı her uygulama, desteklediği nesnelerle birlikte, Windows kayıt defterinde (Windows'un bu türden bilgileri saklamak için kullandığı sistem çapında bir veri tabanı-registry) tanımlanmıştır. OLE nesnelerini, OLE denetimini ilk kullandığımızda görünen Insert Object (nesne ekle) iletişim kutusunu kullanarak seçebiliriz. OLE nesneleri arasında Excel çalışma sayfaları, Excel grafikleri, Word belgeleri ve Microsoft ClipArt görüntülerini sağlayabiliriz. OLE nesneleri yeni, boş uygulama belgeleri ya da diskten yüklenen önceden hazırlanmış dosyalar olabilir.

HAFTALIK İZLEYEN KİŞİ SAYISI



ORJİNAL ADI:
THE LORD OF THE RINGS THE TWO TOWERS

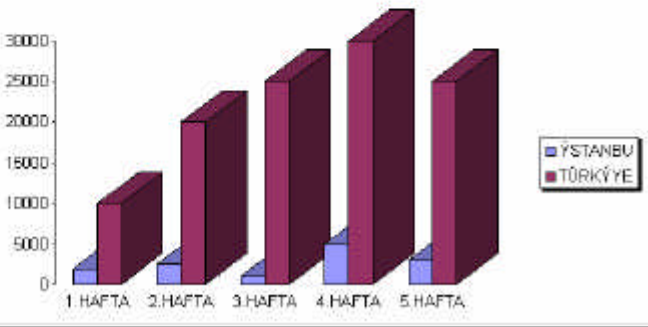
TÜRKÇE ADI:
YÜZÜKLERİN EFENDİSİ İKİ KULE

BÖLÜM SAYISI:
İKİNCİ BÖLÜM

ÇIK: **FİLMLER**

YÜZÜKLERİN EFENDİSİ İKİ KULE HAFTALIK TOPLAM KİŞİ SAYISI		
	İSTANBUL	TÜRKİYE
1. HAFTA	2000	10000
2. HAFTA	2500	20000
3. HAFTA	1000	25000
4. HAFTA	5000	30000
5. HAFTA	3000	25000

TOPLAM KİŞİ SAYISI



Legend: ■ İSTANBUL ■ TÜRKİYE

Microsoft Visual Basic 6.0

Sekildeki gibi bir işletme bilgi sistemi olusturmak için bir veri nesnesi, bir görüntü nesnesi iki OLE nesnesi ve çeşitli etiket ve metin kutuları kullanılmıştır. Bu örnek uygulama, bir fotografi, Access veri tabanından alınan kayıtları ve Excel çalışma sayfası ve grafiginden alınan verileri görüntüleyecektir. Çalışma bittiginde yukarıdaki şekil elde edilir.

Bu uygulamayı yapabilmek için:

1. Visual Basic başlatılır ve yeni bir standart proje açılır ve ya Visual Basic açık ise File menüsündeki New Project komutu tıklanır.
2. Label denetimini kullanarak formun üst ortasında geniş bir etiket oluşturulur. Bu nesneye programı adı yazılır.



3. Imge denetimini kullanarak etiketin altında, formun sol tarafında bir görüntü nesnesi oluşturulur. Bu nesne elektronik tarayıcıda taranmış ve bmp dosyası olarak kaydedilmiş olan fotografi görüntüleyecektir.



4. TextBox denetimi kullanılarak, görüntü nesnesinin sağ tarafında beş tane metin kutusu nesnesi oluşturulur.



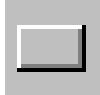
Bu metin kutuları, yapılan uygulamanın veri tabanında bulunan filmin orjinal adı, türkçe adı ve bölüm sayısını içeren bir Access veri tabanı ile bağlanacaktır. Aşağıdaki şekilde veri tabanı gösterilmektedir.

YÜZÜKLERİN EFENDİSİ : Tablo			
Kimlik	Alan1	Alan2	Alan3
1	THE LORD OF THE RINGS THE TWO TOWERS	YÜZÜKLERİN EFENDİSİ İKİ KULE	İKİNCİ BÖLÜM
(OtomatikSayı)			

5. Data denetimi kullanılarak, metin kutusu nesnelerinin sağına bir veri nesnesi eklenir. Kayıtlarda gezinmek için kullanılmaktadır.



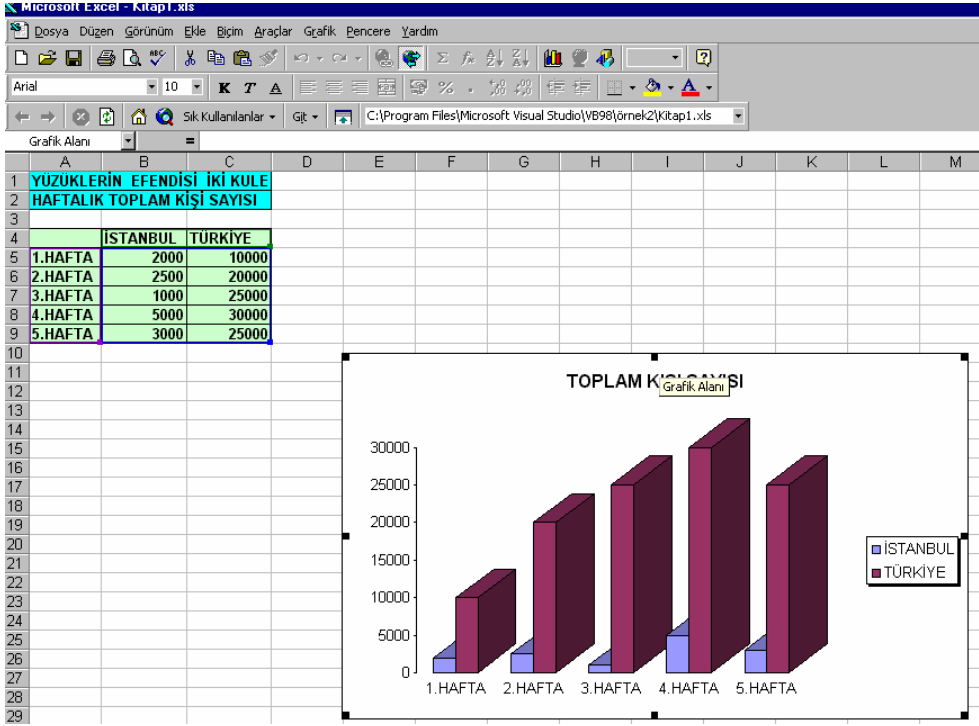
6. Veri nesnesi altına bir komut düğmesi eklemek için CommadButton denetimi kullanılır. Bu komut düğmesi programdan çıkmak için



kullanılmaktadır.

7. Formun nesneleri için su ayarlar yapılır:

<u>Nesne</u>	<u>Özellik</u>	<u>Ayarlar</u>
Form1	Caption	"yüzüklerin efendisi iki kule"
Label1	Caption	"Haftalık İzleyen Kisi Sayısı"
	Font	Verdana, Kalın, 16 Punto
	ForeColor	mavi
Imge1	BorderStyle	1-Fixed Single
	Stretch	True
Picture	C:\Program File\Microsoft Visual Studio\Vb98\örnek	
Data1	Caption	"Filimler"
	Connect	Access
DatabaseNeme	C:\Program File\Microsoft Visual Studio\Vb98\örnek\vtb1	
	ReadOnly	True
	RecordSource	Artists
Text1	DataSource	Data1
	DataField	Alan1
	Text	(bos)
Text2	DataSource	Data1
	DataField	Alan2
	Text	(bos)
Text3	DataSource	Data1
	DataField	Alan3
	Text	(bos)



Command1 Caption

"çık"

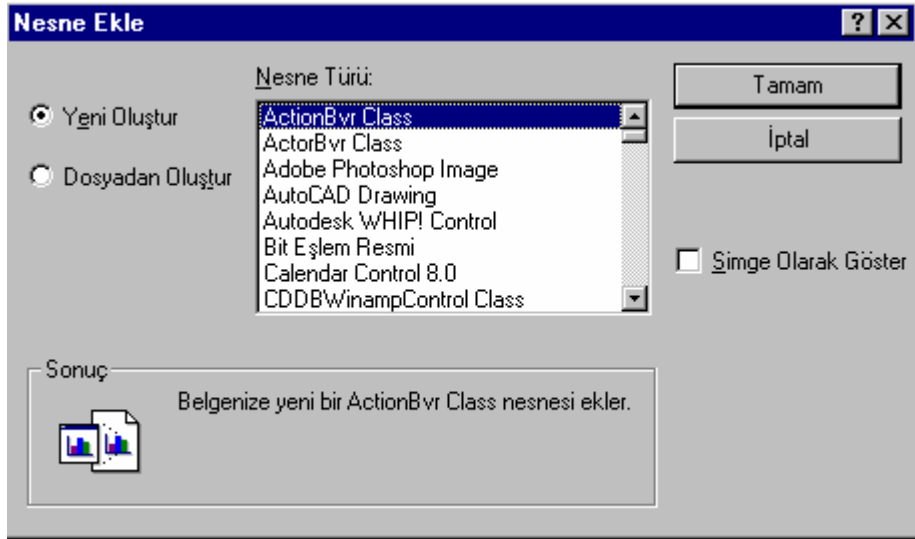
8. Çık komut düğmesini çift tıklayın ve Command1_Click olay yordamına End yazılır. Bu program bildirisi, Çık düğmesi tıklandığı zaman uygulamayı kapatacaktır.

```
ect1 - Microsoft Visual Basic [design] - [Form1 (Code)]
Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
Ln 9, Col 1
Form Load
Private Sub Command1_Click()
End
End Sub
```

9. Yapılan uygulama kaydedilir. Artık OLE denetimi kullanılarak uygulamamıza Excel çalışma sayfası ve grafiği eklenebilir.

10. Uygulama için hazırlanan Excel çalışma sayfası:

11. OLE denetimi tıklanır ve uygulamamızın formunun üzerinde geniş bir dörtgen çizilir. Dörtgeni çizip farenin düğmesini bıraktığınızda InsertObject iletişim kutusu ekrana gelir.



InsertObject (Nesne Ekle) iletişim kutusunda, uygulamaya bağlanabilecek ya da karşılaştırılabilir nesnelerin bir listesi bulunur.

12. Insert Object iletişim kutusunda Create From File (dosyadan oluşturun) seçeneği tıklanır. İletişim kutusunda bir nesne yolu görünür. Programınıza var olan bir dosyayı eklemek istediğinizde, Create From File (dosyadan oluşturun) seçeneği seçilir.



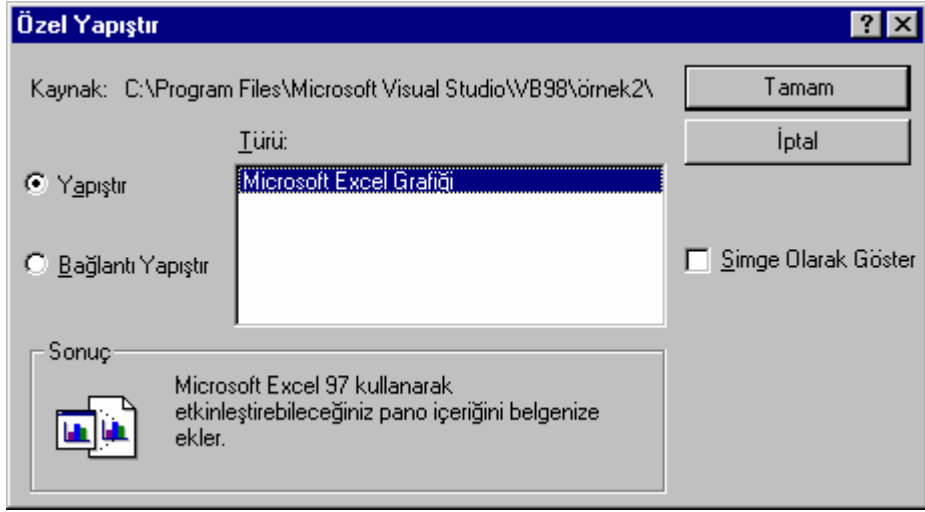


13. Bağlantılı bir nesne oluşturmak için Link (bagla) onay kutusu seçilir. Link onay kutusu seçildiğinde, seçilmiş olan dosyanın resmi OLE nesnesine yüklenecektir. Dosya bağlantılı bir nesne olacağından, Visual Basic uygulamasının dışında ver olmaya devam edecek ve bu dosya üzerinden yapılan değişiklikler, oluşturulan programa yansiyacaktır.

14. Programa bağlamak istenilen Excel dosyasını bulmak için Browse (gözet) seçeneği tıklanır. Uygulamamıza koyulacak Excel dosyasının yeri belirlenir ve gerekli işlemler yapılarak onay verildiğinde, Excel çalışma sayfası dosyasının bir resmi OLE nesnesinde görünür.

15. OLE denetimi yeniden tıklanır ve ekranın diğer köşesine bir dörtgen çizilir. Farenin düğmesi bırakıldığında Nesne Ekle (Insert Object) iletişim kutusu ekrana gelir.

16. Nesne Ekle iletişim kutusu kapatılır. Çünkü OLE nesnesine çalışma sayfasının sadece bir kısmı (Excel grafiği) yerleştirilecektir. Bir dosyanın belirli bir kısmını yerleştirebilmek için, yerleştirilmek istenilen kısım kopyalanır ve Past Special (Özel Yapıştır) komutunu kullanarak OLE nesnesinin içine yapıştırılır.

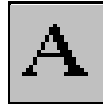


17. Excel çalışma sayfasını gelinir ve grafik seçilir ve kopyalanır. Daha sonra tekrar Visual Basic ortamına geri gelinir ve ekrana açtığımız dörtgenin içine farenin tusu tıklanır ve çıkan pencereden Paste Special seçeneği seçilir. Ardından Bağlantı Yapıştır seçeneği seçilir ve tamam tıklandıktan sonra grafigimiz artık istenilen yere eklenir.
18. Uygulanma baslatıldığında Excel dosyalarına olan bağlantıların güncellemesini sağlamak için program kodu girilmesi gerekmektedir. From_Load olay yordamının Code penceresine :

OLE1.Update

OLE2.Update

Program bildirileri yazılır. Bu bildiriler, Excel dosyalarında yapılan değişiklikleri Visual Basic uygulamalarına yüklemek için kullanılır.



Uygulama kaydedilerek çalıştırılabilir ve yapılan bağlantılar incelenebilir. Form üzerindeki "Yüzüklerin Efendisi Haftalık Toplam Kisi Sayısı" olan bölüm tıklanır. Karsımıza hazırlanan Excel çalışma sayfası açılır. Eklenmiş olan nesne, çalışma sayfasına bağlı olduğundan, Visual Basic özgün çalışma sayfası üzerinden değişiklikler yapmamıza ve bu değişiklikleri gene özgün dosyaya kaydetmeye olanak sağlar.

F. Automation'i Kullanarak Uygulama Nesnelerini Programlamak

Automation'i tam olarak destekleyen Windows tabanlı uygulamalar, uygulama islevlerini, bağlantı özellik ve yöntemlerle birlikte erişime açar. Nesnelerini erişime açan Windows tabanlı uygulamalara nense uygulamaları ya da sunucu uygulamaları adı verilir. Buna karşılık erişime açılmış bu nesneleri kullanan programlamalar ise denetçi ya da istemci uygulamalar denir. İstemci ya da sunucu olarak kullanılacak Microsoft uygulamaları:

- ❑ Microsoft Visual Basic
- ❑ Microsoft Word 97
- ❑ Microsoft Excel 97, Microsoft Excel 95, Microsoft Excel 5.0
- ❑ Microsoft PowerPoint 97
- ❑ Microsoft Project 97, Microsoft Project 95
- ❑ Microsoft Outlook 97, Microsoft Outlook 98 (VBScript diliyle geliştirilen özel formlar)

F.1. Visual Basic'te Automation'i Kullanmak

Microsoft Visual Basic 6 ile Automation'i destekleyen nesne ve denetçi uygulamaları yaratılabilir. İşlevliklerini erişime açan uygulamalar oluşturmak, hem Visual Basic'in Professional ya da Enterprise sürümlerine sahip olması gerekiyor. Nesne uygulamaların özelliklerini kullanan denetçi uygulamalar oluşturmak ise, Visual Basic'in tüm sürümlerde oldukça kolaydır.

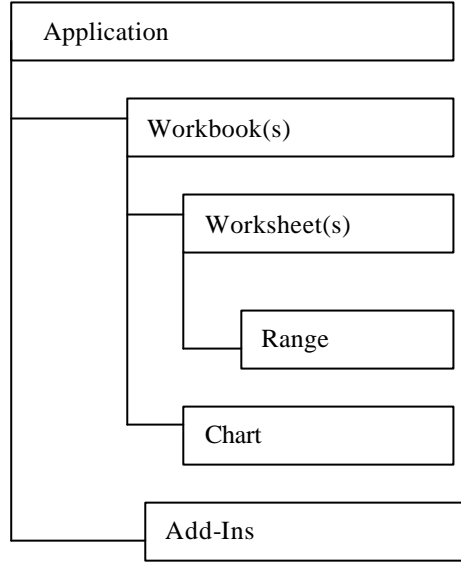
F.2. COM Teknolojisi

Client uygulamalar yaratmak için server nesnelerinin ve COM (Component Object Model) teknolojisinin bilinmesi gerekir. COM teknolojisini iki görevi vardır. Birinci bir nesnenin tanımlanması. İkincisi ise nesnenin yaratılması ve client ile server arasında iletişim kurulmasıdır. COM teknolojisi bileşenlerin nasıl yaratılacağını ve client uygulamaların bileşenlere nasıl bağlanacağını tanımlar.

COM terminolojisi: Components (bileşenler), Object (nesne), Automation (otomasyon). Bir bileşen belli fonksiyonu işletilebilir kod parçasıdır. EXE, DLL, OCX olabilir. Server uygulamalar bir ya da daha çok bileşenlerden oluşur. COM teknolojisi bileşenlerin nasıl yaratılacağını ve client uygulamaların buna nasıl bağlanacağını belirler. Nesneler ise bir birim olarak kullanılan kod birimleridir. Belli bir süre için yaratılırlar ve dağıtılırlar. Automation terimi ise bileşen yaratma ve nesne kullanma yöntemlerini tanımlar.

F.3. Nesne Teknolojisi

Nesne modeli bir server uygulamadaki nesneler arasındaki ilişkileri tanımlar. Nesne modeli sayesinde nesnelerin yapısını ve içerdiği nesneleri görmek mümkündür.



Microsoft Excel Nesneleri

F.4. Visual Basic ile Bir Client Yaratmak

Bir client uygulama yaratmak için nesneler arasındaki ilişkinin ve nesnelerin kullanımının bilinmesi gerekir. Visual Basic ile bir client uygulaması yaratmak için gerekli adımlar :

1. Bilesen kütüphanesine bağlantı kurulur.(referans)
2. Nesne değişkeni tanımlanır.
3. Nesne yaratılır.
4. Nesnenin metotları ve özellikleri kullanılır.

F.4.1. REFERANSLARIN DÜZENLENMESİ

Bir client uygulaması yaratıldığında; client bir bileşene bağlanır. Bir client'i bir bileşene bağlamak için bileşenin kütüphanesine referans gerekir. Bu işlem için: Project menüsünden References komutu tıklanır. References iletişim kutusunda bağlanılacak kütüphane seçilir.

F.4.2. NESNELERİN TANIMLANMASI

Bir nesne degiskeni kullanima göre generik ya da spesifik olarak tanımlanır. Generik nesne tanımlamaları her tip nesneye pointer görevi görür.

Dim nesne1 As Object

Spesifik nesne tanımlaması ise sadece o nesneye ilişkin pointer görevi görür.

Dim nesne1 As Excel.Application

F.4.3. NESNELERİN YARATILMASI

Bir client uygulaması yaratmak için üçüncü adım ise nesnenin yaratılmasıdır. Visual Basic'te nesne yaratmak için dört ayrı yol varır:

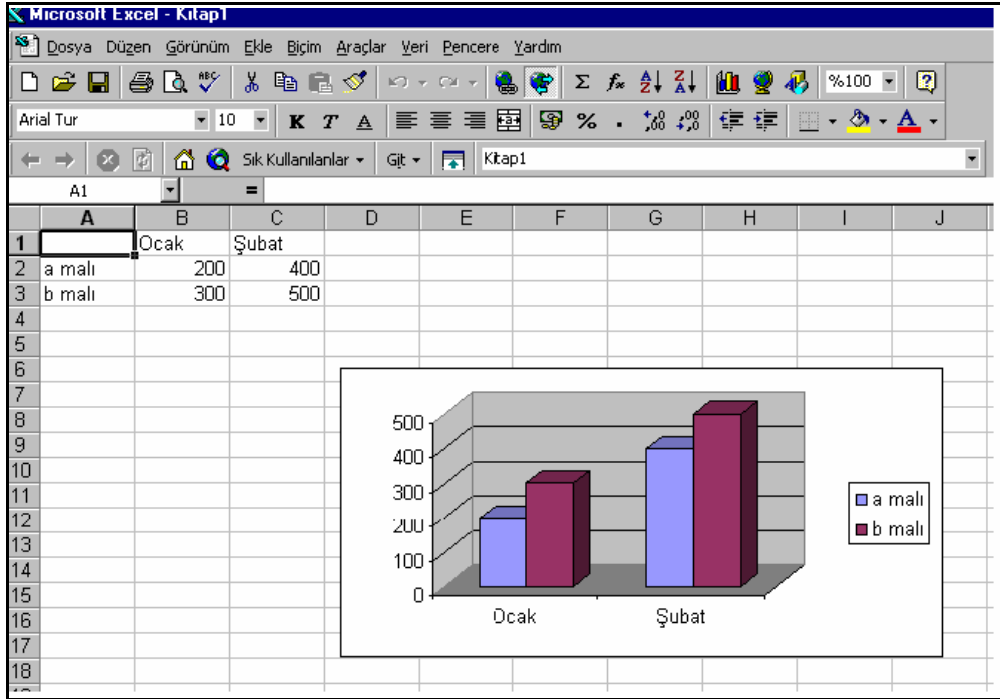
- ☐ CreateObject fonksiyonunu kullanmak.
- ☐ Set ve New deyimlerini kullanmak
- ☐ Modülün tanımlama kesimlerinde As New deyimini kullanmak
- ☐ GetObject fonksiyonunu kullanmak

CreateObject fonksiyonunun kullanımı:

```
Dim xlApp as Excel.Application  
Set xlApp = CreateObject("Excel.Application")
```

GetObject fonksiyonunu kullanma :

```
Dim xl As Object  
Set xl = GetObject("c:\my documants\butce.xls")
```



Örnek: Microsoft Excel ile Bağlantı

```

Dim xlApp As Excel.Application
Dim xlBook As Excel.Workbook
Dim xlSheet As Excel.Worksheet
Dim xlc As Excel.Chart
Set xlApp = CreateObject ("Excel.Application")
xlApp.Visible = True
Set xlBook = xlApp.Workbooks.Add
Set xlSheet = xlBook.Worksheets(1)
xlSheet.Range ("b1").Value = "Ocak"
xlSheet.Range ("c1").Value = "Şubat"
xlSheet.Range ("A2").Value = "a mali"
xlSheet.Range ("A3").Value = "b mali"
xlSheet.Range ("b2").Value = "200"
xlSheet.Range ("b3").Value = "300"
xlSheet.Range ("c2").Value = "400"
xlSheet.Range ("c3").Value = "500"
xlApp.ActiveWorkBook.SaveAs FileName := "c:\Kitap1.XLS"
Msgbox "çıkamak istiyor musunuz", vbYesNoCancel
If vbYesNoCancel Then xlApp.Quit
xlSheet.Range ("A1:C3").Select
Set xlc = xlApp.Charts.Add
ActiveChart.ChartType = xl3DcolumnClustered

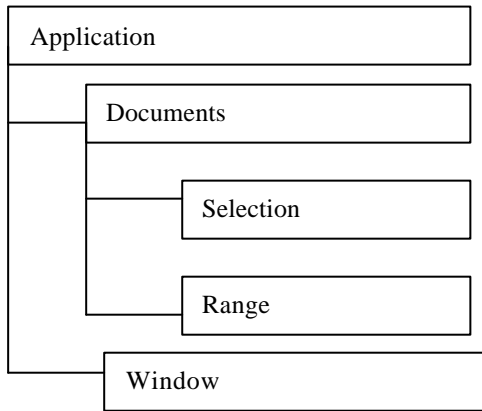
```

Microsoft Visual Basic 6.0

```
ActiveChart.SetSourceData      Source:      =      Sheets  
("Sheet1").Range("A1:c3"), PlotBy:=xlRows  
ActiveChart.Location Where:=xlLocationAsObject, Name:"Sheet1"  
Whit ActiveChart  
.HasTitle =False  
.Axes(xlCategory). HasTitle =False  
.Axes (xlseries) . HasTitle =False  
.Axes(xlValue). HasTitle =False  
End Whit
```

F.5. Application Nesnesi ile Çalışmak

Bir Microsoft Excel ya da Word ortamı baslatıldığında otomatik olarak bir Application nesnesi yaratılır.



Microsoft Word Nesne Modeli

Programci Application nesnesinin özelliklerini ve metotlarını kullanarak uygulamanın tamamı üzerinde işlemler yapılabilir. Örneğin baskı ön izleme için:

```
Application.PrintPreview=True
```

Application nesnesinin bazı özellikleri ise uygulamanın görünümünü düzenler. Örneğin DisplayStatusBar özelliğine True olması durum çubuğunun görünür olmasını sağlar. Yine benzer biçimde WindowState özelliğinin wdWindowStateMaximize olması uygulamanın penceresinin ekranı kaplamasını (Maximized) sağlar.

```
With Application
    .WindowState = wdWindowStateNormal
    .Height = 450
    .Width = 600
End With
```

Application nesnesinin altında Documents ve Windows koleksiyonları yer alır. Documents koleksiyonu Word içinde açılmış bütün belgeleri temsil eder. Örneğin bir belgeyi açmak için:

```
Application.Documents.Open FileName:="c:\my documents\mektup.DOC"
```

Ya da;

```
Documents.Open FileName:="c:\my documents\mektup.DOC"
```

F.6. Document Nesnesi ile Çalışmak

Microsoft Word içinde bir belgeyi yaratmak ya da açmak için Document nesnesi ile kullanılır. İstenilen bir belgeye erişmek için bu belge adıyla ya da indeks numarasıyla gösterilir.

İstenilen belgenin gösterilmesi: Documents (indeks) ya da; Documents ("Rapor.doc")

Örnekte Rapor.doc adlı bir belge tanımlanmaktadır:

```
Set WordDoc = Documents ("Rapor.doc")
```

Belgenin açılması için; Open metodu kullanılır. Open metodu ile Documents koleksiyonundan belirtilen bir nesne açılır:

```
Set WordDoc = Documents.Open (FileName="c:\my documents\mektup.doc")
```

Belgelerin yaratılması ve kayıt edilmesi için; Add metodu kullanılır.

```
Documents.Add
```

Add Metodu yeni bir document nesnesi yaratır. Böylece Word ortamına bir belge eklenmiş olur. Ardında istenilen özellikler kullanılarak belge islenir. Örnekte bir belge yaratılmakta ve üst sinirine bir değer atanmaktadır:

```
Dim myDoc As Document
Set myDoc = Documents.add
myDoc.PageSetup.TopMargin = InchesToPoints(1.25)
```

Microsoft Visual Basic 6.0

Belge yaratildiktan sonra (ilk defa kayit etmek için)SaveAs metodu kullanılır.

```
ActiveDocument.SaveAs FileName:="rapor.doc"
```

Çok sayıda belge arasından birini aktif hale getirmek için Activate metodu kullanılır.

```
Documents("özelmehtup.doc").Ativate
```

Bir belgeyi yazdırmak için Printout metodu kullanılır.

```
ActiveDocument.PrintOut
```

Aktif belge birinci sayfadan üçüncü sayfaya kadar yazıcıdan bastırmak için:

```
ActiveDocument.PrintOut Range:=wdPrintFromTo, From:="1", to:="3"
```

Bir belgeyi kapatmak için Close metodu kullanılır.

```
Documents("rapor.doc").Close
```

Eğer belgede değişiklik varsa Close metodu değişiklikleri kayit etmek için soru sorar. Bu işlemi engellemek için wdDoNotSaveChanges ya da wdSaveChanges sabitleri kullanılır. Örnekte belge kayit edilip kapatılmakta:

```
Documents("Sales.doc").Close SaveChanges:=wdSaveChanges
```

Örnekte belge kayit edilmeden kapatılmaktadır:

```
Documents.Close SaveChanges:=wdDoNotSaveChanges
```

Belgeye nesne eklemek için .Add metodu kullanılır. Böylece belgeye tablolar, açıklamalar, vb öğeler eklenebilir.

Örnekte 4x4boyutlarında bir tablo eklenmektedir:

```
ActiveDocument.Tables.Add Range:=yer, NumRows:=4, NumColumns:=4
```

'burada yer degiskeni tablonun eklenecegi yeri belirtir.

Range nesnesi belge içindeki bir yeri gösterir. Bu ter daha sonra bir nesne eklemek için ya da biçimleme yapmak için kullanılır. Range nesnesi aynı zamanda belgenin bir kısmını da temsil eder.

Range nesnesi başlangıç ve bitiş karakterlerinin konumu ile tanımlanır. Range nesnesi herhangi bir seçim nesnesi ile ilgisi yoktur. Bu nedenle range nesnesi kullanılarak herhangi bir seçim yapmadan belge üzerinde belli kısımla biçimlenebilir.

Range nesnesini tanımlamak için Start, End ve StoryType özellikleri tanımlanır. Satır ve End özellikleri alanın basındaki ve sonundaki karakterlerin konumunu gösterir.

```
Set myRange = ActiveDocument.Range (Start:=0; End:=10)
```

Aktif belgenin ilk on karakterini biçimlemek için:

```
Set myRange = ActiveDocument.Range (Start:=0; End:=10)  
MyRange.Bold=True
```

Aktif belgenin basına metin girmek için:

```
Set myRange = ActiveDocument.Range (Start:=0; End:=0).InsertBefore_  
Text:="metin bilgi"
```

Aktif belgenin ikinci ve üçüncü cümlesini seçmek için:

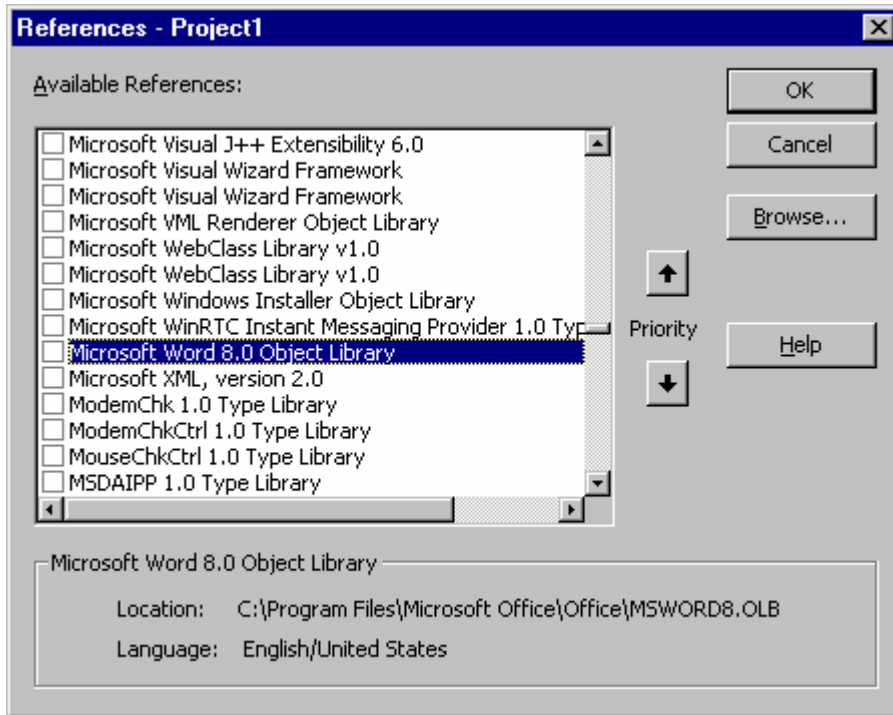
```
Set myDoc = ActiveDocument  
Set myRange = myDoc.Range (Start:=myDoc.Sentences(2).Start,_  
End:=myDoc.Sentences(3).End)
```

G. Visual Basic Object Browser

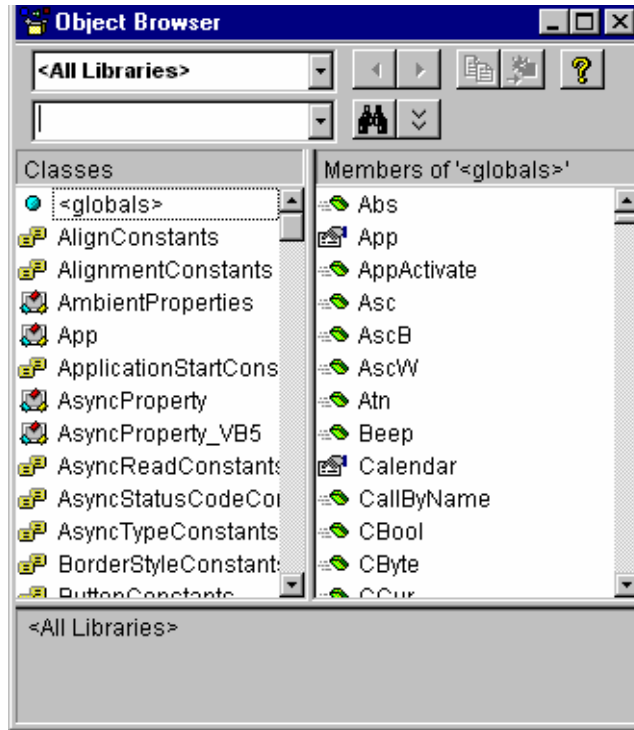
Visual Basic Object Browser iki farklı kullanımı olan bir görüntüleme yardımcı programıdır: Gözet'i Visual Basic programlama ortamında üzerinde çalışılan programda kullanılan nesne, özellik ve yöntemlere göz atılabilir.

G.1. Word Nesnelerini Görmek için Object Browser Kullanımı

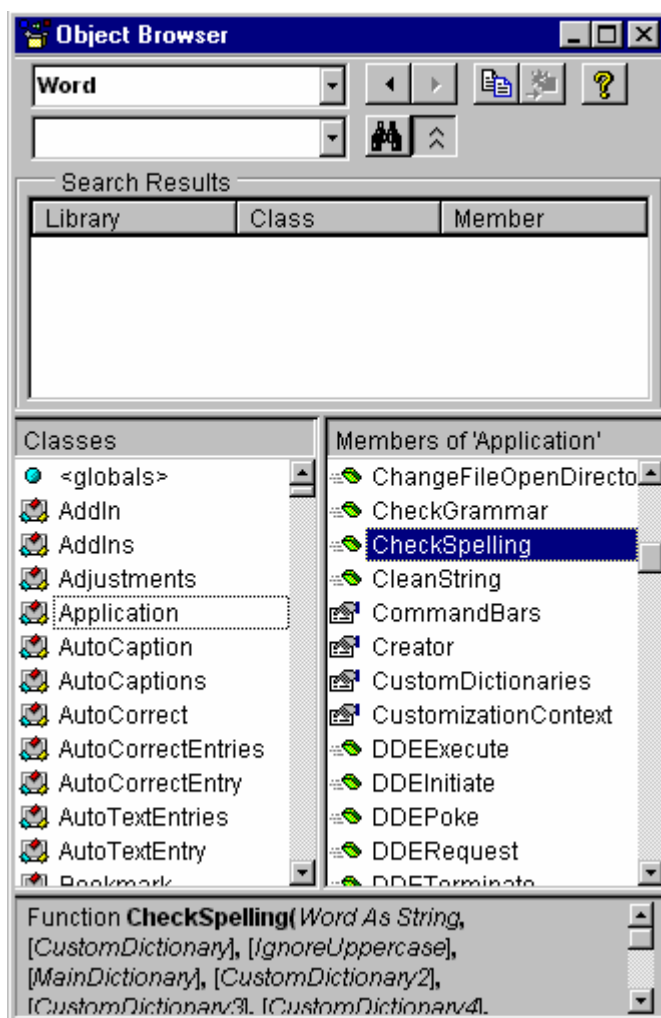
1. Yeni bir proje açılır.
2. Project menüsündeki References komutu tiklanır. Sisteminizde bulunan herhangi bir nesne kütüphanesine yönelik başvuruları projeye katmaya olanak veren References iletişim kutusu ekrana gelir. Projeye başvurular eklemek derlenmiş programı daha fazla genişletmez ancak, programa ne kadar fazla başvuru eklenirse, Visual Basic'in programı derlemesini o kadar uzamış olur. Bu nedenle, Visual Basic isteğe bağlı olarak başvuruları Automation nesnesine ekler.
3. Microsoft Word 8.0 Object Library başlıklı başvurunun yanındaki onay kutusu seçilir.



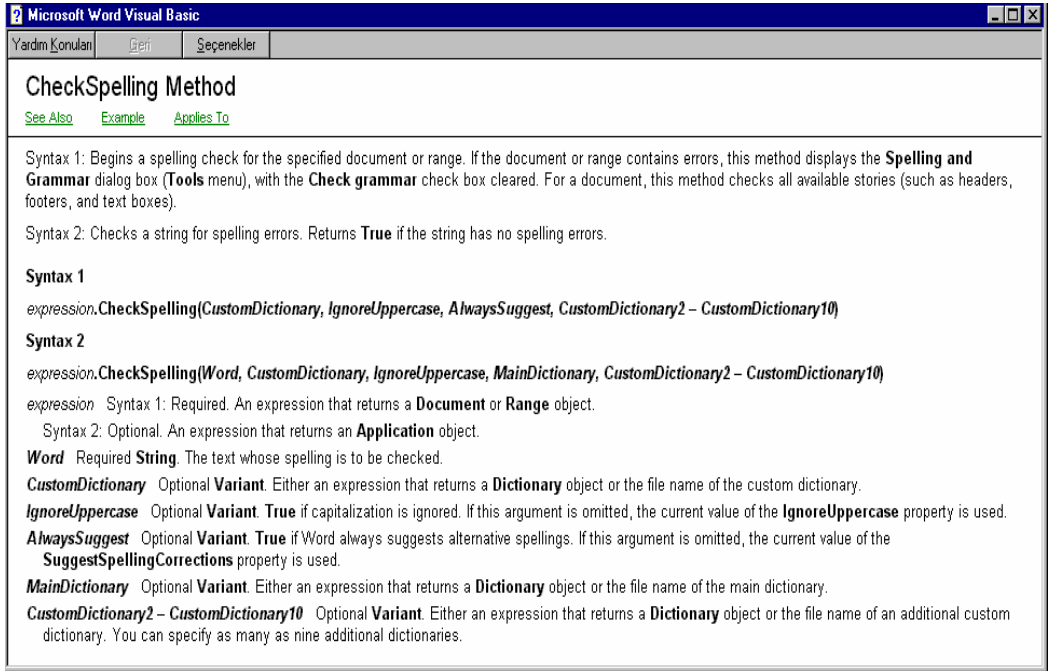
4. İletişim kutusunu kapatmak ve başvuruyu projeye eklemek için Ok tiklanır.
5. View menüsündeki Object Browser komutu tiklanır.



6. Project/Library liste kutusu açıldıktan sonra Word nesne kütüphanesi tiklanır.
7. Classes liste kutusundan Application nenesi tiklanır. Application ile ilgili yöntem ve özelliklerin listesi, Members liste kutusunda açılır. Bunlar çalışma sayfasındaki bilgileri işlemek için Word'ün sağladığı komutlardır.
8. Memebers kutusundan CheckSpelling yöntemi tiklanır. Objcet Browser'in altında CheckSpelling yöntemine ait söz dizimi görünür. Bu yöntem Word'ün Visual Basic'ten erisebilecek bir islevi olan, azim denetimini harekete geçirir.



9. Object Browser'in üst kısmındaki soru işareti kısmi tıklanır. Office 97 kurulum programı çalıştırıldığında Visual Basic for Application yardım dosyası yüklendiyse , CheckSpelling yöntemi için bir yardım yöntemi programlama ortamında açılır.



Bu yardım dosyası, Word nesne kütüphanesinde bulunana özellik ve yöntemlerin nasıl kullanılacağı konusunda ayrıntılı bilgi verir.

10. CheckSpelling yöntemi hakkında gerekli bilgiler okunduktan sonra Object Browser kapatılır.

Visual Basic'ten Word'ü Kullanmak

Word CheckSpelling yöntemi bir visual basic programında kullanmak için aşağıdaki programlama adımlarını uygulamamız gerekir. Söz konusu olan teknikler, birçok uygulama nesnesine uygulayabileceğimizden, burada öğreneceğimiz bilgileri, Automation desteğine sahip birçok uygulamadaki işlevleri kendi programlarınıza eklemek için kullanabilirsiniz.

Adım 1 References komutunu kullanarak gerekli nesne kütüphanelerine yönelik başvuruları projenize ekleyin.

Adım 2 Kendi Visual Basic programınızı yazın. İçinde Automation kullanmayı düşündüğünüz olay yordamında Dim bilgisini kullanarak bir nesne değişkeni yaratın ve sonra CreateObject işlevi aracılığıyla bir Automation nesnesini nesne değişkenine ekleyin.

Dim X As Cbject

'x' i değişken adı olarak kullanın

Microsoft Visual Basic 6.0

Set X = CreateObject ("Word.Aplication")

Adim 3 : Olay yordamındaki Automation nesnesinin yöntem ve özelliklerini kullanın. Gerektilğinde, söz dizimi için Object Browser'daki yardım dosyalarına yada nesne uygulamasının kullanım kılavuzlarına bakabilirsiniz:

X.Visible = False	'Word 'ü gizle
X.Documents.Add	'Yeni belge aç
X.Selection.Text	'Metin kutusunun içeriğini bu belgeye
Kopyala.	
X.ActiveDocument.CreckSpelling	'yazılım denetimini çalıştır
Text1.Text = X.Selection.Text	'Düzeltilmiş metni Visual Basic' e kopyala

Adim 4 Nesne uyulamasını kullanmayı bitirdiğinizde uygulamayı kapatın ve bellekte yer açmak için nesne değişkenini boşaltın:

X.Quit	'Word' ü kapat
Set X =Nothing	'Nesne değişkenini boşalt

Bir sonraki alıstırmada, bir Visual Basic metin kutusundaki yazım hatalarını denetlemek için Word yazım denetçisini kullanan bir uygulama yaratacaksınız. Hazırlayacağınız program tamamen Visual Basic içinde olacak ve Automation'ı aracılığıyla uzaktan Word işlevini kullanacak.

NOT : Bir sonraki alıstırmada, sisteminizde Word 97 ya da Office 97'nin (Word 97'yi içerir) yüklü olduğu varsayılmıştır.

Özel bir yazılım denetçisi hazırlayın.

1. Project menüsündeki References komutunu tıklayın. Microsoft Word 8.0 Object Library başvurusunun yanında bir onay işareti bulunduğundan emin olduktan sonra OK'i tıklayın.

Microsoft Word 8.0 Object Library, Word nesne uygulaması tarafından erişime açılan nesne, yöntem ve özelliklere erişmemizi sağlar. Bu, dersin başında eklemiş olduğunuz nesne kütüphanesi başvurusudur.

NOT : Nesne kütüphanesi başvuruları, References komutu aracılığıyla her yeni projeye eklenmelidir.

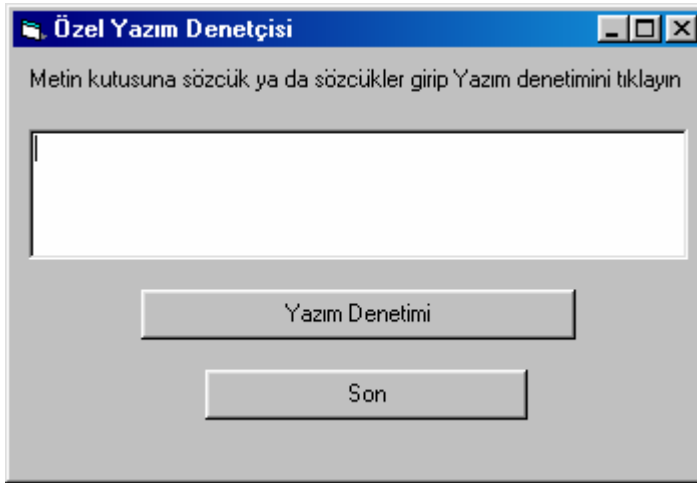
Siradaki işlemimiz ise yazım denetçisi için bir form yaratıp, program kodunu yazmaktır.

2. Daha küçük ve dikdörtgen görünümüne gelene kadar formu küçültün.

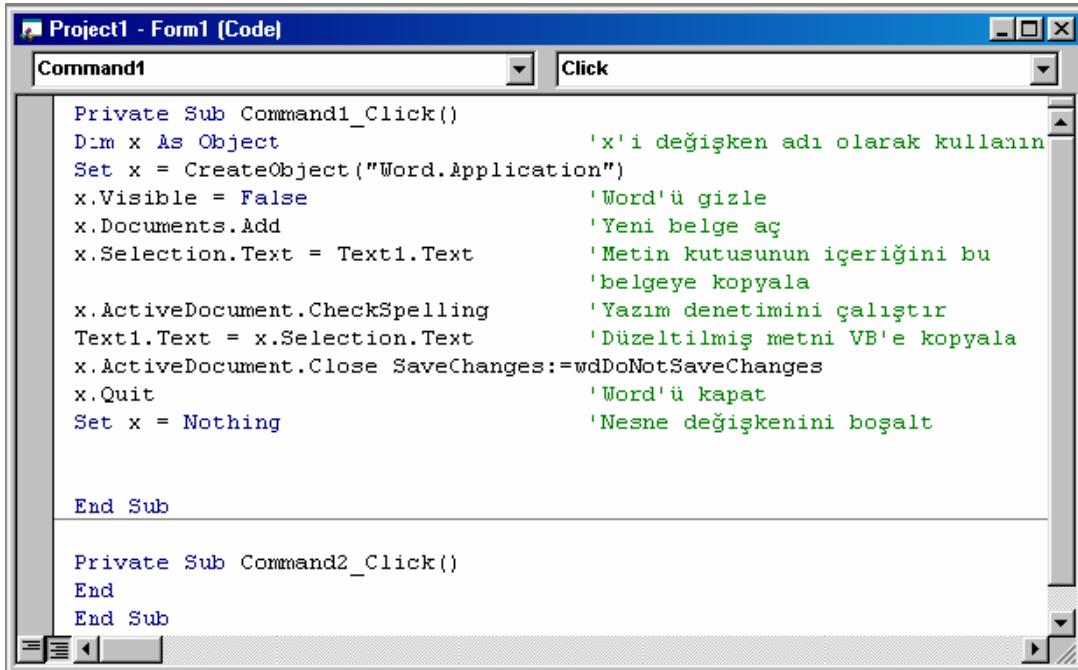
3. Label denetimin kullanarak formun üstünde uzun bir etiket oluşturun. Oluşturduğunuz bu etiket, programın kullanımı için gerekli yönergeleri görüntüleyecek.
4. Formun ortasında geniş, dört satır yüksekliğinde bir metin kutusu oluşturmak için TextBox denetimini kullanın. Bu metin kutusu kullanıcının yazım hatalarını denetleyeceği metni saklayacak.
5. Metin kutusu nesnesinin altında iki komut düğmesi oluşturmak için CommandButon denetimini kullanın, Birinci komut düğmesi Word'ü baslatıp metin kutusundaki metnin yazımını denetlemek amacıyla CheckSpelling yöntemini harekete geçirmek için, ikinci komut düğmesi ise programı kapatmak için kullanılacaktır.
6. Programdaki nesneler için aşağıdaki özellik ayarlarını yapın:

Nesne	Özellik	Ayar
Form1	Caption	"Özel Yazım Denetçisi"
Label1	Caption	"Metin kutusuna sözcük yada sözcükler girip Yazım Denetimini tıklayın "
Text1	Multiline	True
Text1	ScrollBars	2-Vertical
Text1	Text	(Bos)
Command1	Caption	"Yazım denetçisi"
Command2	Caption	"Son"

Özellikleri ayarladığınızda, formunuzun aşağıdaki gibi olması gerekir:



7. Command1_Click olay yordamini açmak için, Yazım Denetimi komut düğmesini çift tıklayın. Olay yoramına sonraki sayfadaki program kodunu girin:



Bu bildiriler, olay yordamında bir Word Automation nesnesi yaratır, Word'ü başlatır, Word nesnesinin bazı özelliklerini ayarlar, bazı

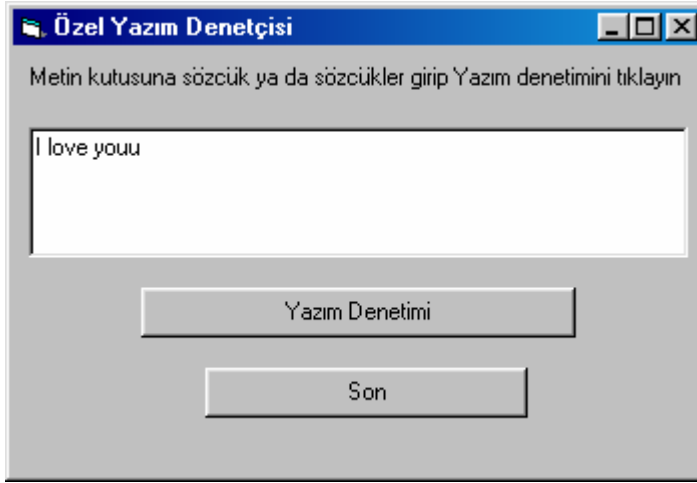
yöntemlerini çağırır ve sonrada nesne tarafından kullanılan belleği serbest bırakır. Nesne degiskenine ilk basvuru yapildiginda, Word otomatik olarak açılır. Sonra Word'ün Selection.Text özelliği, metin kutusunun içeriğini yeni bir Word belgesine kopyalamak için kullanılır.

CheckSpelling yöntemi yürütüldüğünde, Word yazım denetçisini çalıştırır ve metinde yazım hatası olup olmadığını denetler. Eğer bir yazım hatası bulursa, Word Spelling (Yazım Denetimi) iletişim kutusunu açar ve kullanıcıya düzeltme yapmak için bir şans verir. Metin kutusu bir satırdan daha fazla metin içermek için bir şans verir. Metin kutusu bir satırdan daha fazla metin içerse bile, Word metin kutusundaki bütün sözcüklerin yazı denetimini yapar. Denetim bittiginde, düzeltilmiş sözcükler tekrar Visual Basic metin kutusuna kopyalanır ve Word uygulaması kapanır. Command1_Click yordamının sonunda bulunan Set bildirisi nesne degiskeninin isgal ettiği belleğe serbest bırakılır.

8. Command1_Click olay yordamını kapatın ve Son komut düğmesini çift tıklayın.
 9. Olay yordamına End yazın ve Code penceresini kapatın. Özel yazım denetçisi programını tamamladınız.
 10. Formu diskinize Benim Yazım Denetçim.frm adıyla kaydetmek için, Save Form1 As komutunu kullanın. Save Project As komutunu kullanarak projeyi Benim Yazım Denetçim.vbp adıyla diske kaydedin. (Her iki dosyayıda \ AaVb6 \ Ders14 klasörüne kaydedin.)
- Şimdi Automation'ın nasıl çalıştığını görmek için programı çalıştıracağız.

Yazım Denetçim programını çalıştırın

1. Araç çubuğundaki Start düğmesine tıklayın. Bir sonraki sayfada gösterildiği gibi, program programlama ortamında çalışır:

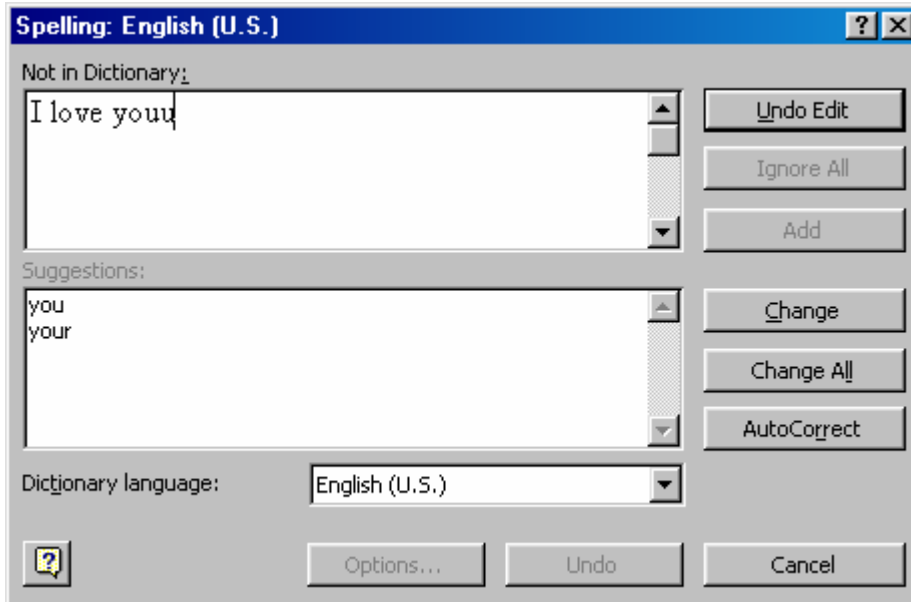


2. Metin kutusuna seni çok seviyorum (I lovve you so mucch) yazın.

NOT : Sizin bilgisayarınızdaki yüklü Word İngilizce ise İngilizce metni, Türkçe ise Türkçe metni yazın.

3. Yazım denetimi düğmesini tıklayın.

Visual Basic bir Automation nesnesi yaratır ve Word'ü başlatır. Bir süre sonra, Word yazım denetimi (Spelling) iletişim kutusu ekrana gelir ve ikinci sözcüğün sözlükte olmadığını bildirir. Ekranınızın aşağıdakine benzer olması gerekir.



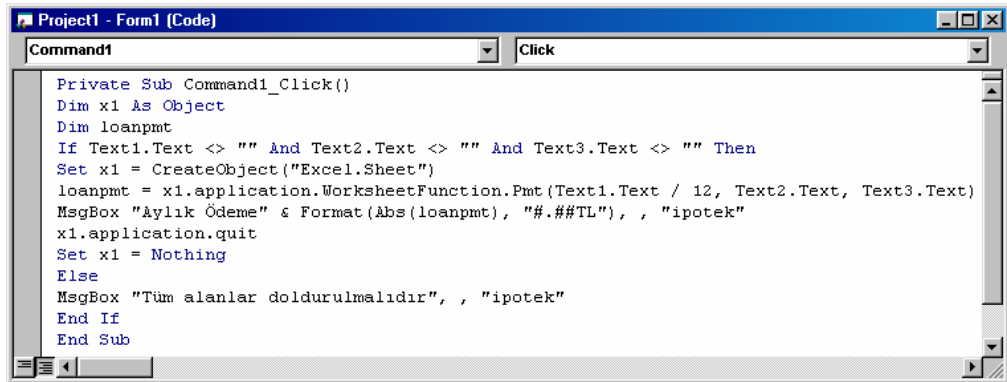
4. Birinci hatayı düzeltmek için, uygun sözcüğü seçip Degistir (Chance) düğmesini tıklayın. Word ikinci yazım hatasını vurguladığı zaman, bu hatayı düzeltmek için aynı işlemi yineleyin.

Yazım denetimi iletişim kutusu kapanır ve metin kutusundaki sözcükler düzelir.

5. Programı kapatmak için Son'u tıklayın.

Visual Basic'ten Excel'i Kullanmak

Microsoft Excel, Visual Basic programlarınızı geliştirebileceğiniz çeşitli karmaşık hesaplama ve veri çözümleme araçları içerir. Aşağıdaki yordam Excel'in Pmt işlevini, bir dizi Visual Basic metin kutusunda belirlediğiniz faiz, borcun ödeneceği ay ve anapara bilgileriyle borcun aylık ödemesini hesaplamak için kullanılır.



```

Private Sub Command1_Click( )
Dim x1 As Object                'Excel için nesne yarat
Dim loanpmp                    'dönen degeri bildirir
'Tüm alanlar deger içeriyorsa
if Text1.Text <> "" And Text2.Text <> ""
And Text3.Text <> "" Then      'nesneyi yarat ve pmt'yi çağır
Set x1 =CreateObject ("Exel.Sheet")
Loanpmt =x1.application.worksheetFunction.Pmt
(Text1.Text) \ 12, Text2.Text, Text3.Text)
MsgBox "Aylık Ödeme" &
Format ( Abs (Loanpmt), "#.= #TL"), , "İpotek"
x1.Application.Quit
Set x1 =Nothing
Else
MsgBox "Tüm alanlar doldurulmalidir", , "İpotek"
    
```

Microsoft Visual Basic 6.0

```
End if  
End sub
```

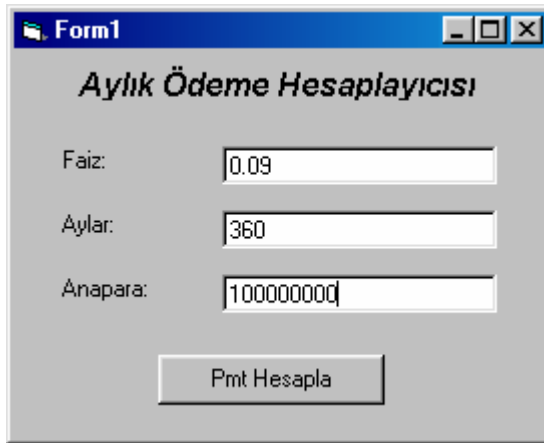
Yordam önce xl adında bir nesne degiskeni yaratir ve onu Excel.Sheet nesnesine atar. Sonra yordam Excel'in Worksheetfunction nesnesini kullanir ve ipotek ödemesini Abs (mutlak deger) ilevi ile arti bir degere dönüştürür. Excel'de borç ödemeleri normalde negatif degerlerle ifade edilirler ama Visual Basic formunda ödemeler pozitif olarak daha iyi görünür. Pmt islevi için gerekli bir bagimsiz degisken eksikse, yordam "Tüm alanlar doldurulmalıdır" iletisini görüntüler.

NOT : Asagidaki adimlari tamamlamak için, bilgisayarınızda Excel 97'nin yüklü olması gerekir. Ipotek projesi ayrıca Excel 8.0 Object Library'ye de önemli bir basvuru içerir. Automation'la Excel kullanan uygulamalar yaratirsaniz, References komutuyla projenize Excel 8.0 Object Library'yi eklemeyi unutmayin.

Ipotek programini çalistirin

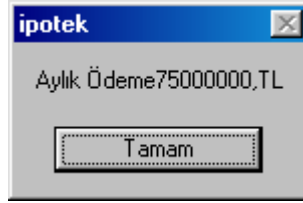
Simdi programi Excel Automation'ın nasil çalistigini görmek için baslatalim.

1. Sabit diskinize \AaVb6\Ders14 klasöründeki ipotek.vbp projesini açin.
2. Run menüsünde Start komutunu tiklayin. Ipotek hesaplayicisi birkaç varsayılan degerle ekranda görünür.
3. Anapara metin kutusuna 100000000 yazin. Formunuz asagidaki gibi görünür.



The screenshot shows a Windows-style window titled 'Form1' with a blue title bar. The main area has a grey background and is titled 'Aylık Ödeme Hesaplayıcısı' in bold. There are three text input fields arranged vertically. The first is labeled 'Faiz:' and contains the text '0.09'. The second is labeled 'Aylar:' and contains the text '360'. The third is labeled 'Anapara:' and contains the text '100000000'. Below these fields is a rectangular button with the text 'Pmt Hesapla'.

4. Pmt hesapla düğmesini tıklayın. Program 100,000,000 TL'lik ibr ipotegin % 9'luk bir faizle 360 ayda ödenmesindeki aylık ödemeyi hesaplamak için Excel'i kullanır. Sonraki sayfada gösterildiği gibi ekranda bir ileti kutusunda sonuc 75.000.000 TL görünür.

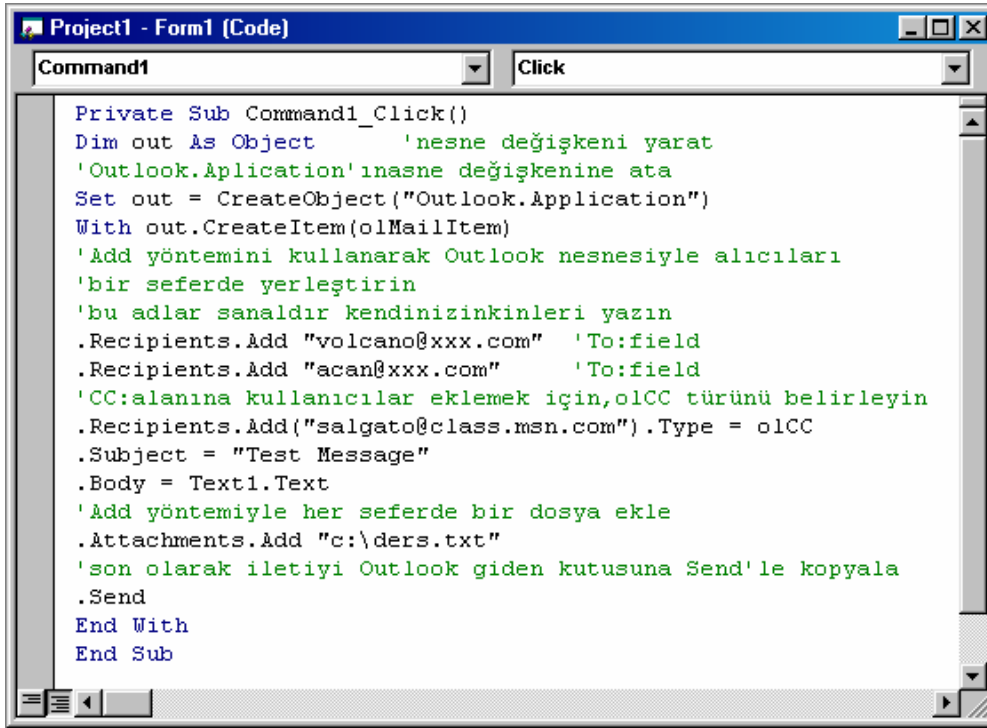


5. OK düğmesini tıklayın ve degisik degerlerle birkaç hesaplama daha yapın.
6. Bitirdiginizde formun baslik çubugundaki Close düğmesini tıklayın. Dilerseniz Code penceresindeki program koduna da bakabilirsiniz.

Visual Basic'ten Outlook'u Otomatiklestirmek

Microsoft Outlook, Microsoft Office'in e-posta,randevu, kisi baglantilari ve is yerindeki takvim ve iletisimle ilgili diger isleri yöneten uygulamasidir. Ben Outlook'u Windows görev çubugunda sürekli çalıştırarak asil e-posta programim olarak kullanıyorum. Yakin zamanda bazı Visual Basic programlarini özellestirip, bu programlarin Outlook uygulamasıyla e-posta gönderebilmesini saglamayi planladik. Bu teknigi bir veriyi (karmasik bir hesaplamanin sonucu,raporlar,veritabanlari ya da yalnızca bir "Merhaba" demek için) otomatik olarak diger kisilere göndermek istediginizde kullanabilirsiniz. Outlook'la posta göndermeningüzel tarafi To, Cc, Subject ve Message alanlarini özellestirebilmeniz ve bir yada daha fazla dosya ekleyebilmenizdir.

Asagidaki alistirmada, Visual Basic'ten Outlook'u kullanarak bir posta göndermek için Mektup Gönder programini kullanacaksınız. Programi çalıştırmadan önce Command1_Click olay yordamindaki kodu, belirli durumlara uygun e-posta adlari kullanabilmek için, asagida gösterildiği gibi degistirmeniz gerekir. Burada verileri kullanmayin, gerçek e-posta adresleri içermedikleri için ve servis saglayicidan hata iletileri alirsiniz.



Simdi Outlook Automation'in nasıl çalıştığını görmek için programı baslatın.

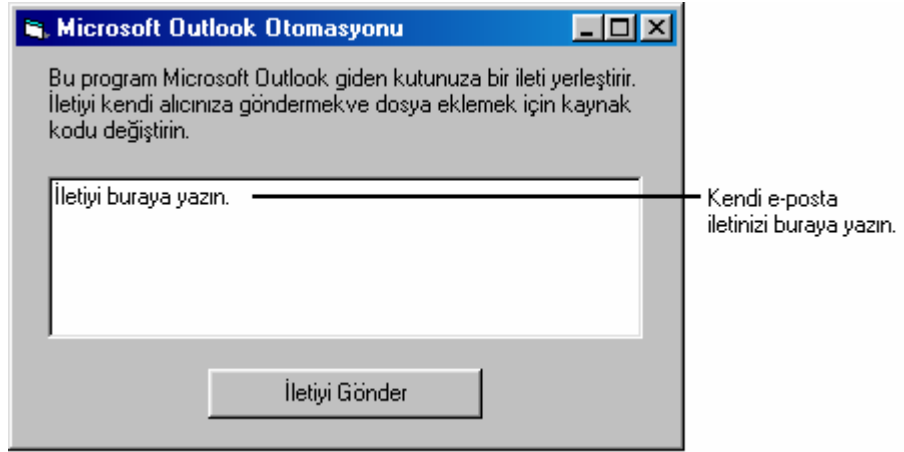
NOT : Asagidaki adımları tamamlamak için Outlook 97 ya da Outlook 98'in bilgisayarınızda yüklü olması gerekir. Mektup gönderme projesi ayrıca Outlook 8.0 Object Library'ye önemli başvuru içerir. Automation'la Outlook kullanan uygulamalar yaratırsanız, References komutuyla projenize Outlook 8.0 Object Library'yi eklemeyi unutmayın.

MektupGönder Programını Çalıştırın.

1. Sabit diskinizdeki \AaVb6\Ders14 klasöründeki MektupGönder.vbp projesini açın.
2. Code projesini açın ve Command1_Click olay yordamını görüntüleyin.
3. Kullanılan üç sanal e-posta adresini (maria@xxx.com, casey@xxx.com ve mike_halfalson@classic.msn.com) gerçekleriyle değiştirin. Birden fazla e-posta adresi kullanmak isterseniz, kullanmak istediğiniz satırdan önce ' karakterini kullanın. Tüm Outlook iletileri için en az bir To alanı gerekir.

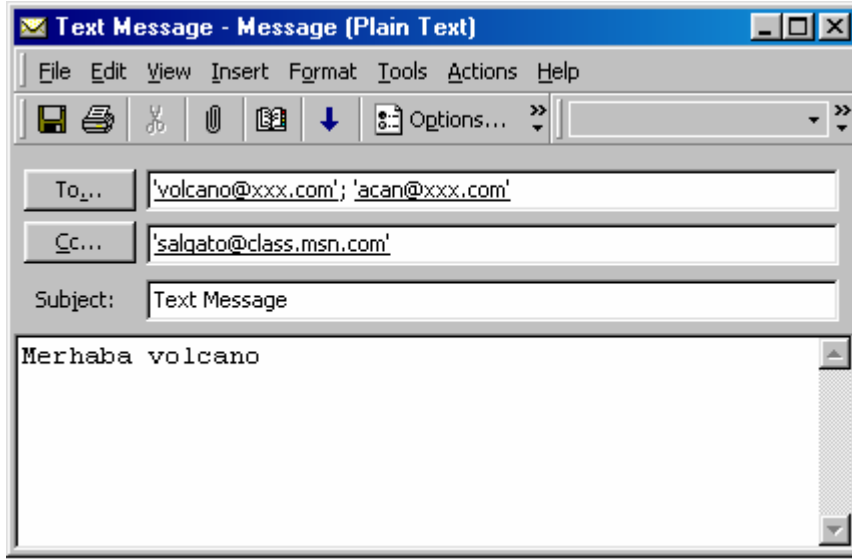
4. Çalışmıyorsa Microsoft Outlook'u başlatmak için Windows Start menüsünü kullanın. Daha sonra Visual Basic'in Outlook'un Outbox klasörüne e-posta iletileri koydugunu doğrulamak isteyeceksiniz.
5. Run menüsünde, Start komutunu tıklayın ve mektup gönder programını çalıştırın.

Programın basit arabirimi görünür:



6. Metin kutusuna bir ileti yazın .
Bu metin e-posta iletinizin gövdesi olarak gönderilecektir. Bu bilgiyi ayrıca programınızın içinden de girebilirsiniz. (Visual Basic programınızdan e-posta göndermek için arabirime gereksiniminiz yoktur.)
7. Şimdi iletiyi göndermek için iletiyi gönder düğmesini tıklayın.
Visual Basic Automation'i Outlook Outbox klasöründe bir mektup iletisi oluşturmak için kullanılır. İleti Outlook'un internet servis sağlayıcınıza bağlanana kadar Outbox klasöründe kalır. (internete bağlı yada yerel ağda çalışıyorsanız Outlook iletiyi hemen gönderir.)
İleti gönderildikten sonra, Outlook iletiyi Outbox klasöründen kaldırır ve kopyasını Sent Items klasörüne yerleştirir, böylece gönderdiklerinizin bir kaydı bulunur.
8. MektupGönder programının başlık çubuğundaki Close düğmesini tıklayın.

9. Simdi Outlook'u yeniden görüntüleyin ve iletiler için Outbox klasörüne bakın.
Bekleyen posta iletinizi asagidaki gibi görürsünüz:
10. Outbox'taki iletiyi çift tıklayın.
Outlook iletiyi açar ve ekranda görüntüler. Gördüğünüz gibi Visual Basic iletiyi yönlendirdiginiz gibi yaratir ve gülen bir yüz resmi ekler (\\AaVb6\\Ders14 klasöründen).

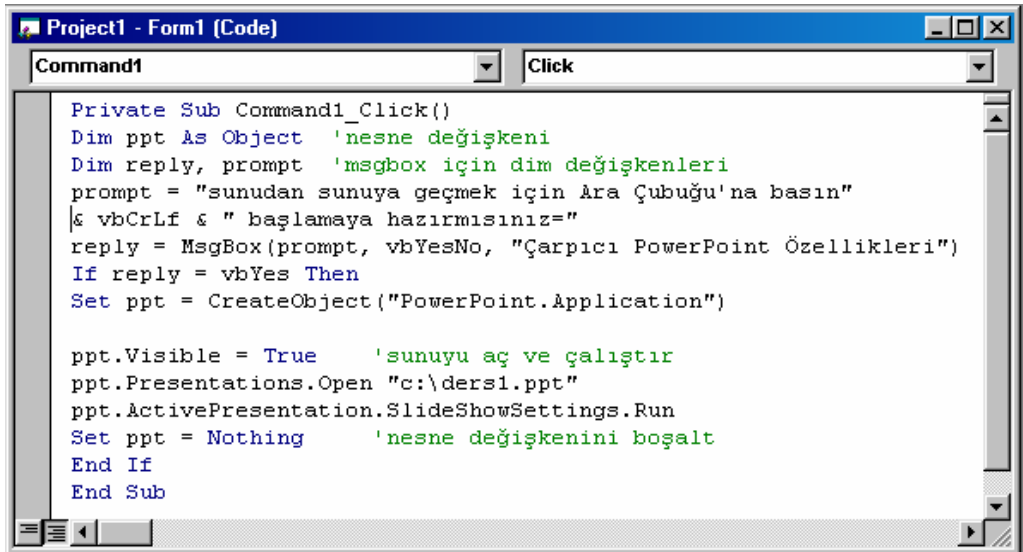


11. Ileti araç çubugundaki Send düğmesini tıklayarak Outbox'ın iletiyi göndermek üzere isaretlemeğini sağlayın.
12. Simdi iletiyi gönderebilir yada silebilirsiniz.
Deneme iletisini silmek istiyorsanız, iletiyi seçin ve Del'e basin. Iletiyi göndermek için Tools menüsünden Check For New Mail'i seçin.

Visual Basic'ten PowerPoint'i Kullanmak

Bu son alıstırmada, Office 97'deki son büyük uygulama olan PowerPoint'te bir sunu gösterisi için Automation'u nasıl kullanacağınızı öğreneceksiniz. PowerPoint sunu gösterileri yaratabileceğiniz, çoklu ortam sunularınızı çalıştırabileceğiniz, özel Web sayfaları yaratabileceğiniz zengin bir sunu programı haline gelmiştir. Hepsinden önemlisi PowerPoint 97 Visual Basic for Applications makro dilini içerir, böylece sunularımızın nasıl yaratılıp gösterildiğini ayarlayan makrolar yazabilirsiniz.

PowerPoint'i, Visual Basic'ten kullanmak Word, Excel ve Outlook'u kullanmaya benzer, kolayca PowerPoint nesne kütüphanesine Reference komutuyla bir köprü yaratin, CreateObject isleviyle bir PowerPoint nesne degiskeni yaratin ve degiskeni PowerPoint komutlarini çalistirmak için kullanin. Command1_Click olay yordaminin asagidaki olay yordami Visual Basic'ten herhangi bir sunuyu nasıl görüntüleyeceginizi gösterir. Kullanicinin bir sunudan digerine nasıl geçtigini anlatmak için kullanıcıya Ara Çubuğu'nun nasıl kullanılacagini gösteren bir ileti kutusu eklenmistir.



```
Project1 - Form1 [Code]
Command1 Click

Private Sub Command1_Click()
    Dim ppt As Object 'nesne değişkeni
    Dim reply, prompt 'msgbox için dim değişkenleri
    prompt = "sunudan sunuya geçmek için Ara Çubuğu'na basın"
    & vbCrLf & " başlamaya hazırmısınız="
    reply = MsgBox(prompt, vbYesNo, "Çarpıcı PowerPoint Özellikleri")
    If reply = vbYes Then
        Set ppt = CreateObject("PowerPoint.Application")

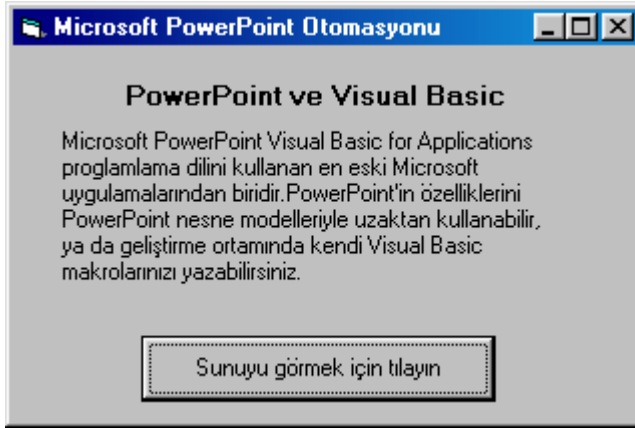
        ppt.Visible = True 'sunuyu aç ve çalıştır
        ppt.Presentations.Open "c:\ders1.ppt"
        ppt.ActivePresentation.SlideShowSettings.Run
        Set ppt = Nothing 'nesne değişkenini boşalt
    End If
End Sub
```

Simdi nasıl çalıştığını görmek için Sunu göster programını başlatın.

SunuGöster Programını Çalıştırın

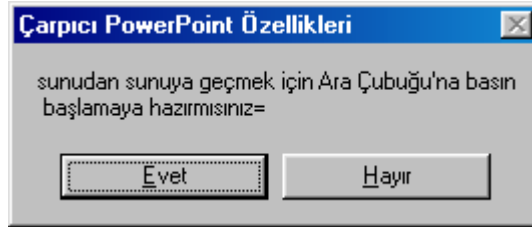
1. \AaVb6\Ders14 klasöründeki sunu göster programını açın.
2. Run menüsünde Start komutunu tıklayın.

Visual Basic programın basit kullanıcı arabirimi görüntüler.



3. Sunuyu çalıştırmak için komut düğmesini tıklayın.

Program sunudan sunuya geçmek için Ara Çubuğu'na basmanızı tavsiye eder ve başlamaya hazır olup olmadığınızı sorar.



4. Sunu gösterisini başlatmak için Yes'i tıklayın.

Visual Basic PowerPoint nesnesini yaratır ve sunuyu başlatarak ilk sunuyu yükler.



5. Sunudaki her görüntüyü ara çubuğu ile birinden diğerine geçerek inceleyin.
6. Bitirdiğinizde, PowerPoint başlık çubuğundaki Close düğmesini tıklayın ve Visual Basic'e dönün.
7. Son olarak SunuGöster başlık çubuğundaki Close düğmesini tıklayın.

Visual Basic'i kullanmaya şimdiye kadar vermek için

- File menüsünde Exit'i tıklayın.
- Save As iletişim kutusunu görürseniz Yes'i tıklayın.

14-Web İçin Dinamik HTML Sayfaları Tasarlamak

Dynamic hypertext markup language(DHTML),World Wide Web Consortium tarafından oluşturulan Microsoft bileşen nesne modeli (COM) ve özelliklerine dayanan gelişmiş bir Internet teknolojisidir. DHTML Page Designer Internet Programcılığı ya da Web sayfa tasarımı hakkında pek fazla deneyiminiz olmasa bile Web uygulamaları yaratabilmenizi sağlar.

Dinamik HTML

DHTML Internet Explorer 4.01 ve daha sonraki sürümlerine dahil edilen bir Internet teknolojisidir. DHTML ile bir Web uygulamasının kullanıcı arabirimini görüntülemek ve geleneksel olarak bir Internet sunucusunun gerektirdiği hesaplama işlemlerini gerçekleştirmek için Internet Explorer'i kullanan bir HTML tabanlı uygulama yaratabilirsiniz. Yarattığınız uygulama bir Internet ya da yerel ağ bağlantısının "istemci tarafında" bir HTML dosyası ve dinamik bağlantı kütüphanesi (DLL) olarak saklanır. Diğer bir deyişle DHTML teknolojisi Internet sunuculara erişim sağlayan Web uygulamaları oluşturmanızı sağlar ama fiziksel olarak son kullanıcının bilgisayarında yer alırlar. Dağıtılan "dinamik" yaklaşımlar DHTML programlarını sunucuda bulunan geleneksel Web uygulamalarından daha iyi tepki verir hale getirir,çünkü DHTML uygulamalarının bilgileri yönlendirmek,verileri saklamak ve istekleri yerine getirmek için uzak bir bilgisayara ulaşmaları gerekmez. Bir DHTML uygulamasında yerel gözetici birçok yordam işleme işlemlerini yapar,sayfanın düzenini değiştirir ve DHTML sayfalarının arkasındaki kodu sunucudan bilgileri yenilemesini istemeden çalıştırır. Bu uygulamanın hızını artırır,web sunucularının iş yükünü azaltır ve (çoğu durumda) son kullanıcıların Internet ya da yerel ağdan yüklemiş oldukları bilgilerle çevrimdisi olarak çalışabilmelerini sağlar.

Visual Basic 6 Professional Edition,Dinamik HTML'i Visual Basic uygulamalarınızla bütünleştirmenizi sağlayan DHTML Page Designer adı verilen özel bir düzenleme bileşeni sunar. DHTML Page Designer'la kendi Web uygulamalarınızı bastan yaratabilir ya da var olan HTML sayfalarını DHTML özellikleri ekleyerek özelleştirebilirsiniz. Bir programlama dili olarak DHTML,en iyi Microsoft Visual Basic Scripting Edition'ın bir uzantisi olarak düşünülebilir ve bu visual basic diliyle çok ortak yönü vardır. DHTML, Visual Basic'le tam olarak uyumlu değildir çünkü önceki HTML standartlarıyla,özellikle de Internet Explorer'in önceki sürümlerinde ki nesne modeliyle uyumlu kalması gerekir. Sonuç olarak,bazı Visual Basic denetimleri,özellikleri,yöntemleri,olayları ve anahtar sözcükleri DHTML tarafından desteklenmez. DHTML Page Designer dilin zorluklarını HTML içeriklerini benzer nesne modeliyle ve geleneksel Visual Basic geliştirme ortamıyla sunarak kolaylaştırır.

Yeni Bir Programlama Teknolojisi

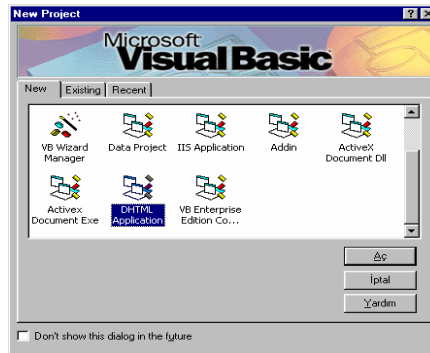
DHTML, Visual Basic'ten biraz farklı bir programlama teknolojisi sunar. Visual Basic formları uygulamalar için temel kullanıcı arabirimi

kullanırken,DHTML kullanıcıya bir yada daha fazla HTML sayfaları aracılığıyla ve destekleyici program koduyla sunar. Bu sayfaları ayrı bir HTML düzenleyicisinde (Microsoft word ya da Microsoft FrontPage) yaratabilir ya da Visual Basic'e eklenen DHTML Page Designer'da bastan yaratabilirsiniz.

Adım Adım DHTML Program Gelistirmek

DHTML Page Designer, Visual Basic program ortamı içinde çalıştığı için,DHTML gelistirmesi işlemi standart bir Visual Basic uygulaması oluşturmaya çok benzerdir. İzlemeniz gereken adımlar :

1. Visual Basic'i baslatın ve yeni bir DHTML Application projesi açın.
2. Project penceresinde,designer klasörünü açın,DHTMPage1 ögesini tıklayın ve View Object düğmesini tıklayın.
3. DHTML proje penceresini web sayfanızı içerecek kadar büyük olacak şekilde boyutlandırın.
4. Web sayfanıza gerekli metni,DHTML araç kutusu öğelerini ve ActiveX denetimlerini ekleyin.
5. Metni biçimlendirme araçlarıyla düzenleyin ve program koduna basvurmasını istediğiniz metin öğelerine ID imlerini atayın.
6. Gerekli her kullanıcı arabirim ögesi için program kodunu yazın.
7. Project menüsündeki AddDHTMLPage komutuyla projenize yeni Web sayfaları ekleyin ve 4 ile 6. adımlarda anlatıldığı gibi metin,denetimler ve olay yordamları ekleyin.
8. Projeyi File menüsündeki Save As komutuyla kaydedin.
9. Visual Basic araç kutusundaki Start düğmesini tıklayarak projeyi çalıştırın ve her özelliği denedinizden emin olun.(programı çalıştırmak için sisteminizde Internet Explorer 4.01 ya da daha sonraki sürümleri kullanılmalıdır.)
10. Uygulamanızı dağıtmak isterseniz,file menüsündeki MakeDHTMLProject.dll komutuyla derleyin. Sonra Windows Start menüsündeki Visual Studio 6.0 Tools klasöründeki Package and Development Wizard'ı kullanarak uygulamanızı dağıtın.



Diger bir farklılık iki dille birleştirilen dosya adı bitisleridir:HTML sayfaları .htm dosyalarında ve Visual Basic formları .frm dosyalarında saklanır.

Bir form gibi bir HTML sayfası metin,grafikler,düğmeler,liste kutuları,ActiveX denetimleri ve veri işlemek ve çıktı görüntülemek için basvurulan diğer nesneleri içerir. Ama bir HTML sayfasını oluşturmak için kullanılan temel denetimler dizisi Visual Basic 6 araç kutusundakilerden farklıdır. DHTML Page Designer,kullanıcı arabirimindeki görevleri HTML özelliklerinde belirtilen kurallara göre yapan öğeler adı verilen biraz farklı programlanabilir nesnelerle sunar. Bu öğelerin her birinin Visual Basic nesnelerindekilerden farklı olan kendine özgü yöntemleri,özellikleri ve olayları vardır. Örneğin,DHTML Button öğesi daha çok Visual Basic CommandButton denetimi gibi görünüp,çalışmasına rağmen,Command1_Click olay yordamı yerine,Button1_Onclick olay yordamını çalıştırır.

DHTML Page Designer ile Çalışmaya Başlamak

DHTML uygulamaları yaratmayı öğrenmenin en iyi yolu DHTML Page Designer'la biraz el alıstırması yapmaktır. Bu kısımda Visual Basic'te Page Designer'ı açacak ve Internet uygulamanız için temel olacak biçimlendirilmiş metinle bir HTML sayfası yaratacaksınız.

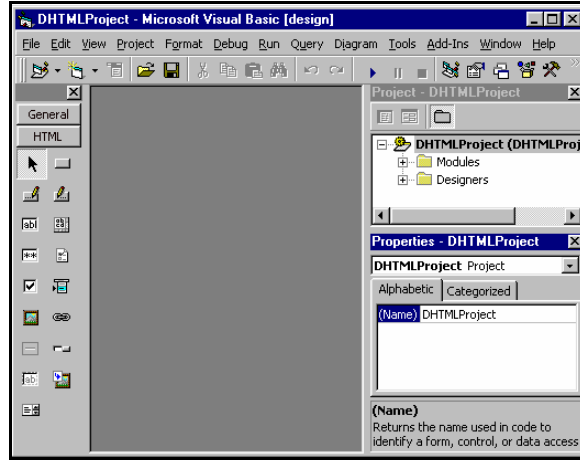
Yeni Bir DHTML Uygulaması Açmak

Visual Basic'te başlatmak ve DHTML Page Designer'da yeni bir DHTML uygulaması açmak için aşağıdaki adımlar izlenir:

1. Visual Basic'i başlatın.
2. New Project iletişim kutusunda,DHTML Application simgesini ve Open'i tıklayın.

Yeni bir proje açmak için DHTML Application simgesini kullandığınızda, Visual Basic DHTML Page Designer'ı yükler ve derleyiciyi ActiveX dinamik bağlantı kütüphanesi (.dll) yaratmak için yapılandırır. Bir ActiveX .dll Dynamic HTML komutlarına basvuran bir HTML sayfasına nesneler ve hesaplama kaynakları yaratan bir dosyadır. (Visual Basic program kodunuz bu .dll'ler de saklanır.)

Page Designer açıldığında ekranınız aşağıdaki gibi görünür:



3. Project penceresinde Designers klasörünü açın.

Projedeki (DHTMLPage1) varsayılan HTML sayfası Designers klasöründe görünür. Bir tasarımcı (designer) uygulamanızın kullanıcı arabiriminin metin,denetimler,diger öğelerini içeren tek bir HTML sayfasıdır. (Microsoft terminolojisinde,bir tasarımcı Visual Basic uygulamanızın bir parçasını yaratan özel bir araçtır. Bu kitapta tasarımcıları DHTML uygulamaları ve ActiveX veri nesneleri yaratmak için kullanacaksınız.) Uygulamana birden fazla HTML sayfası dahil etmek istiyorsanız,project menüsündeki AddDHTML page komutuyla Designers klasörüne ek tasarımcılar ekleyebilirsiniz.

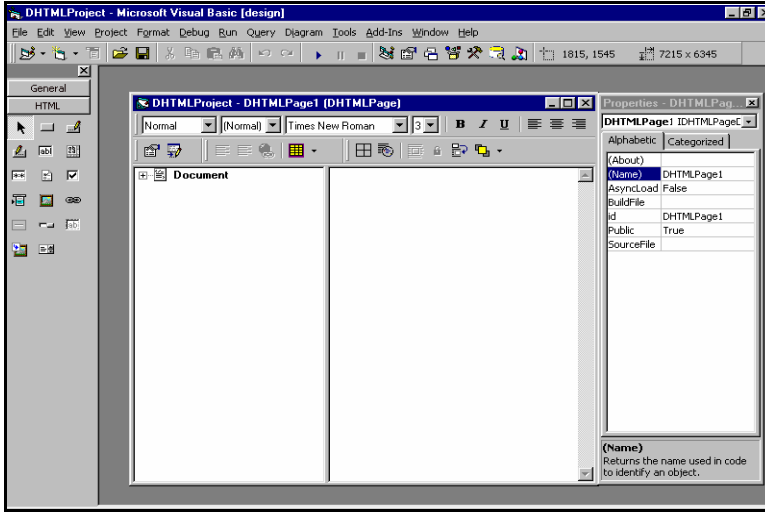
4. Designers klasöründeki DHTMLPage1 tasarımcısını tıklayın ve Project penceresindeki View Object düğmesini tıklayın.

Visual Basic programlama ortamında DHTML Page Designer'ı görüntüler. Varsayılan ayar olarak Page Designer tam boyutunda açılmadığı için,onu şimdi büyütme ve Project,Properties ve Form Layout pencerelerinin kapladığı alanı küçültmelisiniz.

5. Fareyi isaretçi boyutlandırma isaretçisine dönüşene kadar Project penceresinin sol kenarının üzerine getirin ve pencere sinirini program ortamında kapladığı alanı küçültmek için saga sürükleyin.

Simdi DHTML Page Designer'ı genişleteceksiniz.

6. Fareyi Page Designer'ın sag alt kenarının üzerine getirin. Fare isaretçisi boyutlandırma isaretçisine dönüştüğünde,Page Designer'ı asagıdaki gibi görüne kadar genişletin:

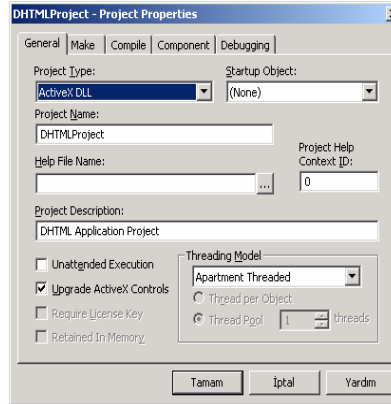


- Varsayılan sayfa, DHTMLPage1 Page Designer'in sağ bölümünde görünür. (Geçerli olarak sayfa bostur.) Sol bölümde belgenizin HTML kodunun ağaç görünümü açıklaması bulunur. HTML sayfanıza metin, denetimler ve biçimlendirme efektleri ekledikçe, seçtiğiniz stiller ve kullandığınız araçlar bu bölmedeki hiyerarsik yapıya yansitilir.

Page Designer'daki iki bölmenin üzerinde, HTML sayfanızdaki öğeleri biçimlendiren, konumlandıran ve düzenleyen düğmeleri içeren biçimlendirme araç çubuğu bulunur. Page Designer'ın sol tarafında, HTML sayfalarınıza programlanabilir öğeler olarak ekleyebileceğiniz asıl DHTML denetimlerini içeren araç kutusu bulunur.

DHTML Page Özellikleri

DHTML sayfası özellikleri hazırlanan sayfanın Visual Basic projesi içinde ya da bir HTML dosyası olarak kayıt edilmesini sağlar.



Bir HTML Sayfasına Metin Ekleme

Denetimler ve özel biçimlendirme efektleri sıkça Web'deki HTML sayfalarına eklence katsa da,iyi tasarlanmış bir HTML uygulamasının esasi yapısal olmayan metinsel arabirimdir. Sonraki adımlarda Page Designer'ın sağ bölmesindeki HTML sayfasına Sansim 7 Oyunu programı için metinsel öğeler ekleyeceksiniz.

1. Page Designer'da HTML sayfasına (sağ taraftaki sayfa) tıklayın. HTML sayfasının üst tarafında yanıp sönen bir imleç görünür ve belgenizin bir ağaç görünümü semasi Page Designer'ın sol bölümünde açılır.
2. Asagidaki metni,bosluklari dogru koymak için [parantez] içindeki yönergeleri dogru izleyerek HTML sayfasına yazın:

Sansim 7 Oyunu [Enter][Enter][Enter]
0 [Space][Space][Space] 0 [Space] [Space] [Space] 0 [Enter]
Kazanma: [Enter]

Sansim 7 Hakkında

Her [Enter] yönergesi için,Enter tusuna basın. Her [Space] yönergesi için bir kez bosluk tusuna basın.

NOT: Visual Basic'te Page Designer'i kullanırken belirli HTML biçimlendirme imlerini bilmeniz gerekmez. Ama,belgenizin nasıl

düzenlendiği hakkında daha fazla bilgi vermek için zaman zaman ağaç görünümü bölmesinde HTML imleri görürsünüz.

Simdi ağaç görünümü bölmesindeki BODY isaretinin üzerinde çalıştığınız HTML sayfasının başlangıç içeriğini içerdigine dikkat edin. HTML'de BODY imi Web sayfasının temel grafik görünümü ve davranisini denetleyen yönergeleri içerir.

Page Designer'daki Metni Biçimlendirme

DHTML'de stiller HTML'de özgün olarak bir Web sayfasındaki metin öğelerini biçimlendirmek için kullanılan biçimlendirme imlerinin yerini alır.(HTML biçimlendirme imleri yine vardır ama Page Designer onları projeniz için bir .dsx dosyasında saklar.) Bir stil HTML belgesindeki öğelerin görünüsünü denetleyen bir özellikler yiginidir. Stil sayfaları bir stili tek bir öğeye ya da bir öğeler grubuna uygulayabilir. Ayrıca,sayfadaki her öğeye birden fazla stil uygulayabilirsiniz (bir başlık ve köprü stili gibi).

Aşağıdaki adımlarda,daha önce Sansim 7 uygulaması için Page Designer araç çubuğundan biçimlendirme stilleriyle yazdığınız metni biçimlendireceksiniz.

- 1) HTML sayfasının üst tarafındaki Sansim 7 Oyunu metnini seçin.

Metnin stilini Page Designer'da değiştirmeden önce onu seçmelisiniz.

- 2) Page Designer'ın sol üst köşesindeki Style açılır liste kutusunu tıklayın ve Heading 1 stilini gösterin.
- 3) Sansim 7 Oyunu metnini Heading 1 stilinde biçimlendirmek için Heading 1 stiline tıklayın.

Page Designer,metni genişleten Heading 1 stilini uygular.

NOT: Page Designer stilleri Microsoft Word ve diğer sözcük işleme uygulamalarındaki stillere benzerdir.

Simdi HTML sayfanızda birkaç stil daha kullanarak listirmalar yapacaksınız.

- 4) Belgede ağaç görünümü bölmesinde ilk bos paragrafi tıklayın P() ve Heading 2 stiliyle biçimlendirin.(P() imlerini görmezseniz,ağaç görünümü bölmesinde BODY ögesinin yanındaki artı simgesini tıklayın ve ilk P() imini seçin ve Heading 2 stiliyle biçimlendirin.)

HTML sayfasındaki metni seçmenin yani sıra,ağaç görünümü bölmesinden tek tek satırları seçip biçimlendirebilirsiniz. Heading 2 stilini tıkladıktan sonra,Page Designer yazı tipini büyütür.(ama satır bos olduğu için,tek fark edilebilen etki satırlar arasındaki boşlukta küçük bir artistir.)

- 5) Belgedeki sonraki bos paragrafi P() tıklayın ve Heading 2 stiliyle biçimlendirin.

Belgenizdeki metin öğelerine biçimlendirme stilleri uyguladıkça,Page Designer'ın biçimlendirme seçimlerini iki karakterlik bir kodla belirttiğine (H1,H2 vb.) ve ağaç görünümü bölümünün alt kısmına tasidigina dikkat edin.(Daha önce HTML ile çalıştıysanız,bu başlık imleri size tanıdık gelecektir.)

- 6) Belgedeki üç sifiri seçin (0,0,0) ve Heading 1 stiliyle biçimlendirin.

- 7) Kazanma:metnini seçin ve Heading 3 stiliyle biçimlendirin.

- 8) Sansim 7 Hakkında metnini seçin (HTML sayfasının altında) ve Heading 4 stiliyle biçimlendirin.

NOT: Keskettiginiz varsayılan biçimlendirme stillerinden baska Page Designer'daki Font Size ve Font Name açılır liste kutularını HTML sayfanızın yazı tipini degistirmek için kullanabilirsiniz. Ayrıca araç çubugundaki Bold,İtalik ve Underline düğmelerini yazı tipi stilini ve Left,Center ve Right düğmelerini sayfanın hizalamasını degistirmek için kullanabilirsiniz.

Karakterleri Ayırmak İçin SPAN İmlerini Kullanmak

Visual Basic'te DHTML uygulamaları yazdığınızda, ortak bir görev özellik ayarlarını kullanarak bir Web sayfasındaki tek tek metin yada karakterlerin içeriğini değiştirmektedir. Örneğin, belirli bir ayda Web bölgesini ziyaret edenlerin sayısını bildiren sayıyı güncellemek isteyebilirsiniz. HTML sayfanızı yaratırken Visual Basic koduyla hangi sözcükleri ya da karakterleri ayırmak istediğinizi biliyorsanız, Page Designer'ın dizilimi SPAN imleriyle çevrelemesini sağlayabilirsiniz. Bu imleri HTML belgenizde görmezsiniz ama Page Designer'da "ekranın arkasında" bulunurlar ve kaydedildiklerini ağaç görünümü bölümünde görürsünüz.

Sansim 7 uygulamasındaki üç bölgeyi SPAN imleriyle tekrar belirlemek, böylece daha sonra olay yordamında üç rastgele sayıyla değiştirebilmek için şu adımları izleyin:

- 1) Belgedeki ilk 0'i seçin ve Page Designer araç çubuğundaki Wrap Selection In SPAN düğmesini tıklayın.

Metni seçmek için klavyeyi ya da fareyi kullanabilirsiniz. (Seçtiğiniz karakterler boşluklarla çevriliyse klavyeyi kullanmak daha kolay olacaktır.)

- 2) Belgedeki ikinci 0'i seçin ve Page Designer araç çubuğundaki Wrap Selection In SPAN düğmesini tekrar tıklayın.

- 3) Belgedeki üçüncü 0'i seçin ve Page Designer araç çubuğundaki Wrap Selection In SPAN düğmesini son kez tıklayın.

Ağaç görünümü bölümünde metin iminin yanında yarattığınız SPAN imlerinin belirtisi olarak bir (+) simgesinin belirdiğine dikkat edin.

- 4) SPAN imlerini görmek için ağaç görünümü penceresinde artı simgesini tıklayın.

NOT: HTML sayfalarındaki SPAN imleriyle ayrılan öğelerin yani sıra, paragrafları DIV imiyle birbirine bağlayabilirsiniz. Bir DIV imi karmaşık bir işlemde çeşitli öğeleri aynı stilde biçimlendirmek istediğinizde yararlıdır.

Bir DIV ögesindeki hersey ayni biçimlendirilmeye sahip oldugundan bir Web sayfasindaki bilgileri gruplandirmanin iyi yoludur. DIV imi için araç çubugu düğmesi,SPAN düğmesinin yanında Wrap Selection In <DIV>...</DIV> adiyla yer alır.

Properties Penceresiyle ID Öznitelikleri Atama

Visual Basic programinda,uygulamanizin kullanıcı arabirimindeki her ögenin derleyicinin çalışma zamanini islerken kullandigi özgün bir adi vardır. Örneğin,formdaki ilk metin kutusu nesnesi Text1 adini,ikincisi Text2 vb. adlarini alirlar. Bir Dinamik HTML uygulamasinda da sayfadaki her ögenin,program koduyla yönetmek istiyorsanız,özgün bir adi ya da ID özniteliği olmalıdır. DHTML uygulamanızda sayfanızdaki çeşitli öğelere atadığınız her öznitelik ögesinin nesne adi gibi çalışır.

HTML sayfanızdaki her metinsel her öğeye ID öznitelikleri atamak için aşağıdaki adımları izleyin.

1. Ağaç görünümü bölmesindeki Sansim 7 Oyunu başlığını tıklayın (H1 imi).
2. Görünmüyorsa ya da başka bir pencere tarafından kapatılıyorsa program penceresini açın ve büyütün.
3. Program penceresinde ID girdisinin yanındaki metin kutusunu tıklayın ve LuckyHead yazın ve Enter'a basın.

Page Designer seçili metnin ID niteligini LuckyHead'a ayarlar. Bu DHTML uygulamasinda başligi programla denetlemenize ragmen,kullandiginiz paragraflarin her birini adlandırmak iyi bir programlama alistirmasidir.

4. Ağaç görünümü penceresinde ikinci başligi tıklayın (belgedeki ilk bos satir) ve Properties penceresiyle ID niteligini Blank1'e degistirin.
5. Ağaç görünümünde üçüncü başligi tıklayın ve ID niteligini Blank2'e degistirin.
6. Dördüncü başligi tıklayın (üç sayinin tümünü alın) ve ID niteligini Num'a degistirin.

Num niteligi (bir küme adi gibi) üç sayiya da uygulanir. Ama SPAN imleriyle belirtilen dizilimlere de bireysel ID'ler atayabilirsiniz.

7. İlk SPAN sayisini tiklayin ve ona ID Num1'e atayin.
8. Ikinci SPAN sayisini tiklayin ve ona ID Num2'yi atayin.
9. Üçüncü SPAN sayisini tiklayin ve ona ID Num3'ü atayin.
10. Kazanma basligini tiklayin ve ona ID Result'i atayin.

Son metin basligi (Sansim 7 Hakkinda) simdilik bir ID niteligi gerektirmez. Sonraki alistirmada bunu bir köprü olarak biçimlendireceksiniz.

Baska bir HTML sayfasina köprü yaratmak

Web uygulamalariniz birden fazla HTML sayfasi sunacaksa, Page Designer araç çubugundaki Make Selection Into Link düğmesi aklınızda tutmak istediginiz yararli bir biçimlendirme araci olacaktir. Make Selection Into Link düğmesi, kullanıcı köprüyü tikladiginda Internet Explorer'daki geçerli sayfayi degistirerek yeni bir HTML sayfasi yükleyerek seçili metni köprü olarak biçimlendirir. Bir metin seçimini köprü olarak biçimlendirdikten sonra Properties penceresinde metin ögesinin href özelligini kullanarak gerekli baglantiyi (bir URL ya da yerel yol adi) belirleyebilirsiniz.

Sansim 7 Hakkinda metnini köprü olarak biçimlendirmek için asagidaki adimler izlenir:

1. Sayfadaki son basligi seçmek için ağaç görünümü bölmesindeki H4'ü tiklayin. Metni HTML sayfasinda biçimlendirmeden önce seçmelisiniz.
2. Page Designer araç çubugunda Make Selection Into Link düğmesini tiklayin. Seçili baslik sag bölmede köprü olarak biçimlendirilir.
3. Simdi HTML belgesinde baska bir satiri tiklayin. Imleç köprüyü içeren satirdan uzaklastiginda köprü biçimlendirmesi (altı çizili

mavi metin stili) görünür olur ve bir (+9) simgesi ağaç görünümü bölmesinde görünür.

4. Köprü (Hyperlink 1) için ID özneliği görmek için ağaç görünümü bölmesinde artı simgesini tıklayın.
5. Köprünün özelliklerini Properties penceresinde görmek için ağaç görünümü bölmesinde Hyperlink1 imini tıklayın.
6. Properties penceresinde href özelliğini bulun ve özelliğin sağındaki onay kutusunu işaretleyin.
7. Köprü için href özelliği girdisi olarak **C:\AaVb6\Ders22\Sansim.htm** yazın ve Enter'a basın.

WebSansim Projesini Kaydetme

1. File menüsünde Save Project As komutunu seçin.
2. Proje dosyanızın (.vbp) adı sorulduğunda WebSansim yazın
3. Tasarımci dosyanızın (.dsr) adı sorulduğunda WebSansim yazın.

Burada ilk defa kullanılan tasarımci dosya,HTML sayfanızı ve tüm biçimlendirmesi ve denetimlerini içeren özel bir dosyadır.

4. Kod modülü dosyanızı (.bas) adlandırmanız istendiğinde WebSansim yazın.

Bir DHTML uygulamasında,.bas dosyası gözatici bir sayfadan diğerine geçerken önemli bilgileri kaydetmek ve geri almak için bir mekanizma olan PutProperty ve GetProperty işlemlerini yöneten işlevleri içerir.

DHTML Uygulamasını Çalıştırma

DHTML uygulamanız güvenli olarak sabit diskinize yüklendiğinde, Internet Explorer'la çalıştırabilirsiniz.

1. WebSanim programini çalistirmek için Visual basic araç çubugundaki Start düğmesini tiklayin.

Bu uygulamayi ilk defa çalistirdiginizda,bir Project Properties iletisim kutusu görebilirsiniz.

Bu kutu uygulamaniza Page Designer'da yarattiginiz HTML formunu yükleyerekmi baslamak istediginizi sorar. Dogru HTML bilezeni yeni belirlenmistir,yalnizca bu iletisim kutusunu görürseniz OK'i tiklayin.

Visual Basic Internet Explorer'i yükler ve DHTML uygulamanizi görüntüler.

Normalde,Internet Explorer'in HTML sayfanizi gösterme biçiminde çok az biçimlendirme farklıliklari görürsünüz. Bu noktada,uygulamaniz yarimdir,programla iletisim kuramazsiniz ama yarattiginiz biçimlendirme ve köprü efektlerini gözaticinin nasıl görüntüledigini görebilirsiniz.

NOT: Gözaticinizin address metin kutusundaki garip yol adini gördünüz mü?Visual Basic DHTML uygulamanizi bellekte derlerken,sabit diskinizde program çalisirken saklamak için geçici bir dosya yaratir. Gördüğünüz yol adi Visual Basic'in dosyayi saklarken kullandigi geçici saklama konumudur.

2. Internet Explorer'i kapatmak için Internet Explorer'in baslik çubugundaki close düğmesini tiklayin. Internet Explorer kapanir ama uygulamaniz çalismaya devam eder,çünkü Internet Explorer basit bir gözaticidir,programin kaynagi degildir.

3. DHTML uygulamasini durdurmak için Visual Basic araç çubugundaki End düğmesini tiklayin. Bir süre sonra program kapanir ve Page Designer programlama ortaminda tekrar görünür.

Microsoft Word'de HTML Belgeleri Yaratmak

Dinamik HTML sayfalarını Visual Basic içerisinde DHTML Page Designer kullanarak yaratabilir ya da HTML belgelerini bir di editör ya da kelime işlemcisi kullanarak yaratabilir ve belgeleri doğrudan programlama projenizle birleştirebilirsiniz. DHTML denetimleri üçüncü grup ActiveX denetimleriyle gelişmiş biçimlendirme etkileri yaratmak isterseniz,DHTML Page Designer çok kullanışlıdır. Ancak,sadece HTML belgelerinize metin giriyorsanız HTML belgenizi başka bir editörde yaratmak ve daha sonra projenize eklemek daha yararlıdır.

Sansli.htm Yardım Dosyasını Yaratmak İçin Word'ü Kullanma

1. Visual Basic geliştirme ortamını simge durumuna küçültün ve bilgisayarınızda Microsoft Word 97'i başlatın.

Microsoft Word'ü Start menüyü tıklayıp,Programs klasörünü gösterip Microsoft Word 97 klasörünü tıklayarak başlatabilirsiniz.

2. Word başlayıp yeni boş bir belge görüldüğünde yazacağınız metni yazın ve gösterildiği gibi biçimlendirin.

Şimdi bu belgeyi bir HTML belgesi olarak kaydedeceksiniz,böylece WebSansim uygulaması tarafından görüntülenebilir.

3. Word File menüsünde,Save As HTML komutunu tıklayın.
4. Save As iletişim kutusu görüldüğünde,C:\AaVb6\Ders22 klasörünü belirleyin ve HTML belgenizin yol adı olarak Sansim yazın.

Dosyayı Word 97 biçiminde kaydetmeden devam etmek isteyip istemediğiniz sorulabilir ya da Web araçlarının yeni sürümlerini yüklemek için Internet'e bağlamak istiyorsanız ilk önce Yes'i ve ikincisinde No'yu tıklayın.

5. World File menüsünde,Word'ten çıkmak için Exit komutunu tıklayın.

Simdi WebSansim programini etkin bir köprüyle nasıl çalışacağını görmek için çalıştıracaksınız.

6. Visual Basic'in ekranı kaplamasını sağlayın.

NOT: Bu yönergeleri tam olarak takip ediyorsanız,programi çalıştırmadan önce hyperlink nesnesinin href özelliğini C:\AaVb6\Ders22\sansim.htm'e değiştirmelisiniz. Böylece yarattığınız HTML dosyasını açar. Href özelliği,bu kısımdaki programları kolayca yükleyip çalıştırabilmeniz için C:\AaVb6\Ders22\sansim.htm'e gönderilmiştir.

7. Gerekliyse hyperlink 1 nesnesinin href özelliğini değiştirin ve sonra Visual Basic araç çubuğundaki start düğmesini DHTML programını çalıştırmak için tıklayın.

8. Websansim sayfası Internet Explorer'da,bagli sayfayı görmek için Sansli 7 Hakkında köprüsünü tıklayın.

9. Websansim HTML sayfasına dönmek için Back düğmesini tıklayın.

10. Internet Explorer başlık çubuğundaki close düğmesini tıklayın.

11. Visual Basic araç çubuğundaki End düğmesini tıklayın.

Microsoft Word (ya da başka bir HTML düzenleyici yada sözcük işlemci) kullanmak Visual Basic'teki DHTML Page Designer'a yardımcı bir kaynaktır.

DHTML Sayfalarına Öğeler ve ActiveX Denetimleri Ekleme

Araç kutusu Öğeleriyle Çalışmaya Başlamak

DHTML Page Designer,Web sayfanızın kullanıcı ara birimini geliştirmek için kullanabileceğiniz, DHTML denetimleri ya da öğeleriyle bir araç kutusu içerir. Bu araç kutusu öğeleri Microsoft Visual Basic denetimleriyle çok benzerlik göstermesine rağmen aynı değildir.

DHTML araç kutusu öğeleri HTML programlama standartlarıyla uyumlu nesneler yaratır, böylece Microsoft Internet Explorer bu standartları kullanan Web sayfalarını görüntüleyebilir. DHTML araç kutusu öğeleri Visual Basic denetimlerinden farklı özellikler ve yöntemler sağlar ve ayrıca farklı olaylara tepki verirler. Son olarak, DHTML denetimleri boyut ve izin anahtar etkenler olduğu Web'te çalıştırılmak için ayarlanmıştır. Aşağıda DHTML araç kutusundaki öğeler görünür:



NOT: Visual Basic araç kutusu, DHTML araç kutusu öğeleri ve Visual Basic araç kutusu denetimleri arasında geçiş yapmanızı sağlar. Öğeler ve denetimler arasında geçiş yapmak için sırasıyla araç kutusundaki DHTML ve General düğmelerini tıklayın.

DHTML Araç Kutusunu Belgelendirmek

Bir DHTML uygulamasında araç kutusu denetimlerinin nasıl kullanıldığıyla ilgili daha fazla bilgi için MSDN Library çevrimiçi yardımının index sekmesinin **HTML Intrinsic Controls** yazın.

NOT: Aşağıdaki bilgiler DHTML araç kutusu öğeleri tarafından sağlanan en önemli özelliklerin ve olayların bir kaçıdır. (Tüm liste için, her öğenin Properties penceresini ve kod penceresindeki procedur açılır ,liste kutusu denetlenir.)

Button

Button ögesi DHTML sayfasında bir komut düğmesi yaratır. Value özelliği düğmede yer alan metni saklar ve OnClick olay yordamı kullanıcı düğmeyi her tıkladığında çalışır. Düğme ögesi normalde formdaki veri girdisinde, yeni değerlerin hesaplanmasında yada web sayfalarının kapatılmasında kullanılır.

SubmitButton

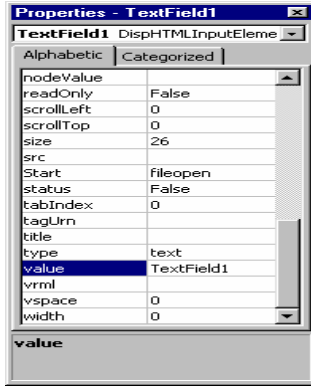
SubmitButton ögesi de DHTML sayfalarında bir komut düğmesi yaratır. Ama SubmitButton normalde bir web sayfasındaki bilgiyi, bir Internet sunucusu gibi, bir back-end işlevine geçirmek için kullanılır. Value özelliği düğmenin üzerinde beliren metni saklar ve OnClick olay yordamı kullanıcı ögeyi her tıkladığında çalışır.

ResetButton

ResetButton ögesi de DHTML sayfasında bir komut düğmesi nesnesi olarak yer alır. Ama, ResetButton ögesi geçerli sayfadaki tüm metin ögelerini temizler. Value özelliği düğmenin üzerinde beliren metni saklar ve OnClick olay yordamı kullanıcı ögeyi her tıkladığında çalışır.

TextField

TextField ögesi bir DHTML sayfasında çalışma zamanında metin girdisi alabilen tek satırlık bir metin kutusu yaratır. Bu metin kutusu ögesi daha çok Visual Basic TextBox denetimi gibi çalışır. Value özelliği metin kutusunda görünen metni saklar. Bu özelliği tasarım zamanında Properties penceresiyle tanımlayabilir yada özelliği çalışma zamanında kullanıcının ne girdiğini görmek için okuyabilirsiniz. TextField ögesi metin kutusunun içeriği seçildiğinde OnSelect olayında ve metin kutusundaki metin değiştirildiğinde OnChange olayını çalıştırır.



TextArea

TextArea ögesi DHTML sayfasında çok satırlı girdi ve çıktıya izin veren daha büyük bir metin kutusu yaratır. Gerekirse TextArea ögesi gizli satırlara erişim sağlamak için kaydırma çubukları sunar. Value özelliği metin kutusunda görünen metni saklar. Rows özelliği ögenin yüksekliğini satır cinsinden ayarlayabilmenizi sağlar ve Cols özelliği ögenin genişliğini karakter cinsinden ayarlamamanızı sağlar. TextField ögesi gibi TextArea ögesi de metin kutusunun içeriği seçildiğinde OnSelect olayında ve metin kutusundaki metin değiştirildiğinde OnChange olayını çalıştırır.

PasswordField

PasswordField DHTML sayfasında bir parolayı yada kullanıcının girdiği önemli bilgiyi gizleyen yada maskeleyen bir kutusu yaratır. Value özelliği parola kutusuna girilen metni saklar. Properties penceresi ile tasarım zamanında gizli bir varsayılan parola belirleyebilir yada çalışma zamanında kullanıcının parolasını almak için value özelliğini kullanabilirsiniz. PasswordField ögesi parola kutusunun içeriği seçildiğinde OnSelect olayını ve parola kutusundaki metin değiştirildiğinde OnChange olayını kullanır.

Option

Option ögesi DHTML sayfasında bir seçenek düğmesi yaratır. Visual Basic'de her düğmeyi formdaki bir frame denetiminin içine yerleştirerek beraber çalışan bir grup seçenek düğmesi yaratabilirsiniz.

Ama DHTML Page Designer'in bir frame denetimi yoktur. Seçenek düğmelerinin biri DHTML sayfasında gruplandırmak için, properties

penceresi kullanılmalıdır. Properties penceresinde, önce her düğmenin name özelliği aynı değere ayarlanır ve her düğmenin ID özelliği özgün bir ID gönderisine ayarlanır. Checked özelliği grupta bir düğmenin varsayılan ayar olarak seçilmesini

saglar ve OnClick olayı kullanıcı tek bir düğme ögesini tıkladığında çalışır.

Checkbox

Checkbox ögesi bir DHTML sayfasına bir onay kutusu eklemek için kullanılır. Visual Basic onay kutusundan farklı olarak, DHTML Checkbox ögesi onay kutusunu açıklayan dahili bir etiket içermez. Checked özelliği onay kutusunun geçerli durumunu belirler. Onay kutusuna bir onay isareti koymak için Checked özelliğini True'ya yada kaldırmak için False'a ayarlayın. OnClick olayı bir onay kutusu ögesi tıklandığında çalışır.

Select

Select ögesi bir DHTML sayfasına bir açılır liste kutusu yada karma kutu ekler. Visual Basic'teki ComboBox denetimine benzer. Size özelliği karma kutuda bir seferde görülebilen öğelerin sayısını belirler ve selected özelliği bir öğenin varsayılan olarak seçilmesini sağlar. Çalışma zamanında select ögesi değiştirildiğinde OnChange olayı çalıştırılır.

Image

Image ögesi bir DHTML sayfasına standart bir grafik görüntü eklemek için kullanılır. Tasarım yada çalışma zamanında bir görüntü eklemek için, yüklemek istediğiniz yol adını Src özelliği ile ayarlayın. Ayrıca title özelliğini kullanıcı sayfadaki Image ögesini tıkladığında açılır metin eklemek için kullanabilirsiniz.

Hyperlink

Hyperlink ögesi uygulamanızdan başka bir HTML sayfasına köprü kurmak istediğinizde kullanılır. Köprüyü kurmak istediğiniz HTML belgesi yada URL'in adını belirlemek için properties penceresinden href özelliğini ayarlayın. Sayfanızdaki biçimlendirilmiş köprüde yer alan metnin hyperlink ögesi ile denetlenmediğine, bunun yerine sayfadaki birleşik metin ögesi ile denetlendiğine dikkat edin.

HorizontalRule

HorizontalRule ögesi DHTML sayfasına yatay bir çizgi ekler. Çizginin kalınlığını Size ve rengini Color özelliđi ile ayarlayabilirsiniz. Çizginin uzunluđunu Width özelliđi ile ayarlanır.

FileUpload

FileUpload özelliđi DHTML sayfasının kullanıcıların kendi yerel sabit disklerinden İnternet sunucusuna bir dosya yüklemek için kullanabilecekleri bir metin kutusu ve komut düğmesi ekler. Kullanıcı FileUpload ögesini tıkladıđında OnClick olayı ve metin kutusundaki metin seçildiđinde OnSelect olayı çalıştırılır.

HiddenFile

HiddenFile ögesi DHTML sayfasına kullanıcı tarafından görülemeyen bir metin kutusu ekler. Bu metin kutusu programınızdaki veriler için geçici bir saklama konumu olduğundan yararlıdır. Ayrıca HiddenFile ögesini Submit işlemi çalıştırıldıđında İnternet sunucusuna bilgi vermek içinde kullanılabilir. Value özelliđi gizli metnin içeriğini saklar ve tasarım yada çalışma zamanında ayarlanabilir.

InputImage

InputImage ögesi DHTML sayfasına resim ekler. Src özelliđi görüntü kutusunda yer alan resmin yol adını yada URL adresini belirler. Bu temel işlevselliğın yani sıra, InputImage ögesi kullanıcının yüklenen resmi girdiđi için bir mekanizma olarak kullanmasına izin verir.

List

List özelliđi bir DHTML sayfasına kayan bir liste kutusu eklemek için kullanılır. List ögesi Visual Basic'teki ListBox denetimine benzer. Size özelliđi karma kutuda bir seferde görülebilen öge sayısını belirler ve Length özelliđi ListBox'ın yüksekliğini belirler. Çalışma zamanında select özelliđi

degistirildiginde , OnChange olayi çalistirilir ve Value özelligi seçilen öğeleri içerir.

Ögeleri Yaratmak ve Özellestirmek

Bir Web sayfasinda fiziksel olarak DHTML araç kutusu öğeleri yaratmak bir Visual Basic formunda araç kutusuyla nesneler yaratmakla hemen hemen aynidir. Bir DHTML araç kutusu ögesini üç sekilde ekleyebiliriz.

- Araç kutusundaki öge simgesini tıklayip,Page Designer'in sag bölmesine sürükleyerek.
- Araç kutusundaki öge simgesini tıklayip,sag bölmede Visual Basic formunda yaptiginiz gibi çizerek.
- Araç kutusu ögesini çift tıklayarak.

Boyutlandirma isaretçisiyle araç kutusu öğelerini Visual Basic formunda bir denetimde yaptiginiz gibi yeniden boyutlandırabilirsiniz. Öğeleri sayfada tam olarak konumlandırabilir ya da diger öğelere ve gözatici penceresine göre göreceli olarak yerlestirebilirsiniz.

NOT: Tam konumlandirma,sayfaya yerlestirdiginiz ögenin tam olarak belirlediginiz konumda görünecegi anlamina gelir. Bagil konumlandirma ögenin sayfanin boyutu degistirildiginde komsu öğelere göre yer degistirilmesine izin verir.

Tasarim zamaninda properties penceresinde özellikleri ayarlayarak araç kutusu öğelerini özellestirebilirsiniz. Kullanilabilecek özellikler ögenin türüne göre degisir. Bazi araç kutusu öğeleri,ögeyi farenin sag düğmesiyle tıklayip,properties kutusunu tikladiktan sonra properties pages iletisim kutusunu doldurarak özellestirebileceginiz properties pages içerirler. Bir araç kutusu ögesini silmek için üç görünüm bölmesinde nesne adini tıklayin ve del'e basin.

Websansim Uygulamasina Öğeler Ekleme

Programi özellestirmeyi bitirdiginizde,çevir düğmesini tikladiginizda rastgele sayilar görüntüleyen bir hizmet programiniz olur. Sayfada bir ya da daha fazla yedi görünürse,multimedia MCI denetimi .Wav dosyasiyla üretilen alkis sesiyle kazandiginiz ilan edilir.

DHTML Uygulamanizdaki Dosyalari Yeniden Adlandirma

Bir DHTML uygulamasinin yeni bir sürümüyle çalışmak istiyorsanız, öncelikle proje (.vbp), tasarımcı (.dsr) ve modül (.mod) dosyalarını yeniden adlandırmalısınız.

1. Visual Basic'i baslatın ve C:\AaVb6\Ders22 klasöründeki Websansim projesini açın.
2. File menüsünde, Save Project As komutunu tıklayın.
3. C:\AaVb6\Ders22 klasörüne geçin, BenimDHTML7 yazın ve enter'a basın.
4. Project penceresindeki designers penceresini açın ve DHTMLPage1 tasarımını açın.
5. File menüsünde, Save Websansim.dsr As komutunu tıklayın.
6. Save As iletişim kutusunda, BenimDHTML7 yazın ve enter'a basın.
7. Project penceresindeki modules klasörünü açın ve ModDHTML modülünü tıklayın.
8. File menüsünde, Save WebLucky.bas As komutunu tıklayın.
9. Save As iletişim kutusunda BenimDHTML7 yazın ve enter'a basın.

NOT: Proje (.vbp), tasarımcı (.dsr) ve modül (.mod) dosyalarına ek olarak, bir DHTML uygulaması kullandığınız tasarımcı için HTML kodları içeren bir .dsx dosyası ve HTML kodları ve diğer ikilik sistem bilgileri ile bir .dca dosyası içerir.

Sayfadaki Bir Metin Ögesini Silme

Page Designer'da herhangi bir ögeyi ağaç görünüm yada tasarım bölmesinde seçip del tusuna basarak silebilirsiniz. Sayfadaki ilk boş satırı (Blank1) silmek için aşağıdaki adımları izleyin.

1. Ağaç görünüm bölmesinde Blank1 ögesini tıklayın.
2. Del tusuna basın.

Metin ögesi geçici olarak sayfadan silinir.

NOT: Bir ögeyi DHTML sayfasından silerken dikkatli olunmalıdır. Page Designer'in kararınızı degistirmek için bir undo özelliği yoktur.

Sayfaya Bir Image Ögesi Ekleme

BenimDHTML7 programında yapacağınız ilk geliştirme yarattığınız sayfaya kumar makinası oyunu için bir Image ögesi eklemek olacaktır. Image ögesi program çalıştığında bir bozuk para yığını görüntüler.

1. Araç kutusundaki Image ögesini çift tıklayın.

Page Designer HTML formunun merkezine bir Image ögesi ekler.

Araç kutusundaki ögeyi tıklayarak ve sayfaya fare ile sürükleyerek de bir Image ögesi yaratabilirsiniz ama araç kutusu ögesini çift tıklayıp yeniden boyutlandırmak genellikle en kolay yoldur.

2. Image ögesi Sansim7 oyunu başlığı ve üç sıfırın arasına gelinceye kadar aşağı ve yukarı ok tuşlarına basın.

NOT: Image ögesini fareyle de yeniden boyutlandırabilirsiniz ama büyük olasılıkla Width ve Height özelliklerini ayarlamak daha kolay gelecektir.

Varsayılan durumunda, Image ögesi sayfada bir görüntüyü tam boyutunda gösterir. Ama Image ögesinin Width ve Height özelliklerini Properties penceresinde ayarlayarak, bir grafiği belirli bir boyutta görüntülemesini sağlayabilirsiniz.

3. Değişikliklerinizi kaydetmek için Visual Basic araç çubuğundaki, Save Project düğmesini tıklayın.

Sayfaya Bir Button Ögesi Ekleme

Şimdi, bir kumar makinasını başlatmak ve üç rasgele sayı görüntülemek için DHTML sayfasına bir Button ögesini ekleyeceksiniz.

1. Araç kutusundaki Button ögesini çift tıklayın.

Page designer sayfaniza bir Button ögesi ekler ağaç bölmede yeni bir girdi olusturur.

2. Dügmeyi Kazanma etiketinin karsisina, sayfanin sag tarafina tasimak için sag ve asagi ok tuslarini kullanin.
3. Properties penceresini tekrar görüntüleyin, value özelligini bulun ve value özellik kutusundaki metni silin.
4. Value kutusunda Çevir yazin ve enter'a basin. Button ögesindeki metin Çevir'e degisir. Böylelikle Button ögesi simdi program kodu için hazirdir.

DHTML Sayfasina ActiveX Denetimleri Ekleme

Bir DHTML uygulamasinda asil Visual Basic araç kutusu denetimlerini kullanamamaniza ragmen, bir DHTML sayfasina ActiveX denetimleri ekleyebilirsiniz. Page Designer uygulamanizin HTML kodunda etrafina <OBJECT> imlerini yerlestirerek bu denetimleri yönetir. Çogu durumda, özgün ActiveX denetimi tarafından desteklenen geleneksel özellikler, yöntemler ve olaylari birkaç ilginç sinirlama ile kullanabilirsiniz. ActiveX denetimleri tarafından saglanana standart özellikler, yöntemler ve olaylar yiginina ek olarak, denetimlerimiz için HTML özelliklerinden birkaçina daha erisebilirsiniz: ClassID, CodeBase, CodeType, ID vb. bu fazladan özellikler, ActiveX denetimleri tarafından saglanan temel islevselligi kullanirken, HTML sayfa öğelerinin gelismis davranislarindan da yaralanmanizi saglar.

DHTML uygulamalarına ActiveX denetimleri ekleme yetenegi Visual Basic Programcilarina büyük yarar saglar.

Araç Kutusuna Bir ActiveX Denetimi Ekleme

DHTML sayfanizda bir ActiveX denetimini kullanmadan önce, denetimi Visual Basic araç kutusuna eklemelisiniz. ActiveX denetimleri araç kutusunda, General düğmesini tıklayarak erisebileceginiz General bölmesine saklanir.

1. Project menüsünde Components komutunu tiklayin.

Listede asagi kayin ve Microsoft Multimedia Control 6.0 girdisinin yanindaki onay kutusunu tiklayin.

2. Bu ActiveX denetimini araç kutusuna eklemek için OK'yi tıklayın

Multimedia MCI denetimini sonraki araç kutusundaki General bölümünde görülür.

Sayfada Bir Multimedia MCI Denetimi Yaratma

ActiveX denetimleri sayfada diğer denetimler gibi çizilir. Bu denetimlerin Visible özelliği olmadığı için, program çalıştığında görünmelerini istemiyorsanız, sol kenar boşluğunun dışına sürükleyebilirsiniz.

Uygulamanıza Multimedia MCI denetimi eklemek için aşağıdaki adımları izleyin:

1. Araç kutusundaki Multimedia MCI denetimini tıklayın.

Page designer sayfasının ortasında yeni bir denetim yaratır. Denetim için araç bölme de girdi ekler.

2. Multimedia MCI denetimi bölmenin sol kenarından dışarı tasıncaya kadar sol OK tusuna basılır. Denetimi görünümün dışına tasımak onu görünmez yapar.
3. Değişikleri kaydetmek için, Visual Basic araç çubuğundan Save Project düğmesini tıklayın.

DHTML Öğeleri İçin Olay Yordamları Yaratmak

DHTML sayfanızdaki öğeleri ve ActiveX denetimlerini programlı olarak, Visual Basic uygulamalarında olduğu gibi olay yordamları ve iyi seçilmiş program komutlarıyla denetleyebilirsiniz. DHTML olay yordamları yazarken karşılaşacağınız en büyük farklılık olay ve nesne adlarının çoğunu değişmiş olması, böylece öğrenmek için biraz alıştırma yapmayı gerektirmesidir. Ama If...Then...Else,For...Next ve Do...Loop gibi tanıdık yapılar aynıdır.

DHTMLPage_Load Olay Yordamını Yaratma

Bu olay yordamı genellikle Visual Basic'teki Form_Load olay yordamına, DHTMLPage_Load olay yordamının sayfa her yenilendiğinde tekrar çalıştırılması dışında denktir.

1. Code penceresini görüntülemek için project penceresindeki View Code düğmesini tıklayın.
2. Code penceresindeki object açılır liste kutusunu tıklayın ve DHTMLPage nesnesini tıklayın.

DHTMLPage_Load olay yordamı Code penceresinde görünür.

3. Aşağıdaki program kodunu yazın:

```
'Baslik için altı çizgili stili ayarla
LuckyHead.Style.textDecorationUnderline=True
'sayıların rengini maviye ayarla
Num.Style.Color="blue"
'rastgele sayı olusturucusunu baslat
Randomize
'bozuk para yiginini görüntüle
Image1.src="C:\proje\..\paralar.wmf"
'Multimedia MCI denetimini düzenle ve
çalistir
MMControl1.Notify=False
MMControl1.Wait=True
MMControl1.Shareable=False
MMControl1.DeviceType="WaveAudio "
MMControl1.FileName="C:\AaVb6\Ders22\alkis.wa
v"
MMControl1.Command="Open"
'Property Bag'deki daha önceki kazanmaları
belirlemek
'için GetProperty islevini kullan.(HTML
sayfasi
'yükleme ve geri yükleme işlemleri sırasında
değerini
'koruyan bir saklama yeri)."Sansim 7
Hakkında"
'Köprüsüne ya da diğer Web sayfalarına
geçişler
'sirasındaki kazanma sayısını bu kodla
kaydedebilirsiniz.
```

```
Result.InnerText="Kazanma:" &  
GetProperty("Kazanma")
```

4. Visual Basic tarafından belirlenen yazım ya da söz dizimi hatalarını onarın.
5. Eklemelerinizi kaydetmek için Visual Basic araç çubuğundaki Save Project düğmesini tıklayın.

DHTMLPage_Load Kodunu İnceleme

DHTML sayfası gözeticiye yüklendiğinde çalışan olay yordamı bazı önemli görevleri gerçekleştirir. İlk olarak kod LuckyHead ögesini sayfada görsel olarak ayarlamak için altı çizili stiliyle biçimlendirir. Benzer bir biçimde sayfadaki sayıların rengini de NumID,Style ve Color özelliği kullanılarak maviye ayarlanmıştır.

DHTMLPage_Load olay yordamındaki Randomize bildirisi rastgele sayı üreticisini sistem saatinden rastgele bir sayıyla başlatır. Sonraki bildiri paralar.wmf meta dosyasını formdaki Image ögesine ögenin Src özelliğini sabit diskteki grafiğe ayarlayarak yükler.

Bu olay yordamındaki üçüncü grup program bildirileri Multimedia MCI denetimini kullanım için yapılandırır ve açar. Burada önemli olan DeviceType özelliğinin WaveAudio'a ve FileName özelliğinin Alkis.Wav'a ayarlanarak bu belirli ses dosyasına erişebilmesidir. Alkis sesi kullanıcı oyunu kazanana kadar çalınmaması sonraki kısımda yazacağınız Button1_OnClick olay yordamıyla belirlenir.

Son program bildirisi kümesi DHTML uygulamasını property Bag'inde kullanıcının oyun sırasında daha önce kazanıp kazanmadığını belirlemek için Kazanma adında genel bir değere başvurur. Bir property bag geçerli DHTML sayfasının dışında yer alan geçici bir saklama kabidir. Bu değeri sonra geri almak için,GetProperty işlevine başvurulur ve istediğiniz değerin adını belirlersiniz. GetProperty işlevinin sayfa ilk defa yüklendiğinde bir değer üretmediğine dikkat edin ama oyun ilerledikçe, bu işlev kullanıcı Sansin 7 Hakkında ya da diğer Web sayfalarına geçtiğinde sayıyı saklar.

Button1_OnClick Olay Yordamını Yaratma

1. Code penceresinde Object açılır liste kutusunu açın ve Button1 nesnesini tıklayın.

Button1_OnClick olay yordamı Code penceresinde görünür.

2. Aşağıdaki program kodunu yazın:

```
'Kazananlar için yerel değişken x'i tanımla
(çevirmeler
'arasında Property Bag'e kopyalanır.)
Dim x
'üç rastgele sayı al
Num1.innerText=Int (Rnd*10)
Num2.innerText=Int (Rnd*10)
Num3.innerText=Int (Rnd*10)
'Herhangi bir sayı 7 ise yigini göster ve
biple
If Num1.innerText=7 Or Num2.innerText=7 Or _
    Num3.innerText=7 Then
    'Bir kazanan varsa, .wav dosyasını çal
    (alkis.wav)
    MMControl1.Command= "Prev" 'gerekliyse geri
    al
    MMControl1.Command= "Play" 'bir .wav dosyası
    çal
    've Property Bag'deki kazanma sayısını
    arttır.
    X=GetProperty ("Kazanma")
    Result.innerText="kazanma:" ve x+1
    PutProperty "Kazanma",x+1
Endif
```

3. Visual Basic tarafından belirlenen söz dizimi hatalarını düzeltir.
4. Olay yordamın kaydetmek için araç çubuğundaki Project düğmesini tıklayın.

Button1_OnClick Kodunu İncelemek

Button1_OnClick olay yordamı kullanıcı DHTML sayfasındaki Çevir düğmesini tıkladığında çalışır. Öncelikle olay yordamı Property Bag'de saklanan kazanma sayısını saklamak için bir x yerel değişkeni tanımlanır.

0 ve 9 arasındaki rasgele sayılar Rnd işlevi tarafından yaratılır, Int işlevi tarafından yuvarlanır ve innerText özelliğinin sırasıyla Num1, Num2 ve Num3 öğeleri tarafından sayfaya kopyalanır. InnerText özelliği sayfadaki var olan metin için belirlediğiniz metin bağımsız değişkenin yerini alarak metin öğesinin yerini değiştirir. InnerText özelliği bir öğenin değerini de temsil edebilir. Bu işlevsellik sayfada yer alan rasgele sayılardan birinin 7 olup olmadığını belirlemek için if bildirisinde kullanılmıştır. Yedi görünürse, Multimedia MCI denetimindeki Prev ve Play bağımsız değişkenleri kullanılarak zaferi ilan eden .wav dosyası çalınır. Alkisi baslandıktan sonra, GetProperty işlevi ile Kazanma değeri yerel x değişkenine kopyalanır. Sonra formdaki Kazanma sayısı Result öğesinde artırılır. Son olarak, artırılan kazanma sayısı PutProperty işlevindeki Property Bag'a kopyalanır.

BenimDHTML7 Uygulamasını Çalıştırma

1. Internet Explorer'i başlatmak ve DHTML sayfasını yüklemek için Visual Basic araç çubuğundaki start düğmesi tıklanır.
2. Sayfada bir yada daha fazla yedi görüne kadar Çevir düğmesini tıklayın.

Oyunu kazandığımızda, alkis.wav dosyası çalıştırılır.

3. Kazanma toplamınızı beş yada altı yapmak için birkaç kez daha tıklayın. Her yedi görüldüğünde toplamınıza başka bir kazanma eklenir.
4. Ekranı yenilemek ve DHTMLPage1_Load olay yordamını çalıştırmak için Internet Explorer araç çubuğundaki Refresh düğmesini tıklayın.
5. Son olarak programdaki köprüyü tıklayın ve gözeticinin Back düğmesini tıklayarak sayfa yenilendiğinde doğru Kazanma sayısını listelendiğini doğrulayın.
6. BenimDHTML7 uygulamasıyla alıştırmalar yapmayı bitirdiğinizde , Internet Explorer'ın başlık çubuğundaki Close

düğmesini tıklayın. Ve Visual Basic araç çubugundaki End düğmesini tıklayın.

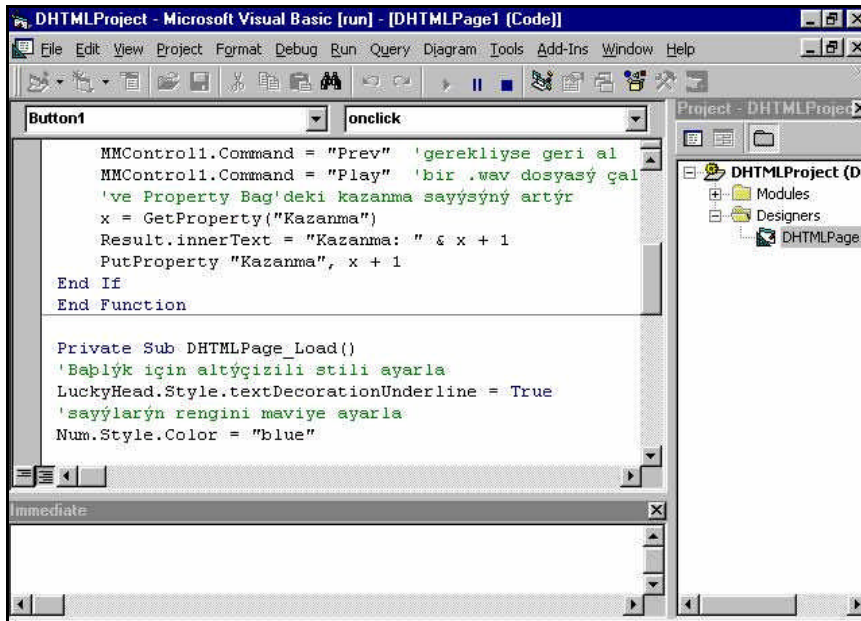
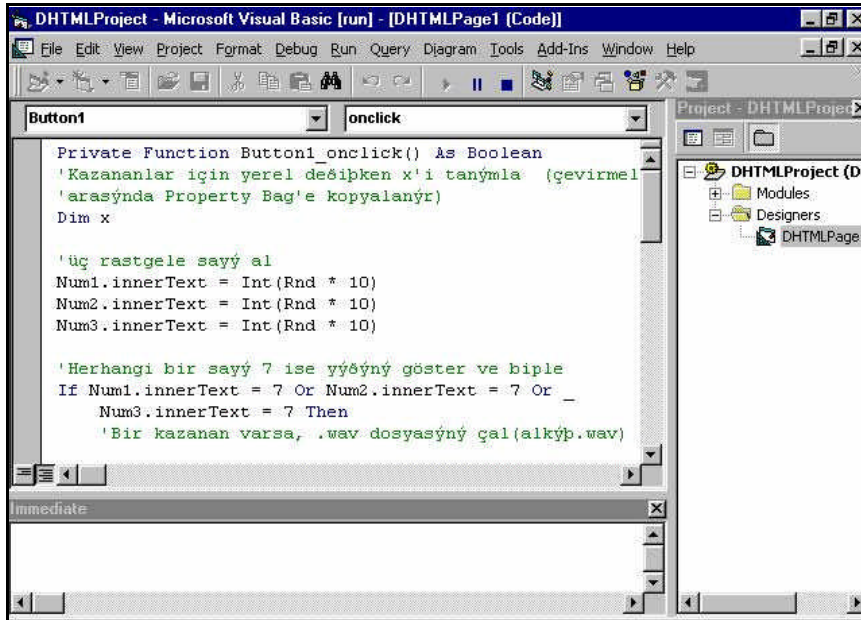
ÖZET

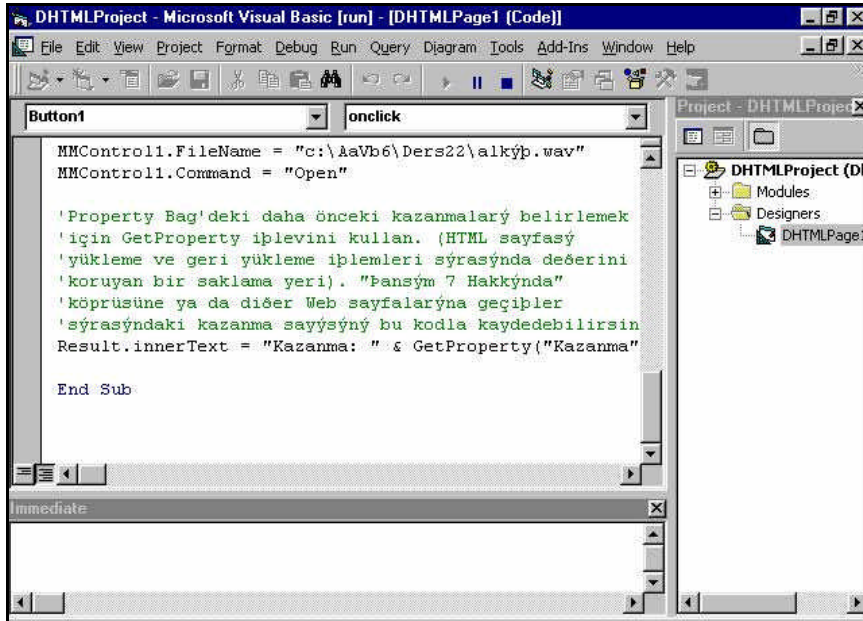
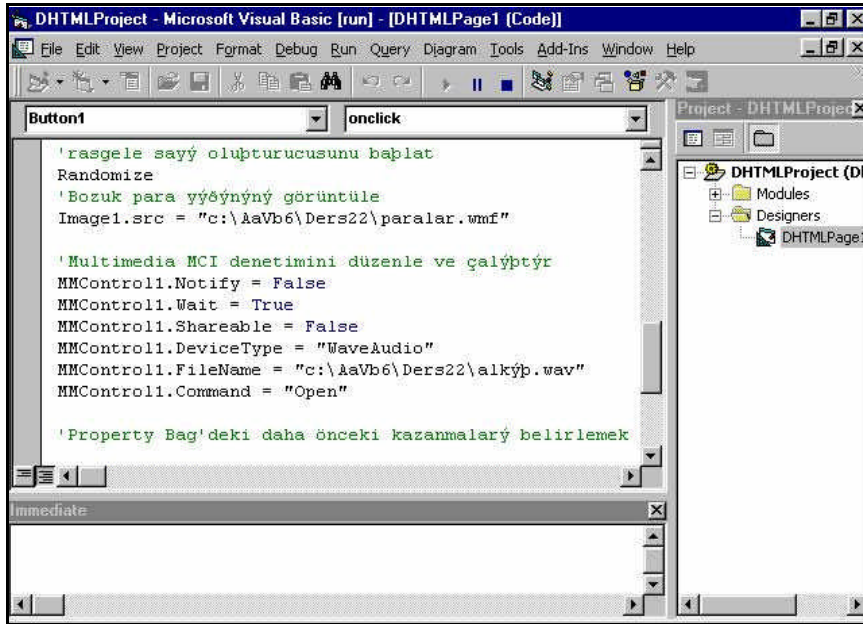
Visual Basic 6.0 ile yeni bir tür uygulama türü geliştirilmiştir. Bu uygulama: DHTML Application'dir. Bir DHTML Application, DHTML Designer'i kullanarak HTML elemanlarının özellikleriyle bir Web sayfası yaratmayı sağlar. DHTML Page Designer ise Visual Basic kodunu bir ActiveX DLL olarak paketleyerek tarayicinizda çalışmasını sağlar. DHTML uygulamalar çok sayıda üstünlüğe sahiptir:

- Azaltılmış server yükü: DHTML uygulamaların server yükü daha azdır. Çünkü bu uygulamaları client tarafında çalıştırılır.
- Daha hızlı yanıt: Bir sayfanın güncellenmesi gerektiğinde tarayıcının (browser server ile ilişki kurması gerekir.
- Gelişmiş durum (state) yönetimi: HTML sayfalarının durumu takip edilmez. Diğer bir deyişle tarayıcı sayfayı değiştirdiğinde ya da kapattığında bütün değişkenler kaybolur.
- Offline yeteneği: Bir DHTML uygulaması aracılığıyla kullanıcılar yerel bir Intranet'i kullanabilirler.
- Kod Güvenliği: Bir HTML sayfasının kodları kolayca görülebilir ve kopyalanabilir. Oysa DHTML uygulamalarını derlenerek program haline getirildiği için içeriği görülemez.

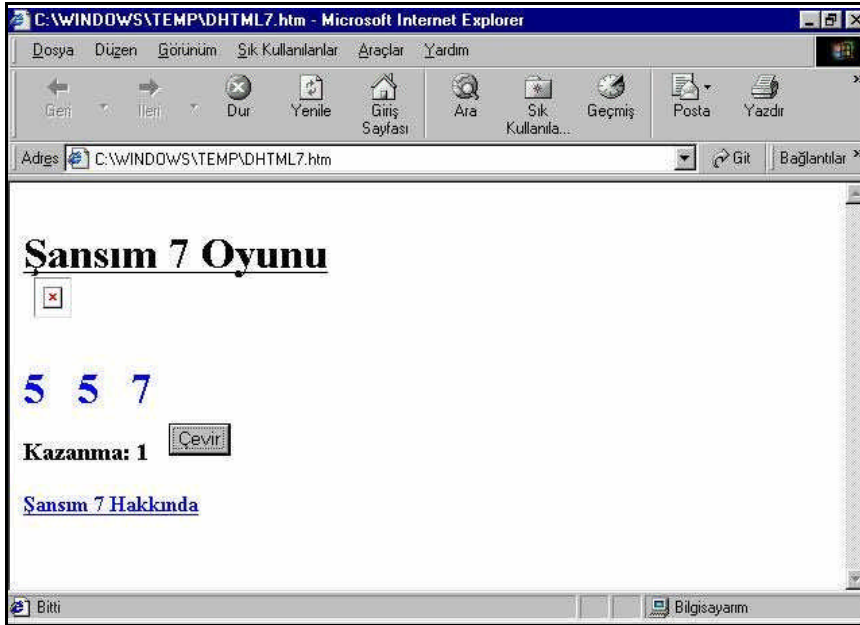
Örnek:

Microsoft Visual Basic 6.0





Microsoft Visual Basic 6.0



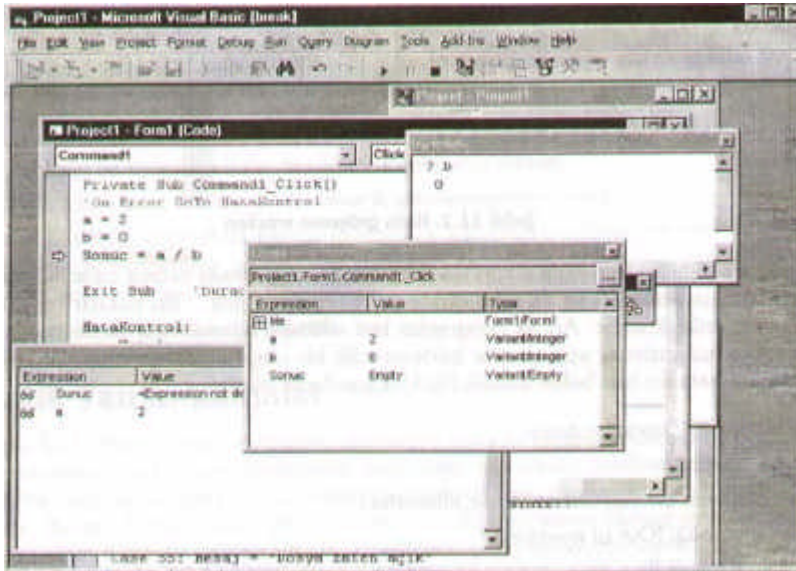
15-VISUAL BASIC ERRORLER

Amaçlar:

- Hata denetimini tanımlamak.
- Hata denetim araçlarını tanımlamak.
- Hata denetim deyimlerini açıklamak.
- Hata denetimin kullanıldığı alanları açıklamak.
- Hata kodlarını açıklamak.
- Hata oluştuğunda yapılacak işlemleri açıklamak.

Hata Denetimi

Hata denetimi (troubleshooting) programda oluşan çalışma zamanı ve diğer kodlama hatalarının tanımlanması (teshis) ve giderilmesi için yapılan çalışmalardır. Visual Basic, programların yazılması derlenmesi ve çalıştırılması sırasında karşılaşılan hatalara karşı geliştirilmiş bir hata bulma ve düzeltme (debugging) olanına sahiptir.

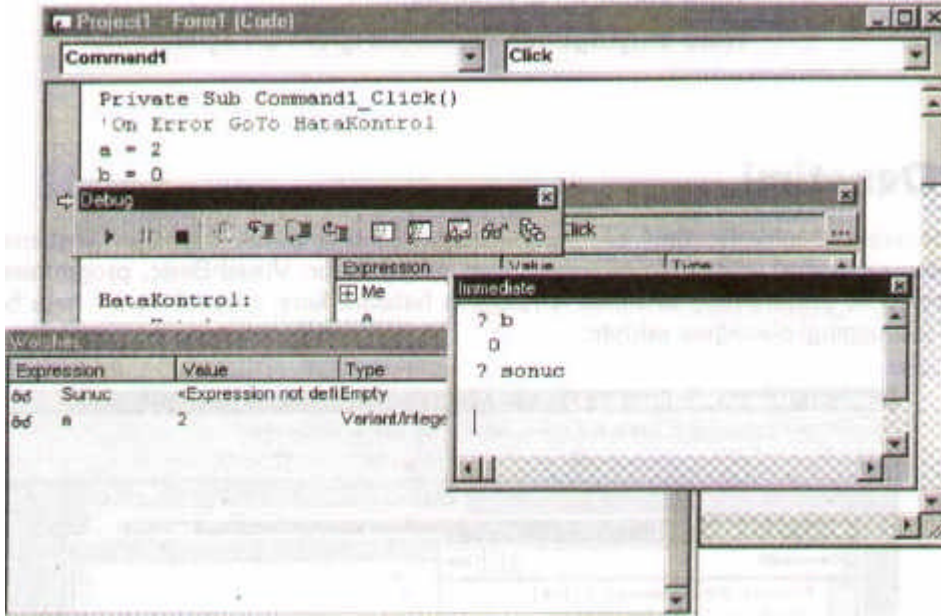


Sekil-1 Hata Denetimi

Visual Basic'te karsilasilacak hata türleri sunlardir:

- Derleme hatalari (compile errors)
- Çalışma zamani hatalari {run-time errors)
- Mantik hatalari (logical errors)

Derleme (compile) hatalari, programi yazarken yapılan hatalardir, örneğin **For** olmadan kullanılan **Next** deyimi, bir tirnak ya da dogru yere konmamış bir virgül gibi. Program yaziminda dilin bileşenlerinin uygun olarak kullanılmaması, diğer bir deyişle sözdizim (syntax) hatalari da derleme sırasında ortaya çıkar. Visual Basic bu tür hatalarin aninda ve çalışmadan önce ve çözülmesine ilişkin çok sayıda araca sahiptir.



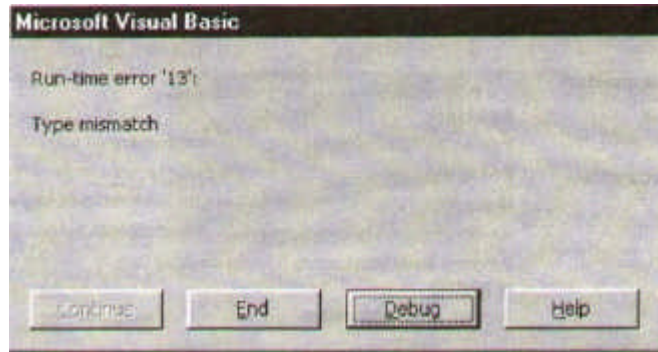
Sekil-2 Hata Giderme Araçları

Çalışma zamani hatalari (run-time error) ise program çalıştığı sırada ortaya çıkar. Olmayan bir dosyayı açmaya kalkmak ya da sabit diskte yer kalmaması gibi... Bu hatalar programın geliştirme sürecinde anlaşılmazlar. Ancak programın test edilmesi aşamasında ya da programcılar tarafından

kullanilmasi asamasinda beklenmedik bir olaydan dolayi olusurlar. Bunun disinda çalisma zamani hatalari bazi beklenmedik (tipik) durumlarda olusur:

- Sifira bölünme (Divide by zero)
- Tasma (Overflow)
- Uygun olmayan bir veri atama (Type Mismatch)
- Bellek yetersizligi (Out of memory)
- Disk dolu (Disk Full)

Çalisma zamani hatalarini önlemek için programda hatalara karsi önlem alinir. Bu nedenle bu tür hatalara tuzaklanabilir (trappable errors) denir.



Sekil-3 Bir çalisma zamani hatasi

Ipucu : İyi bir programcinin Visual Basic hata giderme olanaklarini çok iyi biçimde bilmesi gerekir. Programi iyice tespit edip, hata vermesi durumunda hata bulma ve giderme araçlari kullanarak sonuca gidebilir.

Daha Az Hatayla Karsilasmak

Yazilan programda daha az hata olusmasi için; program tasariminin en iyi sekilde yapılmasi gerekir. Bu konuda geniş bilgi için birinci bölüme bakiniz. Bunun disinda program hatalarinin azalmasi için asagidaki konulara dikkat edilmesi gerekir:

- Degiskenlerin ve tiplerinin daha önceden tasarlanmis olmasi
- Sub ve Function'larin her birinin iyi tanımlı isler için geliştirilmesi
- **Option Explicit** deyiminde modül tanımlama kisminde mutlaka yer verilerek; degiskenlerin uya zorlanmasi ve böylece degisken tanımlamasindan ve yanlış kullanimlarindan Kök hatalarin en aza indirilmesi.

- Programi çok iyi şekilde test etmek.

Otomatik Tamamlamalar

Visual Basic kod ortamı içinde; kod yazarken birçok konuda yardımcı olur. Bu düzenlemeler; yazılan kodların otomatik olarak yazım denetiminin yapılması, nesnelerin özelliklerinin otomatik olarak listelenmesi yazılması durumunda parametrelerinin otomatik olarak ekrana gelmesi gibi bunun dışında Visual Basic, program kesildikten sonra (debug modda) fare ile üzerine gidilen değişkenin değerini de gösterir. Bu özelliklerin çoğu varsayım olarak aktiftir. Eğer aktif değilse Tools menüsünden Options menüsünden düzenlemelerle sağlanır.

NOT

Birçok programda olduğu gibi Visual Basic 6.0'da da programın ortam düzenlemelerinin yapılması Tools menüsünden Options komutu ile yapılır. Options iletişim kutusu çok sayıda sekmeye sahiptir. Bu sekmelere tıklanarak istenilen alanlarda değişiklikler yapılabilir.

Editör Sekmesi seçenekleri

Kod düzenlemeleri

Auto Syntax Check : Yazılan kodların otomatik olarak yazım denetiminin yapılmasını sağlar.

Require Variable Declaration : Kod içinde kullanılan bütün değişkenlerin tanımlanmasını zorunlu kılar.

Auto List Members : Nesnelerin özelliklerinin otomatik olarak listelenmesini sağlar.

Auto Quick Info : Bir deyimın yazılması durumunda parametrelerinin ekrana gelmesini sağlar.

Auto Data Tips : Program kesildikten sonra (debug modda) fare ile üzerine gidilen değişkenin değerini gösterir.

Auto Indent : İçerden başlamayı sağlar.

Pencere Düzenlemeleri

Drag-and-Drop Text Editing : Kod satırları üzerinde fare ile seçip sürüklemeye işleminin yapılmasını sağlar.

Default to Full Module View : Bir procedure ya da bütün procedure'ların görülmesini sağlar.

Procedure Separator: Procedure'lar arası ayraç.

Bunun dışında hata bulunduğu hatalı kodun rengini belirlemek için **Options** menüsünün **Editör Format** sekmesindeki düzenlemeler kullanılır.

Kod düzenlemeleri

Normal Text: Normal kod satırlarının rengini düzenler.

Selection Text: Seçilen kod satırlarının rengini düzenler.

Syntax Error Text: Sözdizimi hatası olan kod satırlarının rengini düzenler.

Foreground : Yazı rengi.

Background : Zemin rengi

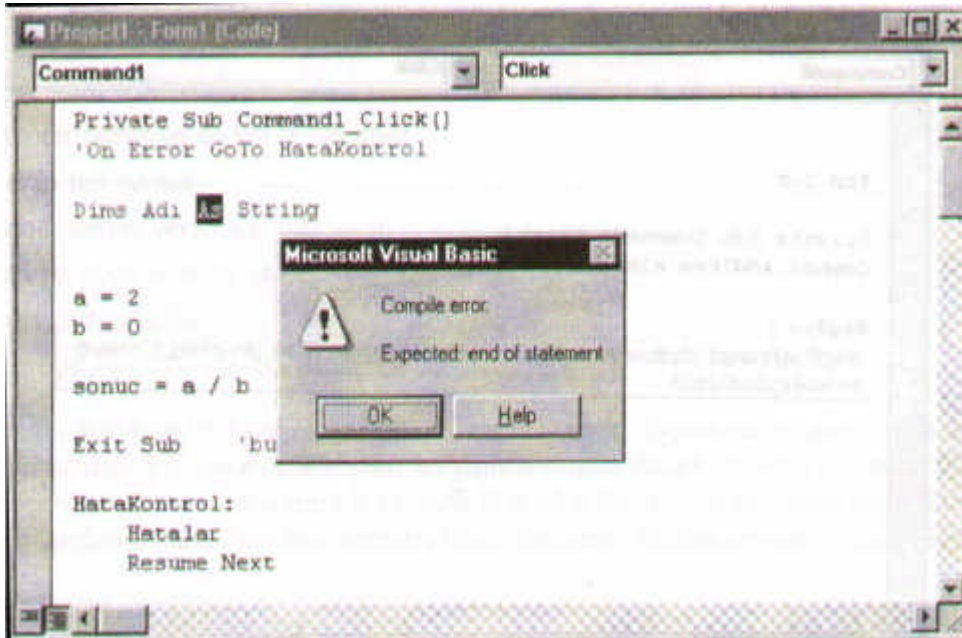
Indicator: Göstergenin rengi

Auto Syntax Check ile yazılan kodların yazım denetimi otomatik olarak yapılır. (Şekil-4).

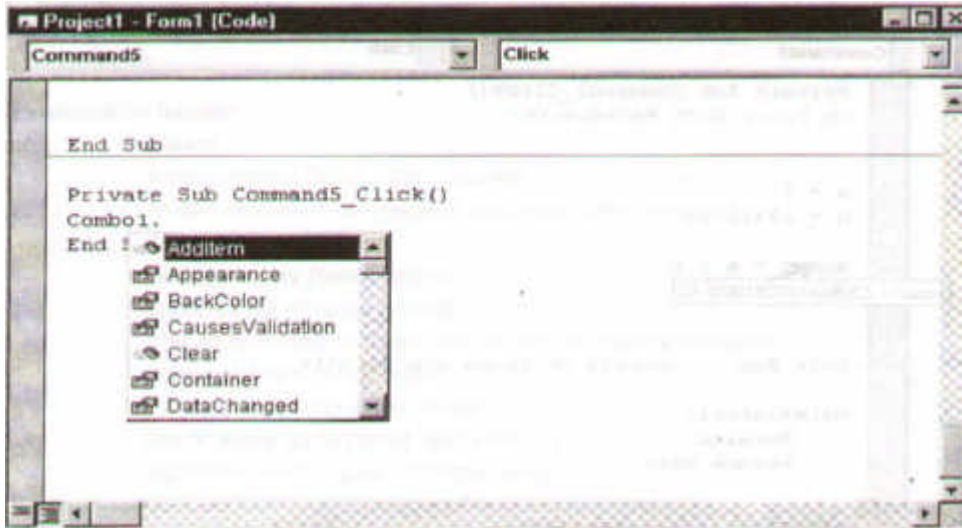
Auto List Members özelliği nesnelerin özelliklerinin otomatik olarak listelenmesini sağlar { Şekil-5}.

Auto Quick Info özelliği ile bir deyim yazılması durumunda parametrelerinin ekrana gelmesi sağlanır { Şekil-6}.

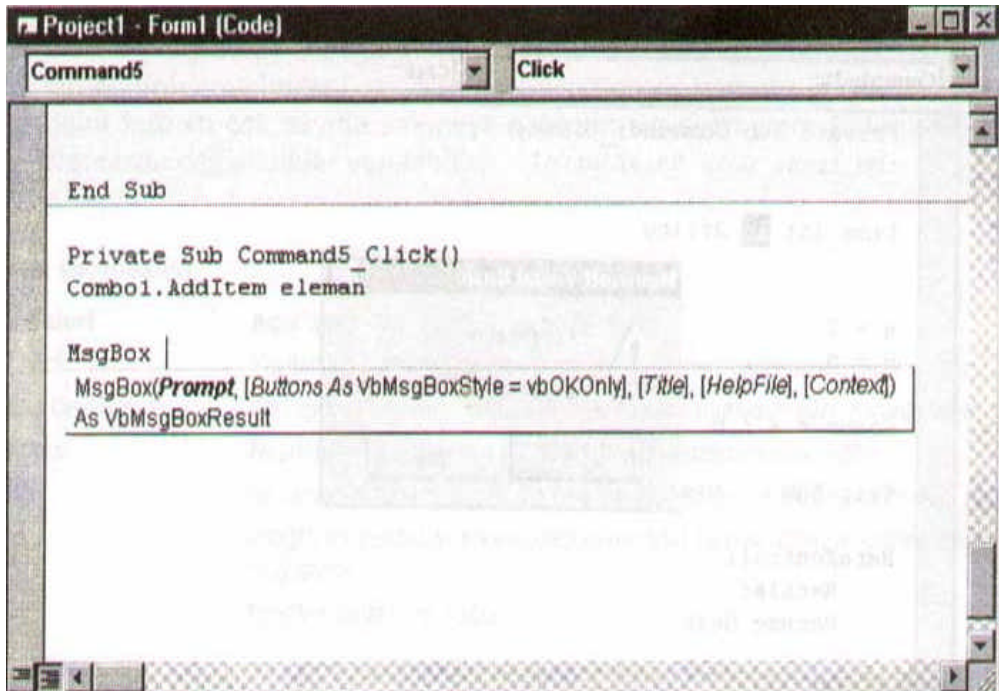
Auto Data Tips özelliği de çok yararlıdır. Bu özellik Program kesildikten sonra (debug modda) fare ile üzerine gidilen değişkenin değerini gösterir(Şekil-7).



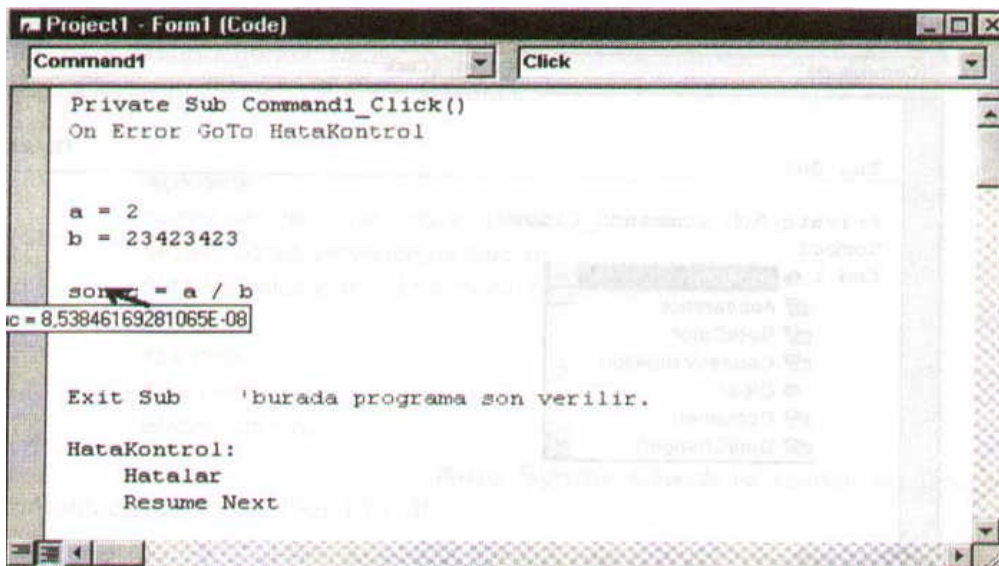
Sekil-4 Otomatik Sözdizimi Kontrolü



Sekil-5 Nesnelerin Özelliklerinin Listelenmesi



Sekil-6 Deyimlerin Ipuçlari



Sekil-7 Kesilme Durumunda Degiskenlerin Degeri

Çalışma Zamani Hataları

Çalışma zamani hataları (run-time errors) programın çalışması sırasında oluşan hatalardır. Program hazırlanmış ve kullanıcıya sunulmuştur. Ya da test aşamasındadır; çalışma sırasında yapılan işlemlerle birlikte bir takım (beklenmeyen) hatalar oluşur. Örneğin bir tamsayı tanımlı degiskene bir milyon değerinin atanması ya da sabit diskin dolması gibi.

Bu Hatalar:

- Bir değer sifıra bölünebilir.
- Belirtilen bir dosya ya da yol bulunamaz.
- Degiskenin tipi uymaz.
- Degiskenin tanımı veriyi alamaz, tasar.
- Belirtilen bir aygıt (sürücü, yazıcı, vb.) bulunamaz.
- Bellek yetersiz kalabilir.
- Diğer bir uygulama ile yapılan iletişimde sorun olur.

Çalışma zamani (run-time) hatalarına karşı herhangi bir önlem alınmazsa program hata verir ve kesilir. Diğer bir açıdan; bu durum kullanıcılar karşısında düşülebilecek en kötü durumlardan birisidir. Mu nedenle çalışma zamani hatalarına karşı belli önlemler daha önceden alınmalıdır. Bu hataların tanımlanıp önlemlerini daha önceden alınması işlemi hataların tuzaklanmasını ortaya çıkarır.

Tuzaklanabilir Hatalar

Tuzaklanabilir hatalar (trappable errors) program çalışırken oluşur (bazı hatalar tasarım zamanında da oluşur). Bu hatalar 1-1000 arasında hata kodu döndürürler. Bu hata kodları **On Error** deyimi ve **Err** nesnesi ile birlikte ele alınarak hataların değerlendirilmesi ve programın uygun biçimde yönlendirilmesi sağlanır.

Tuzaklanabilir hatalar

Kod Mesaj

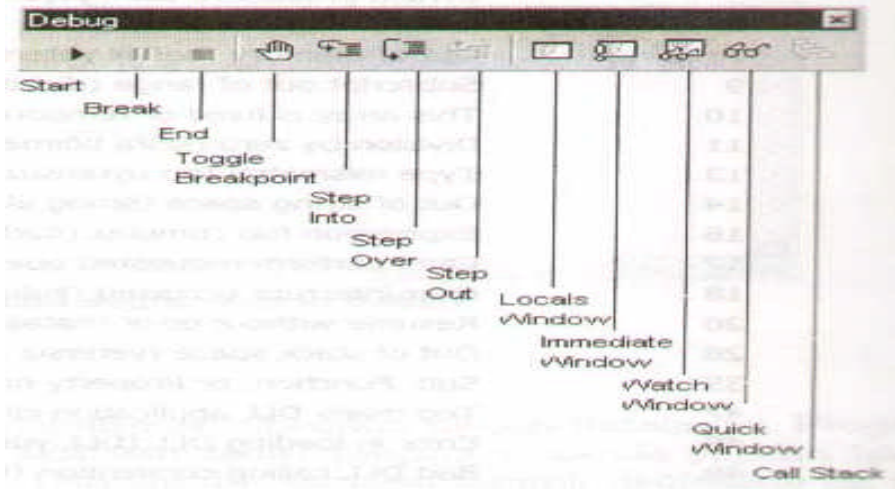
3	Return without GoSub (GoSub'siz Return)
5	Invalid procedure call (geçersiz procedure çağırması)
6	Overflow (taşma)
7	Out of memory (bellek yetersiz)
9	Subscript out of range (dizi dışı)
10	This array is fixed or temporarily locked (dizi sabit ya da kilitli)
11	Division by zero (sifıra bölme)

13	Type mismatch (tip uyumsuzluğu)
14	Out of string space (string alanı yetersiz)
16	Expression too complex (ifade çok karışık)
17	Can't perform requested operation (işlem işletilemiyor)
18	User interrupt occurred (kullanıcı kesmesi oluştu)
20	Resume without error (hatasız devam)
28	Out of stack space (yetersiz stack alanı)
35	Sub, Function, or Property not defined (... tanımlanmamış)
47	Too many DLL application clients (çok sayıda DLL uygulama)
48	Error in loading DLL (DLL yüklemede hata)
49	Bad DLL calling convention (DLL hatası)
51	Internal error (iç hata)
52	Bad file name or number (yanlış dosya)
53	File not found (dosya bulunamadı)
54	Bad file mode (kötü dosya biçimi)
55	File already open (dosya zaten açık)
57	Device I/O error (giriş/çıkış aygıt hatası) T1.
58	File already exists (dosya zaten var)
59	Bad record length (kötü kayıt uzunluğu)
61	Disk full (disk dolu)
62	Input past end of file (dosya sonunu geçme)
63	Bad record number (kötü kayıt numarası)
67	Too many files (çok fazla dosya)
63	Device unavailable (aygıt kullanılamaz durumda)
70	Permission denied (izin yok)
71	Disk not ready (disk hazır değil)
74	Can't rename with different drive (başka bir sürücü adıyla değiştirilemez)
75	Path/File access error (dosya adı hatası)
76	Path not found (yol bulunamadı)
91	Object variable or With block variable not set (nesne değişkeni düzenlenmemiş)
92	For loop not initialized (döngü başlatılmamış)
93	Invalid pattern string (yanlış string düzeni)
94	Invalid use of Null (Null yanlış kullanılmış)

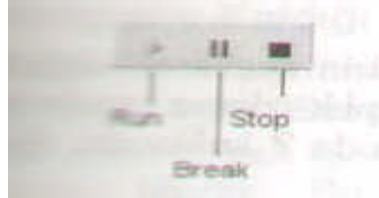
Hata mesajları için Help'ten Index bölümünden "Trappable Errors" a bakınız. Bu hatalar **On Error** deyimi ile karşılanarak kontrol altına alınır. Bakınız: "On Error deyimi".

Hata Düzeltme Araçları

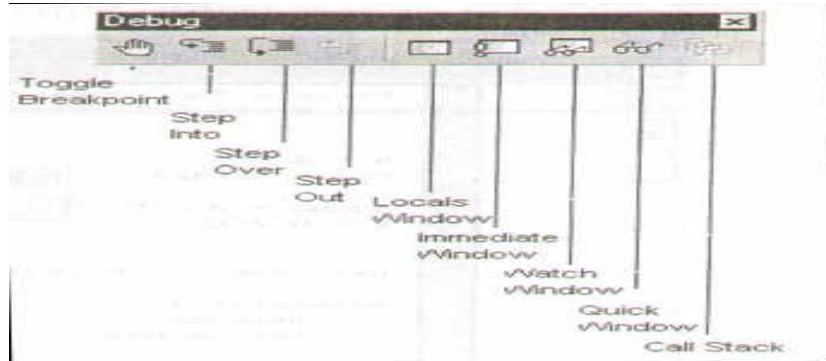
Bir Visual Basic programı geliştirirken ya da programın çalışması sırasında bir hata ile kesilmesinden sonra hatanın bulunması ve giderilmesi için belli araçlar kullanılır. Bunların başında **Debug** (Hata Giderme) araç çubuğu gelir.



Debug araç çubuğunun kullanılması için programın kesilmesi gerekir. **Kesme (break)** işlemi hata sonucunda ya da manuel olarak da yapılabilir.



Programın kesilmesi ya da **Break** düğmesine tiklanarak program debug moda alınır. İşte bu durumda programcı Visual Basic hata giderme araçlarını kullanır. Hata giderme araçlarının başında değişkenleri ve programdaki belli yerleri izleyebileceğimiz pencereler gelir.

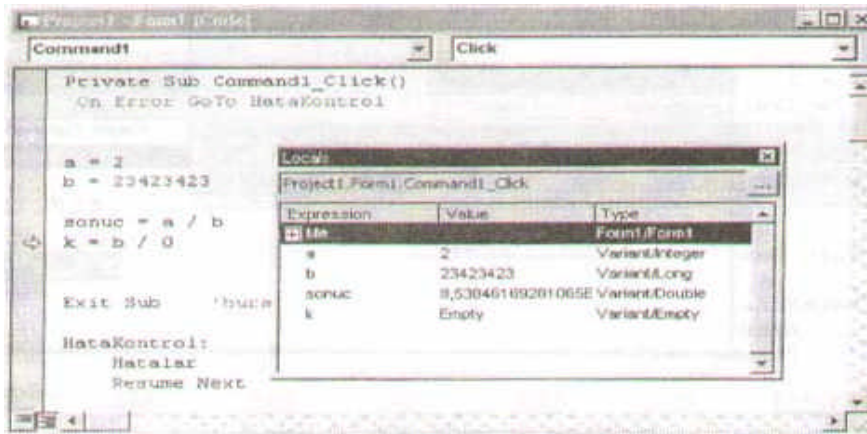


Locals Window, Immediate Window ve Watch Window

Program kesildikten sonra; Procedure içinde bir degiskenin ya da bir ifadenin seçilerek degerinin ne olduguna bakilmak istenirse o zaman anlik İzleme penceresi (instant watch) kullanilir.

Locals Window

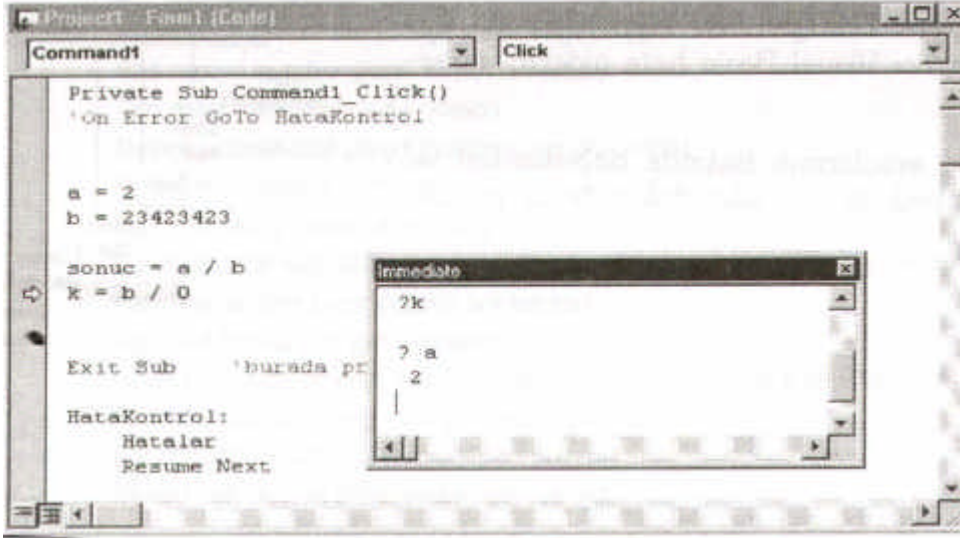
Program kesildiginde bulunan procedure içinde tanımlanan bütün degiskenlerin degerini verir. **Expression** sütununda degiskenlerin adları listelenir. Bütün class modüllerinde buradaki ilk deger <Me> degeridir. Bu degerin üzerine tiklanarak genisletilir. **Value** sütununda ise degiskenin degeri görünür. Bu degerin üzerinde çift tiklanarak degisiklik yapılabilir.



Sekil-8 Locals Penceresi

Immediate Window

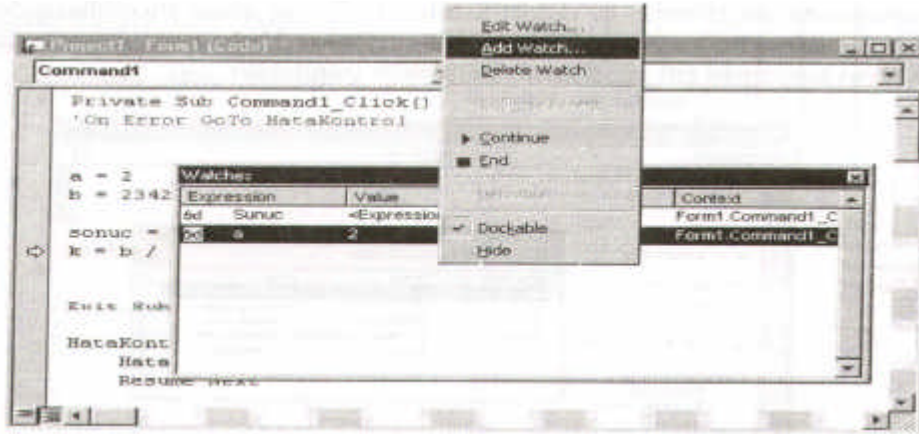
Program kesildikten sonra; procedure içinde bir degiskenin ya da bir ifadenin seçilerek degerinin ne olduguna bakilmek istenirse o zaman anlik izleme penceresi kullanilir. Anlik izleme penceresinde degiskenlerin ya da ifadelerin degerlerini göstermek için (Print) deyimi ya da ? kullanilir. Ayrica istenirse kod satirindan Immediate Window ekranina sürükleme yapilir.



Sekil-9 Anlik izleme

Watch Window

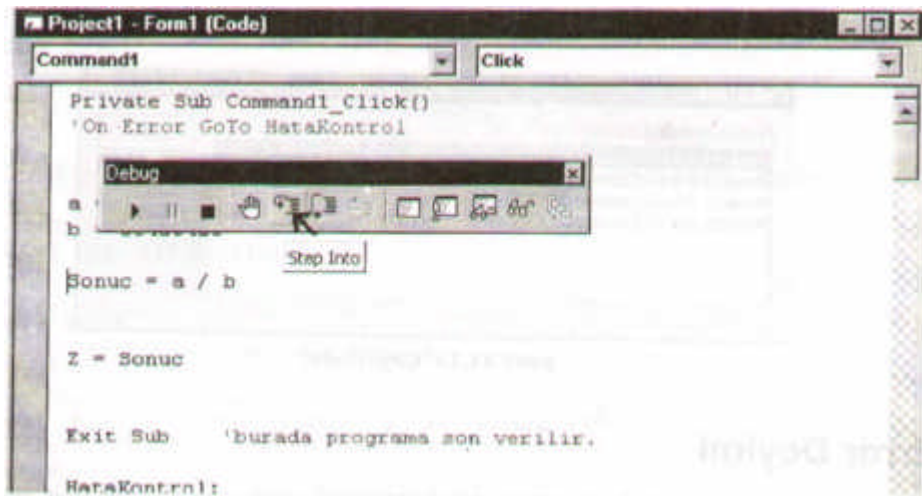
Watch Window ise projedeki ifadelerin izlenmesi için kullanilir. Watch Window penceresi üzerinde **Add Watch** komutu ile istenen ifade Watch Window'a eklenir. **Expression** ve **Value** sütununda çift tiklanarak degisiklik yapılabilir.



Sekil-10 Watch penceresi

Programı Kaldığı Yerden İletme

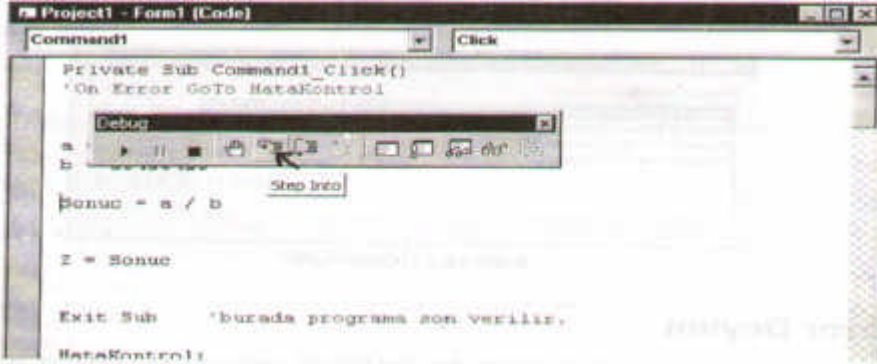
Step Into düğmesi ile program kesildikten sonra; program deyimlerini (satırlarını) adım adım çalıştırmayı sağlar. **Step Into** programı kaldığı yerden devam ettirir. **Step Over** düğmesi procedure'yi bir birim olarak iletir. Step Over düğmesine tıklandığında bulunan procedure içindeki bir sonraki deyim iletir. Bu işlemler için F8 ve SHIFT+F8 tuşları kullanılır. **Step Out** düğmesi ise procedure'in kalan deyimlerinin tamamı iletir (CTRL+SHIFT+F8).



Sekil-11 Kaldığı yerden devam etme

Kesme Noktası (Toggle Breakpoint)

Visual Basic kesme noktasından önce programın isletimini durdurur. Kod içinde kesme noktası ile işaretlenen satır isletilmeden önce program durur. Kesme noktası genellikle programı belli bir yerde durdurup o asamada bazı degiskenlerin degerlerinin kontrol edilmesini saglar. **Toggle BreakPoint** isaretinin kaldırılması için aynı düğmeye bir kez daha basılır.



Sekil-12 Kesme noktası

Kodun Yenibastan Çalıştırılması

Degisken tanımlamalarında yapılan degisikliklerin ardından ya da yapılan «|ista düzeltmelerinin ardından kodun yeni bastan çalıştırılması gerekebilir. Bu durumda **Stop** düğmesi kullanılır.

Çağrılar Diyalog Kutusu

Program kesildiğinde procedure içinde o anda çağırılmış diğer procedure'leri listeler. Özellikle iç içe çağırılmış procedure'lann kontrolünde kullanılır.

On Error Deyimi

Hata olusumunda; mesaj vermek ve programın kontrolünü saglamak için belli bir program parçasının isletilmesini saglanabilir. Bu işlem için hata kontrolüyle ilgili olarak bir program blogunun yazilmasini saglanır.

Yapisi:

On Error GoTo satir

On Error Resume Next

On Error GoTo 0

GoTo 0 : Procedure'daki hata kontrolünü iptal eder.

GoTo satir/ etiket : Hata kontrolü satirinin belirtilen satirdan ya da etiketten baslatilmasini saglar.

Resume Next : Hatanin ardindan programin bir sonraki deymi isleyerek programi çalistirmayi sürdürmesini saglar.

Eger programci, programinda bir **On Error** deymi kullanmaz ve olasi hatalari kontrol altina ,almazsa; hata olustugunda program çalışma zamanı içinde hatayı verir ve kesilir. Böyle bir durumla karsilasmak hiçbir programda onaylanmayacağı için, programin olası durumlara göre hata kontrol mekanizmalarına sahip olması gerekir.

Program içinde bir hata denetimi yapilrsa; hata olustugu anda program kontrolü, hata kontrolü kismina atlar. Hata denetimi; Resume, Exit Sub ya da Exit komutuna kadar aktif kalir. Hata denetiminin ardindan hatanın tanımlanması için **Err** nesnesinin **Number** özelligi kullanilir.

Asagidaki örnekte bir dosyanin tanımlanmasının ardindan; dosya açikken dosya silinmeye (yok edilmeye) kalkisilmektedir. Bu durumda hata olur:

```
private Sub Command3_Click ()
On Error GoTo HataKontrol
' hata denetimi için bu bölüme git.
Open "TESTFILE" For Output As #1 ' dosya tanimi.
Kili "TESTFILE" ' dosyayi silme islemi.
Exit Sub ' çık.
Hata Kontrol:
Select Case Err.Number
' hata numaralarini kontrol.
Case 55: MsgBox "Dosya zaten açık" dosya açık hatasi
Close #1 ' dosyayi kapat.
Case Else
End Select
Resume
End Sub
```

Asagidaki örnekte **OnError** ile 11 (Division by Zero) hatasi kontrol edilmektedir:

```
Sub Button32_Click ()
On Error GoTo HataKontrol
A = 2
B = 0
```

Microsoft Visual Basic 6.0

```
Sonuç = A / B
Exit Sub 'burada programa son verilir.
Hata kontrol:
Select Case Err.Number
Case 11: Mesaj = "Sifira bölünme hatasi olustu!!!"
End Select
MsgBox Mesaj
Resume Next
End Sub
```

Örnek: Ekrandan degerin yanlis girilmesi hatasina karsi önlemler

```
Private Sub Command1_click()
On Error GoTo HataKontrol
Dim Tüketim As Double
Dim Para As Double
Dim Mesaj As String
'deger alma
Tüketim= InputBox ("tüketim degerini girin")
Select Case Tüketim
Case 0 To 30#
Para = Tüketim * 200000
Case 31# To 101#
Para = (30# * 20000) +(70# * 40000) + ((tüketim-
100#)*100000#)
End Select
Msgbox str(para)
Exit sub
Hata kontrol:
Select case err.number
Case 6: mesaj ="tasma hatasi oldu!!"
Case13: mesaj= "yanlis tipli bir deger girdiniz!!"
End Select
Msgbox mesaj
Resume next
End sub
```

Hata Denetimi İçin Yöntemler

Hata denetimi için gerekli olan hata degerlendirme deyimlerinin **her bir** procedure'da ayn ayn yapılması yerine bütün hataların yer aldığı bir genel amaçlı procedure yazılır ve proje içincde açılan standart bir Modül için konur. Projeye eklenen bir modül içinde:

```
Sub Hatalar O
'olasi hatalarin listelenmesi
Select Case Err.Number
Case 6 : Mesaj= "Tasma hatasi oldu!!"
```

```
Case 7 : Mesaj= "Bellek yetersiz"
Case 9 : Mesaj= "Dizi disi eleman"
Case 11: Mesaj= "Sifira bölünme hatasi olustu!!!"
Case 13: Mesaj= "Yanlis tipli bir deger girdiniz!!"
Case 52: Mesaj= "Yanlis dosya"
Case 53: Mesaj= "Dosya bulunamadi"
Case 55: Mesaj= "Dosya zaten açık"
Case 57: Mesaj= "Giris/çikis aygiti hatasi"
Case 58: Mesaj= "Dosya zaten var"
Case 61: Mesaj= "Disk dolu"
Case 67: Mesaj= "Çok fazla dosya"
Case 71: Mesaj= "Disk hazir degil"
End Select
MsgBox Mesaj
```

Ardindan programların hata kontrol bölümlerinde bu procedure çağirilir:

```
HataKontrol:
    Hatalar    'hatalar procedure'i çağirilir
    Resume Next
End Sub
```

NOT

Projeye böyle bir modül eklemek için Project menüsünden Add Modüle komutu kullanilir.

Resume Deyimi

Programın çalışması sırasında; çalışma zamanı hatası ile karsilasildiginda "hata kontrolü" kismina sapilir. Hata kontrolü kisminin Isletilmesinin ardindan hata mesajı verilir. Sonra ise ya program kesilir ya da hatalı deyim atlanır. İşte **Resume** deyimi hatalı deyimin atlanarak programın isletimine devam etmesini saglar.

Yapisi: **Resume**

Resume Next

Resume satir

Resume deyimlerinin islevleri sunlardır:

Deyim

Açıklama

Resume : Program kontrolü hata olusan deYiminden devam eder.

Resume Next : Program hata olusan satirdan bir sonraki satira döner.

Böylece hata gögüslenmis olur.

Resume satir : Program satir ile belirtilen satir ya da etikete geri döner.

Belirtilen satir hata kontrolünün yapıldığı procedure içinde olmalıdır.

```
Sub Button1_Click ()
    On Error GoTo HataKontrol
    A = 2
    B = 0
    Sonuç = A/B
    Exit Sub
HataKontrol:
    Select Case Err.Number
        Case 11: Mesaj = "Sifira bölünme hatasi olustu!!"
    End Select
    MsgBox Mesaj
    Resume Next
End Sub
```

Err Nesnesi

Çalışma zamanı hataları {run-time errors} bakında bilgi verir. Err nesnesinin varsayılan özelliği Number'dır. Çalışma zamanı hatası olustugunda **Err** nesnesi hatayı temsil eden hata bilgisini içerir. **Exit Sub**, **Exit Function**, **Exit Property** ya da **Resume Next** deyiminin ardından **Err** nesnesinin değeri sıfırlanır. Bunun dışında Err nesnesi sıfırlanmak istenirse Clear metodu kullanılır. Asagidaki örnekte Err nesnesinin **Clear ve Raise** metotları kullanılmaktadır. Clear metodu hata değerini silerken, Raise metodu istenilen hata değerini oluşturmaktadır:

```
Private Sub command4_click()
    Dim Mesaj
    ' hata denetimi.
    On error Resume next
    Err.clear
    Err.raise 6 ' "Overflow" hatasi.
    ' hata numarasinin denetimi
    if err.number <> 0 then
        Mesaj = "hata # " & str(err.number) &
            " olusturuldu " & err.source & chr(13) &
err.description
```

```
        MsgBox Mesaj  
    End If  
End Sub
```

End ve Stop Deyimi

End deyimi uygulamayi (hemen) sona erdirir. End deyiminin ardından baska bir kod isletilmez ve baska bir olay olmaz. Bununla birlikte **Stop** deyimi ise uygulamayi durdurur. Stop deyimi hata giderme asamasinda kullanilir. Çünkü kullanılan nesneleri serbest bırakmaz.

```
Sub Button1_Click ()  
    'son  
    End  
End Sub  
Sub Button2_Click ()  
    stop  
End Sub
```

Ek- Hata Mesajlari,Sebepleri ve Çözümleri

Program yazarken sik sik karsilasilan hata mesajlarinin bir kismini asagiya almayi uygun gördük. Hata mesajıyla birlikte hatanın çıkis sebebi ve çözüm yollarini arastirdik.

Buradaki hata kodlarini kullanarak programinizin hata ayiklama kodlarinda hatanın sebebini de giderebilirsiniz.

```
On Error Goto Hata
```

```
Hata: If err=x then
```

gibi bir kodda x yerine asagidaki hata numaralarini kullanarak hatanın olusma sebebini program içerisinde öğrenip uygun kodlari yazabilirsiniz.

Çalışma zamanı hata mesajlari

Program hatasiz olarak yazilip çalıştirilmasina ragmen, program çalışirken kullanılan ifadelerden veya bilgisayarın o anki durumundan dolayi hatalar olusabilir. Asagida bunların sik olusmaları verilmistir. İyi yazilmis bir program bir islemi yapmadan önce gerekli hata yakalayicilari kurarak olusan hatayi degerlendirmeli ve kullanıcıya alternatifler sunmalidir.

Buradaki hata kodlarini kullanarak programinizin hata ayiklama kodlarinda hatanın sebebini öğrenip giderebilirsiniz.

```
On Error Goto Hata
```

```
Hata: I f err=x then
```

gibi bir kodda x yerine asagidaki hata numaralarini kullanarak hatanin olusma sebebini program içerisinde öğrenip uygun kodlari yazabilirsiniz.

Invalid procedure call or argument (Error 5)

Bir fonksiyona geçersiz bir deger girildiginde bu hata olur. Örneğin Sqr(-1) ifadesi bu hatayi olusturur. Çünkü negatif sayilarin kökü olmaz.

Overflow (Error 6)

Bu hata genellikle bir degiskene sinirlan disinda bir deger atandiginda meydana gelir. Örneğin Integer olarak tanimlanmis bir degiskene 1000000 sayisi atanirsa bu hata olur.

Out of memory (Error 7)

Bellek bittiginde bu hata meydana gelir. Windows altinda çok fazla programi çalistiriyorsaniz bazilarini kapatmayi deneyin.

Yazdiginiz Modüle veya Prosedür çok uzun oldugunda bu hata meydana gelir. Bunlari uzunlugu 64K boyutunu geçemez. Kodunuzu farkli prosedürlere dagitin.

Subscript out of range (Error 9)

Dizi içinde var olmayan bir elemana erisim yapmak istenmistir. Erisilmek

istenen eleman muhtemelen dizi araliginda tanimlan alt veya üst sinirdan ya

büyüktür yada küçüktür. Bu durumda dizinin sinirlarini kontrol etmelisiniz.

Eger belirtilen dizi elemani bir degisken olarak tanimlanmissa o zaman degisken adinin dogrulugunu kontrol etmelisiniz.

Bir dinamik dizi tanimlamissiniz fakat dizinin eleman sayisini belirtmemissiniz.

Bu durumda da bahsi geçen hata meydana gelecektir:


```
Dim DinamikDizi () As Integer  
DinamikDizi (1( =100 '9 nolu hata meydana gelir..
```

Bu durumda dizi boyutunu Redim kullanarak belirlemeniz gerekir.

```
Dim DinamikDizi() As Integer  
Redim DinamikDizi(10)  
DinamikDizi (1) =100 ' 9 nolu hata meydana gelir..
```

Division by zero (Error 11)

- Bir sayı sifıra bölünüyorsa bu hata meydana gelir: Bu durumda ifadedeki değişkenlerin değerlerini kontrol edin.

Type mismatch (Error 13)

- Değişken veya özelliğe atanan değer doğru tipte değildir. Örneğin tamsayı olarak tanımlanmış bir değişkene string bir ifade aktarıldığında bu hata mesajıyla karşılaşılar.

Resume without error (Error 20)

- Resume deyimi sadece hata yakalayıcı deyimlerinden sonra kullanılabilir.
Hata yakalama kodu dışına Resume deyimi kullanmıyorsunuz. Deyimi hata yakalama aralığına alın yada silin.
- Herhangi bir hata olmamasına karşın sizin kodunuz hata yakalama bölümünü giriyorsa bu hata oluşur. Bu çoğunlukla hata ayıklama kodundan önce Exit Sub yazmamanızdan kaynaklanır. Hata olmadığı halde hata ayıklama kodu devreye giriyordur.

Out of stack space (Error 28)

- Stack alanı programların kullandıkları önemli bir hafıza bölgesidir. Çok sayıda Function, Sub veya Property çağırısı yapıldığında, özellikle kendi kendini
- 685

Microsoft Visual Basic 6.0

çağırılan fonksiyonlarda bu hata meydana gelebilir. İki prosedürün birbirlerini sürekli çağırmadıklarından emin olun.

- Bu hata iki kontrolün Change olaylarının birbirlerini sürekli değiştirmelerinden de meydana gelebilir. Change olayına yazdığınız kodu kontrol edin.

Sub, Function, or Property not defined (Error 35)

- Tanımlanmamış bir özellik veya prosedür çağrıldığında bu hata meydana gelir. Çağırduğunuz prosedürü tanımlayın.
- Prosedürün ismi yanlış yazılmış olabilir. İsmin doğruluğunu kontrol ederek bir yanlışlik varsa düzeltin.
- İstenilen prosedür başka sınırlanmış içinde tanımlanmış olabilir. Bir modül içinde Private olarak tanımlanan prosedürler o modülün dışındaki prosedürler tarafından çağrılmazlar.

Error in loading DLL (Error 48)

- Ulaşılmak istenen dosya bir Windows DLL dosyası değildir.
- Ulaşılmak istenen DLL dosyası başka bir DLL dosyasına ihtiyaç duyuyor fakat bu DLL dosyası bulunamamaktadır.

Bad DLL calling convention (Error 49)

- Çoğunlukla DLL dosyasından çağırdığınız bir fonksiyonun parametreleri orijinal parametrelerle aynı tipte olmadığında bu hata oluşur. Declare

ifadesini
kontrol edin.

Bad file name or number (Error 52)

- Erisilmek istenen dosya adi yada dosya numarasinin ait dosya ya kapalıdır yada Open deyimi ile belirtilen dosya değildir. Verdiginiz Ismi veya numarayı kontrol edin.
- Verdiginiz dosya numarası 0-511 arasında değilse de bu hata oluşur. Dosya numarası bu sinirin dışında olamaz.
- Verdiginiz dosya ismi geçersiz olabilir. Dosya isimleri 255 karakteri geçemez, *,: gibi özel karakterler içeremez, CON,LPT1,PRN gibi özel isimler olamaz.

File not found(Error 53)

- Kill,Name, veya Open deyimleriyle erişilmek istenen dosya bulunamamaktadır. Bu durumda dosya adının ve dosyanın bulunduğu yolun doğruluğunu kontrol etmelisiniz.

Bad file mode (Error 54)

- Dosyaya yaptığınız işlem o dosyaya uymuyorsa bu hata oluşur. Input, Output, Append modunda açtığınız dosyalarda Get,Put komutlarını, Random olarak açtığınız dosyalarda ise Print, Input, Line Input komutlarını kullandığınızda bu hata meydana gelir.
- Read Only modda açtığınız dosyaya yazmaya çalıştığınızda bu hata meydana gelir.

File already open (Error 55)

- Dosya açık olduğu halde tekrar dosya açma girişiminde bulunulmuştur.

Microsoft Visual Basic 6.0

- Yada dosya Input modunda açık olduğu halde Output modunda tekrar açılmaya çalışılıyordur.
- Kill, SetAttr, veya Name deyimleri açık bir dosya için kullanıldığında da bu hata mesajı meydana gelir. Bunun için de bu deyimler kullanılmadan önce açık olan dosya kapatılmalıdır.

Device I/O error (Error 57)

- Diskete veya yazıcıya yazdırırken bir hata oluşmuştur. Örneğin kopyalama bitmeden disket çıkarılmıştır.

File already exists (Error 58)

- Bu hata genellikle Name komut ile bir dosyanın adı değiştirilirken, verilen yeni isimle aynı isimde bir dosya olduğunda oluşur.

Bad record length (Error 59)

- Get ve Put deyimleriyle kullanılan kayıt degiskeninin uzunluğu Open deyimiyile belirlenenden daha uzun olduğunda bu hata oluşur. Open deyiminin sonunda Len=x satiri ile kayıt boyunu artirin.

Disk fail (Error 61)

- Diskete yazan komutlarda disket veya harddisk dolarsa bu hata oluşur.

Input past end of file (Error 62)

- Dosyadan okurken dosya sonu geçilirse bu hata oluşur. EOF(#no) ile dosya sonunu kontrol etmelisiniz.

Bad record number (Error 63)

- Put veya Get deyimleri ile birlikte kullanılan kayıt numarası sıfır veya dah küçük olamaz. Bu durumda verilen kayıt numarası kontrol edilmelidir.

Too many files (Error 67)

- Bu hata genellikle anda açılacak dosya limitinin aşılması durumunda meydana gelir. Bu durumda açık olan bazı dosyalar kapatılmalı veya config.sys dosyasında yer alan files deyiminin karşısındaki sayısal değer artırılmalı ve bilgisayar yeniden başlatılmalıdır. Files=60 gibi.
- Bu hata ana dizinde 512 den fazla dosya olduğunda da meydana gelir. Ana dizinde oluşturulabilecek maksimum dosya sayısı 512'dir. Dosyayı farklı dizinlere kaydetmeyi deneyin veya ana dizindeki bazı gereksiz dosyaları farklı dizine kaydedin.

Permission denied (Error 70)

- Write protect diskete yazarken bu hata oluşur. Disketin proteğini açın
- Bir anda çalışıyorsanız o dizinde kaydetme hakkınız olmayabilir.
- Erismeye çalıştığınız kayıt veya dosya başka bir kullanıcı tarafından kilitlenmiş olabilir. Diğer kullanıcının kapatmasını bekleyin. Eğer program ağ altında çalışacak şekilde tasarlanıyorsa bu hatayı tespit edip, kayıt kilidi açılana kadar beklemeyi sağlayacak kodu yazın.

Disk not ready (Error 71)

- Diskete yazan komutlarda sürücüde disket bulunmazsa bu hata oluşur.

Path/File access error (Error 75)

- Read Only bir dosyaya yazmaya çalıştığınızda bu hata oluşur. Dosyanın Read Only özelliğini kaldırın.

Path not found (Error 76)

- Verdiğiniz yol yanlış olduğunda bu hata meydana gelir. Verdiğiniz yolu kontrol edin.

Object variable or With block variable not set (Error 91)

- Nesne değişkenlere Set değerini kullanmadan atama yapamazsınız. Esitliğin başına Set koyun.

Dim x As Node

x = TreeView1 . SelectedItem ataması geçersizdir.

Set x = TreeView1.SelectedItem
şeklinde kullanılmalıdır.

Can't use character device names in file names (Error 320)

- Bir dosya ismine AUX, CON, COM1, COM2, LPT1, LPT2, LPT3, LPT4, NUL gibi işletim sistemine özel değerleri vermezsiniz.

Object already loaded(Error 360)

- Kontrol dizisi olarak tanımlanan kontrol elemanı zaten yüklenmiş. Çalışma zamanında bir dizi olarak tanımlanmış ve load değimiyle yüklenmek istenen kontrol elemanı daha önceden yüklenmiş olduğundan dolayı bu hata mesajı karşımıza çıkacaktır.

Load Text1(5)

Yukarıdaki satır 5 numaralı bir Text kutusu oluşturmaya çalışmaktadır. 5 numaralı text kutusu zaten varsa bu hatayı oluşturur.

Argument not optional or invalid property assignment(Error 449)

- Fonksiyon parametreleri eksik verilmiştir. Mesela left(ifade,adet) deyiimi iki adet parametreye ihtiyaç duyar, bunlardan birisi yazılmazsa yukarıdaki hata meydana gelecektir.

Error loading from file(Error 31037)

- Muhtemelen FileNumber olarak verilen dosya numarası geçersizdir.
- Dosya Binary modda açılmıyor olabilir.
- Dosya zarar görmüş olabilir.

Error saving to file (Error 31036)

- Dosyaya kayıt yapılamıyor. Verilen dosya numarası geçersiz olabilir.
- Ulaşılmak istenen dosya Binary modda açılmıyordur.
- Disk veya diskette yeterli boş yer bulunmamaktadır.

Tasarım Zamani Hatalari

Programlar yazilirken hatalar yapilmissa VB o programi calistirmayi reddedecektir. Hata giderilinceye kadar program calistirilamaz. Asagida çok karsilasilan mesajlari verdik. Bunlari izleyerek hatayi gidermeye calismalisiniz.

A procedure of that name already exists

- Oluşturulmak istenen prosedürün ismini taşıyan başka bir prosedür vardır.

Aynı aralıktaki prosedür adları farklı olmak zorundadır. Bu durumda aynı olan prosedür isimlerini değiştirmelisiniz.

End If without block If

- If kullanılmadan End If kullanılmış. Herbir End If mutlaka bir If deyimine karşılık gelmelidir. Bu durumda If-End If blok yapısını kontrol etmelisiniz.

End Select without Select Case

- Select Case kullanılmadan End Select kullanılmış. End Select deyimi mutlaka Select Case deyiminden sonra gelmelidir.

Select Case without End Select

- Select Case kullanılmış fakat End Select kullanılmamış. Blogun sonuna End Select koyun.

Exit Do not within Do...Loop

- Exit Do ifadesi Do...Loop deyimi içinde değildir. Exit Do deyimi sadece Do...loop deyimi içinde kullanıldığı zaman geçerlidir.

Exit For not within For...Next

- Exit ifadesi For-Next deyimi içinde değildir. Exit For deyimi sadece For-Next döngüsü içinde kullanıldığı zaman geçerli olur.

Exit Function not allowed in Sub or Property

- Sub veya property'de Exit Function deyimi kullanılmaz. Bunun için Exit Sub yada Exit Property ifadelerini kullanmalısınız.

Exit Property not allowed in Function or Sub

- Sub veya fonksiyonda exit Property deyimi kullanılmaz. Bunun için Exit Sub yada Exit Function ifadelerini kullanmalısınız.

Expected End Function

- Bir fonksiyonu sonlandırmak için End Function kullanılmalıdır. Fonksiyonun sonunda End Function kullanmamışsınız.
- Bir Property sonlandırmak için End Property ifadeleri kullanılmalıdır. Property sonunda End Property kullanmamışsınız,

Expected End With

- With deyimini kullanmışsınız fakat End With deyimi ile sonlandırmamışsınız.

For without Next

- For kullanılmış fakat bunun sonunda bulunması gereken Next kullanılmamış. Her bir for deyimine karşılık mutlaka Next deyimi bulunmalıdır. Eksik olan Next deyimlerini uygun yerlere yazmalısınız.

Label not defined

- Goto deyimi ile yönlendirilmiş etiket bulunamıyor. Goto deyiminden kullandığınız etiketi yazın.

Microsoft Visual Basic 6.0

On error goto hata

hata: 'etiket bu şekilde tanımlanmalıdır

Syntax Error

- Bu hata Visual Basic'in herhangi bir ifadesinin yanlış yada eksik yazılması durumunda meydana gelir. Yazdığını ifadeyi kontrol edin. Özellikle (i ve l) harflerini kullanmayın.

Variable not defined

- Tanımlanmamış bir değişken kullanıldığı zaman bu hata meydana gelir. Bu durumda bahsi geçen değişken içinde bulunan prosedürde Dim veya Static deyimi ile tanımlanmalıdır.

Duplicate declaration in current scope

- Bir değişken aynı yerde iki defa tanımlanırsa bu hata oluşur. Tanımlardan birini silin.

Too many dimensions

- Diziler en fazla 60 boyutlu olabilirler. Daha büyük yapılan bir tanımlama yukarıdaki hata mesajına neden olacaktır.

Program kodu yazarken karşılaşılan hata mesajları

Kullanıcı değişken tanımlarken tanımlamalardan birisini unuttuğu zaman aşağıdaki mesaj penceresi ile uyarılır:

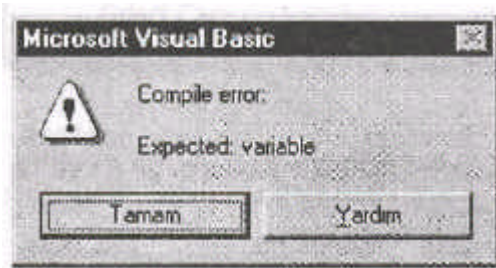
Dim ifadesi yazilip enter tusuna basildigi zaman yandaki mesaj penceresi ile kullanıcı uyarilir.

Mesajın anlamı: Derleme hatası, Tanımlayıcı bekleniyordu

a Dim x,y,

Biçiminde kullanildigi zaman bu defa kullanıcı asagidaki gibi bir mesaj penceresi ile uyarilir:

Değişken bekleniyordu

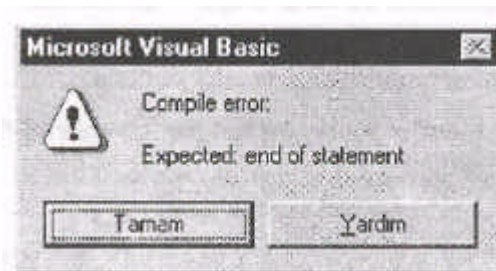


Değişkenler tanımlanırken tek tek tanımlanabileceği gibi aralarına "," konulmak suretiyle tek satırda da tanımlanabilir. Yukarıda verildiği gibi son değişken için "," kullanılmış fakat değişken tanımlanmamıştır.

Mesajın anlamı: Derleme hatası,

if x=1 then ifadesi ifx=1 then gibi, for x=1 to 100 ifadesi forx=1 to 100 gibi

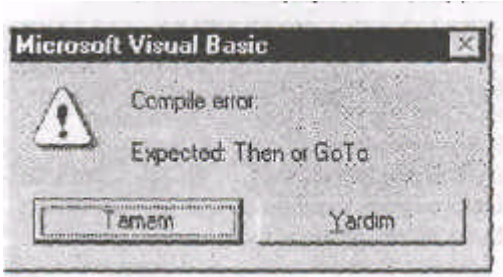
yazıldığında aşağıdaki mesaj penceresi ile kullanıcı uyarılacaktır:



Mesajın anlamı: Derleme hatası, deyim devami bekleniyordu.

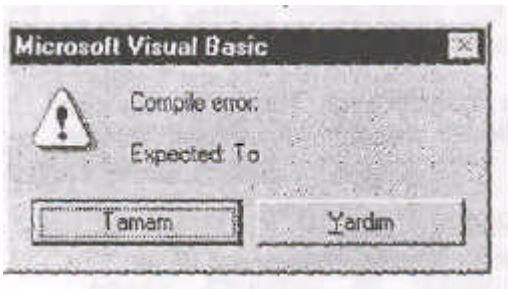
if x=1 ifadesi yazılıp bir sonraki satıra geçilmek istenirse yani then unutulurca aşağıdaki mesaj penceresi ile kullanıcı uyarılacaktır:

Microsoft Visual Basic 6.0

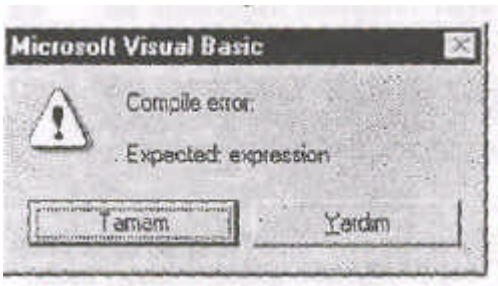


Mesajın anlamı: Der/eme hatası Then veya Goto bekleniyordu

For x=1 ifadesi yazılıp enter tusuna basıldığında aşağıdaki mesaj penceresi ile kullanıcı uyarılacaktır:



"Command1.Caption=" gibi bir ifadenin sol tarafı yazılıp sağ tarafı yazılmadan enter tusuna basılırsa aşağıdaki gibi bir mesaj penceresi ile kullanıcı uyarılacaktır:



Mesajın anlamı; Derleme hatası deyim bekleniyordu

Program yazmak; belli bir kodlama kuralları ya da diğer ortam kurallarına bağlı olduğu için programın gelişmesi sırasında belli hatalar yapılır. Bu hataların bulunup giderilmesi hata giderme çalışmasını oluşturur. Visual Basic, programların yazılması {derlenmesi ve çalıştırılması} sırasında

karsilasilan hatalara karsi gelistirilmis bir hata bulma ve düzeltme (debugging) olanagina sahiptir. Hata türleri:

- Derleme hatalari (compile errors)
- Çalışma zamani hatalari (run-time errors)
- Mantik hatalan (logical errors)

Derleme (compile) hatalari, programi yazarken yapilan deyimsel ya da sözdizimi hatalaridir. Çalışma zamani hatalan (run-time error) ise program çalıştigi sirada ortaya çıkar. Örneğin olmayan bir dosyayı açmaya kalkmak ya da sabit diskte yer katmaması ya da sifıra bölünme (**divide by zero**) gibi.

Çalışma zamani (run-time) hatalarina karsi herhangi bir önlem alınmazsa program hata verir ve kesilir. Diğer bir açıdan; bu durum kullanıcılar karsısında düşülebilecek en kötü durumlardan birisidir. Bu nedenle çalışma zamani hatalarina karsi belli önlemler daha önceden alınmalıdır Bu hatalarin tanımlanıp önlemlerini daha önceden alınması işlemi hatalarin tuzaklanmasını ortaya çıkarır.

Hata tuzaklama **On Error** deyimi ile yapılır. On Error deyim ile tuzaklanan hata değerlendirilerek hata kontrol altına alınır.

Hata giderme konusunda bir diğer durum ise programın kesilmesi (break) modudur. Programın kesilmesinin ardından **Debug** (Hata Giderme) araç çubuğu kullanılarak programın hataları saptanır. Hata giderme araçlarının başında değişkenleri ve programdaki belli yerleri izleyebileceğimiz pencereler gelir.

Locals Window. Program kesildiğinde bulundan procedure içinde tanımlanan bütün değişkenlerin değerini verir.

Immediate Window. Program kesildikten sonra; procedure içinde bir değişkenin ya da bir ifadenin seçilerek değerinin ne olduğuna bakılmak istenirse o zaman anlık izleme penceresi kullanılır.

Watch Window. Watch Window ise projedeki ifadelerin izlenmesi için kullanılır.

16-API Nedir ?

Windows API 'lerinin VB 'de kullanimina girmeden önce API nedir onu açıklamaya çalisalim.

Isletim sistemlerine duyulan ihtiyaçlardan biri standart olarak her program tarafından yapılmasi gereken seyleri ortak bir çatı altında toplamak ve programları sistemde belirli kuralları altında çalışmasını sağlamaktır. Isletim sistemlerinin degerini anlamak için isletim sistemi olmayan bir bilgisayar düşünün.

Yaptığınız programları diske kaydetme ihtiyacınız var. Isletim sisteminiz yoksa programlarımızı diske yazacak ve okuyacak assembly kodları sizin yazmanız gerekecektir. Ve her program diske yazma ve okuma kodlarını içinde bulundurmak zorunda olacaktır. Ayrıca diske yazacağınız programı diskin neresine yazacaksınız. Tabiki herkes kendi programının bası yazılmasını isteyecektir. Bu da diski paylaşım sorununu çıkaracaktır. Ayrıca yazıcı için de problem vardır. Her yazıcı aynı sistemle çalışmayacağı için programınızda yazdırma işlemleri de varsa belli baslı yazıcı tipleri için gerekli kodları yazmanız gerekecektir. Bu örnekler çoğaltılabilir.

Iste PC'ler ilk çıktığında disk işlemlerini kolaylastırmak için DOS ta piyasaya çıktı. DOS disk işlemlerini yapmak için yazılım interruptlarını programcıların hizmetine sunmıstı. Diskle ilgili bir işleminiz için INT X 'in Y numaralı servisini çağırıyordunuz ve bu işlemleri sizin yerinize DOS yapıyordu. Sistemler geliştiçe bilgisayar değişik alanlara da hitap etmeye başlayınca çok değişik arabirimler de çıktı, DOS'a grafik, yazıcı işlemleri gibi standart işlemler de eklendi ve sistemde bulunan standart donanımların hemen hemen hepsine DOS veya BIOS interruptlarıyla erişebiliyordunuz. Ayrıca DOS programların belleği nasıl kullanacağını da belirtiyordu. DOS isletim sistemi olarak kullanıcıya herhangi bir standart arabirim sunmamıştır. Sadece programların sistemdeki standart donanımlara ulaşabilecekleri kodları kullanıcıya sunmıstır. DOS'ta yapılan programların hiçbirisi bir birine benzemez. Her program kendi kullanıcı ara yüzünü belirlemek zorundadır ve bunun için gerekli kodu kendisi yazmak zorundadır. DOS'ta yapılan programların kullanım ve programlanmasının zorluğu da bir ölçüde buradan kaynaklanır.

DOS'un programlara standart bir arabirim sunmaması, bellek sınırlarının olması gibi sebeplerden dolayı çok çok geç kalmış olsa da Windows çıktı. Windows DOS'un sağladığı standart donanıma ulaşma haricinde Ses kartları, Gelişmiş yazıcılar, Scanner 'lar gibi donanımların kullanımını da programların kullanımına sunar. Ayrıca programlara standart arabirimleri (Diyalog kutuları, Formlar, Kontroller gibi) kullanma imkanı da sunmıstır. DOS kendi servislerini yazılım interruptlarıyla sunarken Windows API'lerle sunar.

Simdi söyle bir sey düşünülebilir. DOS'ta çok program yaptim ama diske bir sey yazdirmek için DOS'un interruptlarini kullanmaya hiç ihtiyaç duymadim. Evet eger assembly program yazmadiysanız bunlara da ihtiyaciniz yoktur. Çünkü kullandiginiz programlama dili bu işi sizin yerinize yapıyordu. Bu VB 'de yaptiginiz programlarda da böyledir. API kullanacaksınız diye bir sart yok VB bunlari sizin yerinize kullanir. Ancak DOS'taki programlama dillerinde olduğu gibi VB 'de de programlama dilinin sundugu işlemler her zaman isinizi görmeyebilir, bu durumda Windows API'lerini kullanma ihtiyaci duyarsiniz.

Basic herhalde bütün zamanların en yavas programlar üreten dili olma özelligini kimseye kaptirmak istemiyor. Quick Basic'te yaptiginiz bir program, ayni işi yapan C ile yapilmis programdan en az 5 kat daha yavas çalışacaktır. Bu fark GW Basic'te daha da büyüktür. VB 'de de durumun iç açici olduğunu iddia etmek çok güç. Programlarınızda API kullanmanız bu hiz barajlarını asmanizi sağlayacaktır. Ayrıca VB'nin sunmadigi bazı işlemler için de API kullanmak gerekir. Örneğin sistemdeki bos bellek miktarini verecek herhangi bir komut VB'de bulunmaz bunu da yine API kullanarak öğrenmek zorundasınız.

Windows'un sundugu bu API 'ler gruplandırılarak bir çok DLL ve EXE dosyasına konmustur. VB'de kullanılan OCX dosyalarında da API 'ler bulunabilir. Bu API'lerden birini kullandiginızda API 'nin bulunduğu DLL sisteme daha önce yüklenmemisse önce bu DLL yüklenir ve API çalıştırılır.

Programınızda API kullanmak için Declare deyimiyle API 'yi tanımlamanız gerekir. Bu tanımdan sonra, tanımladiginiz API 'ye bir fonksiyon veya bir altprogram gibi ulaşabilirsiniz.

VB'de API Tanimi

VB'de API'ler iki şekilde tanımlanabilir. Fonksiyon veya altprogram olarak. Fonksiyon olarak tanımlanan API'lerden geriye bir deger dönerken, altprogram olarak tanımlananlardan bir deger geri dönmez. Alt program olarak API tanimi:

Private/Public Declare Sub isim Lib "libname" [(parametreler)]

Fonksiyon program olarak API tanimi:

Private/Public Declare Function isim Lib libname [(parametreler)]
[As tip]

Burada isim fonksiyonun ismidir ve programda API bu isimle çağrılır. Libname kullanılan kütüphanenin ismi, parametreler; fonksiyona giren parametreler, As tip; fonksiyondan dönen degerin tipidir.

API'nin tanımlanacağı yer formun veya modülün General-Declarations kısmıdır. API'yi bir formun Declaration kısmında tanımlarsanız, API'yi yalnız o formun altprogramlarından çağırabilirsiniz. Bu durumda API'yi Private Declaration ifadesi ile tanımlamanız gerekir. API'yi bir modülde tanımlarsanız programınızın her yerinde kullanabilirsiniz.

API'yi doğru olarak tanımladığınız halde VB, ilgili dosyada böyle bir API bulunmadığını söylüyorsa veya API ile aynı isme sahip bir VB komutu var ise bu durumda Alias isimleri kullanmanız gerekir.

Private/Public Declare Function/Sub isim Lib libname Alias "isim" [(parametreler)] [As tip]

API'yi doğru olarak tanımladığınız halde VB, ilgili dosyada böyle bir API bulunmadığını söylüyorsa API isminin sonuna A ekleyerek Alias ismi olarak vermeniz gerekir. Bunun sebebi Windows işletim sistemi farklı dilleri desteklemektedir. ANSI karakter setini destekleyen ülkeler için sonuna A harfi, Unicode veya iki karakter genişliğini kullanan ülke seti için ise sonuna W harfi eklemeniz gerekir.

API tanımı yaparken kullanacağınız tiplerin isimlerini ise C 'den VB' ye çevirmeniz gerekir. Genel olarak tip karşılıkları şöyledir.

C	Visual Basic
atom	ByVal degisken As Integer
bool	ByVal degisken As Long
Byte	ByVal degisken As Byte
char	ByVal degisken As Byte
C	Visual Basic
colorref	ByVal degisken As Long
dword	ByVal degisken As Long
hwnd, hdc, hmenu vb.	ByVal degisken As Long
int, uint	ByVal degisken As Long
long	ByVal degisken As Long
lparam	ByVal degisken As Long
lpdword	degisken As Long
lpint, lpuint	degisken As Long
lprect	degisken As type
lpstr, lpctr	ByVal degisken As String
lptest	degisken As Any
lpword	degisken As Integer
lresult	ByVal degisken As Long
null	degisken As Any veya ByVal degisken As Long
short	ByVal degisken As Integer
void	Sub procedure
word	ByVal degisken As Integer
wparam	ByVal degisken As Long
16 bit	ByVal degisken As Integer
32 bit	ByVal degisken As Long

float	ByVal degisken As Single
double	ByVal degisken As Double

Parametrelerden biri iki farkli tipte deger alabiliyorsa bunu As Any olarak tanımlamaniz gerekir. Hangi parametrenin Any olarak tanımlanmasi gerektigine ancak dosyadaki bilgileri okuyarak anlayabilirsiniz. Örneğin bir parametre hem string içerebiliyor ve hem de Null içerebiliyorsa bu parametre Any olarak tanımlanmalıdır.

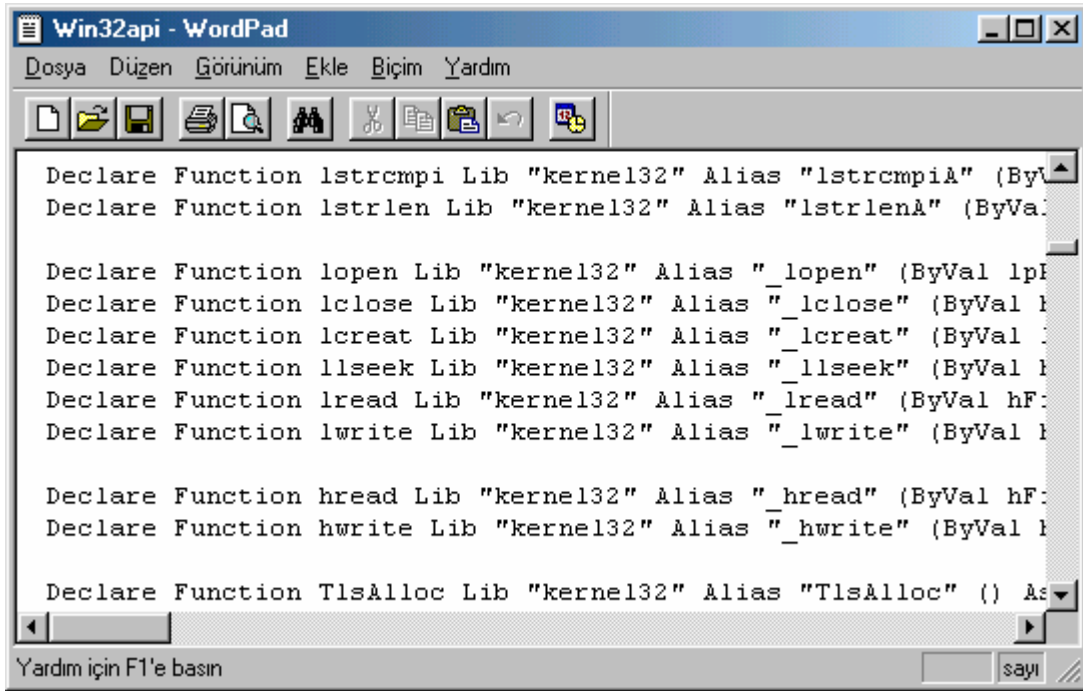
Yaptıkları işlere göre API'lerin bulundukları dosyalar ise şunlardır:

DLL	Fonksiyonları
Advapi32.dll	Sifre ve Kayıt dosyası işlemleri gibi gelişmiş bir çok API'ler
Comdlg32.dll	Diyalog pencereleri ile ilgili API'ler
Gdi32.dll	Grafik API'leri
Kernel32.dll	Çekirdek Windows API'leri
Lz32.dll	32 bit sıkıştırma API'leri
Mpr.dll	Multiple Provider Router Apileri
Netapi32.dll	32-bit Network API'leri
Shell32.dll	32-bit Shell API'leri
User32.dll	Kullanıcı arabirimi API'leri
DLL	Fonksiyonları
Version.dll	Versiyon işlemleri API'leri
Winmm.dll	Multimedia API'leri
Winspool.drv	Print spooler API'leri

API Tanım Dosyası

Yukarıda bir API'nin nasıl tanımlanacağını anlattık. Ancak çoğu API'nin VB'de nasıl tanımlanacağı WINAPI dizini altındaki WIN32API.TXT dosyasında verilmiştir. Bu dosyayı açarsanız istediğiniz API'nin, ona ait tiplerin ve sabitlerin tanımlarını bulabilirsiniz.

Bu dosyadaki tanımları formdaki bir modüle kopyalayarak o API'yi kullanabilirsiniz. Eğer modül kullanmadan form içinde tanım yapmak isterseniz bu tanımların başına Private yazmanız yeterlidir.

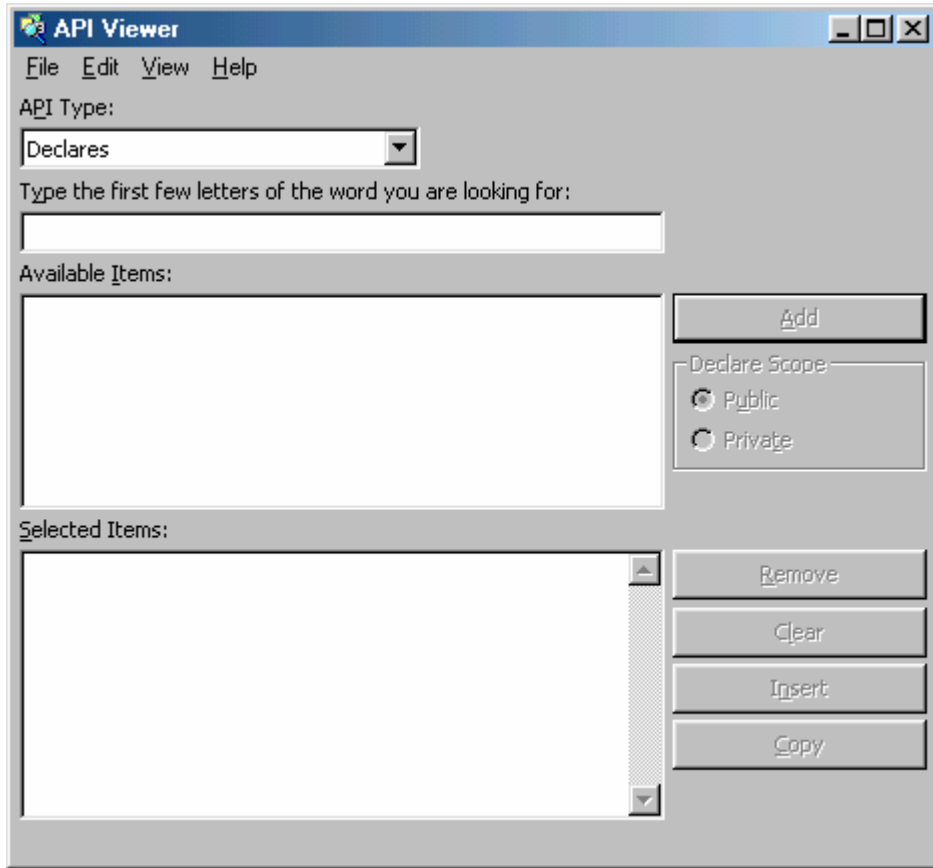


```
Declare Function lstrcmpi Lib "kernel32" Alias "lstrcmpiA" (ByVal lpStr1 As String, ByVal lpStr2 As String) As Integer
Declare Function lstrlen Lib "kernel32" Alias "lstrlenA" (ByVal lpString As String) As Integer

Declare Function _lopen Lib "kernel32" Alias "_lopen" (ByVal lpFileName As String, ByVal dwFlags As Integer) As Integer
Declare Function _lclose Lib "kernel32" Alias "_lclose" (ByVal hFile As Integer) As Integer
Declare Function _lcreat Lib "kernel32" Alias "_lcreat" (ByVal lpFileName As String, ByVal dwMode As Integer) As Integer
Declare Function _llseek Lib "kernel32" Alias "_llseek" (ByVal hFile As Integer, ByVal dwOffset As Integer, ByVal dwOrigin As Integer) As Integer
Declare Function _hread Lib "kernel32" Alias "_hread" (ByVal hFile As Integer, ByVal lpBuffer As String, ByVal dwBytes As Integer) As Integer
Declare Function _hwrite Lib "kernel32" Alias "_hwrite" (ByVal hFile As Integer, ByVal lpBuffer As String, ByVal dwBytes As Integer) As Integer

Declare Function TlsAlloc Lib "kernel32" Alias "TlsAlloc" () As Integer
```

Yukarıdaki text dosyasında bulunan tanımları daha uygun bir formatta gösteren bir program da WINAPI dizini altında bulunur. Bu dizindeki APILOAD.EXE programını çalıştırırsanız text dosyasının içindeki API, tip tanımı ve sabitleri düzenli bir yapıda görebilirsiniz.



APILOAD.EXE dosyasi ile API Viewer programini çalistirdigimizda yukaridaki gibi bos bir pencere gelir. File-Load Text File menüleri ile WIN32API.TXT dosyasini bu programa yükleyin.

Programin üst kisminda bulunan ComboBox araciligi ile hangi bilgilerin gösterilecegini belirleyebilirsiniz. Bu listede üç seçenek bulunur. Constants: Sabitleri, Declares: API tanımlarini, Types: Tip tanımlarini listeler.

Programda, pencerenin orta kismindaki listeden bir ismi çift tıkladiginizda ona ait tanım alttaki pencerede listelenir. Bu tanımi VB'ye

aktarmak için penceredeki Copy düğmesine basabilir, VB'ye geçtikten sonra da Ctrl+V tuşları ile oraya alabilirsiniz.

Windows ve Sistem hakkında Bilgi Veren API'ler

Bos bellek miktarını öğrenmek

```
Declare Sub GlobalMemoryStatus Lib "kernel32" (IpBuffer  
As MEMORYSTATUS)
```

Bu API Windows altında kullanılacak bellek hakkında bilgi verir. Fonksiyonun kullandığı MEMORYSTATUS aşağıdaki gibi tanımlanmıştır. Bu iki tanımı bir modül içine veya formun general declarations kısmına yazmanız gerekir. (Formda tanımlarsanız Declare ve Type ifadelerinin başına Private yazmayı unutmayın.)

```
Type MEMORYSTATUS  
    dwLength As Long  
    dwMemoryLoad As Long  
    dwTotalPhys As Long  
    dwAvailPhys As Long  
    dwTotalPageFile As Long  
    dwAvailPageFile As Long  
    dwTotalVirtual As Long  
    dwAvailVirtual As  
Long  
End Type
```

dwMemoryLoad: Belleğin kullanım yüzdesini verir.

dwTotalPhys: Toplam fiziksel bellek(RAM) miktarını byte olarak verir.

dwAvailPhys: Fiziksel bellekteki boş miktarı byte olarak verir.

ÖRNEK: Örnek olarak bir Timer aracılığı ile belleğin durumunu sürekli gösterecek program yapalım.

```
Option Explicit
```

```
Private Declare Sub GlobalMemoryStatus Lib "kernel32"  
(IpBuffer  
As MEMORYSTATUS)
```

```
Private Type MEMORYSTATUS  
    dwLength As Long  
    dwMemoryLoad As Long
```

```

        dwTotalPhys As Long
        dwAvailPhys As Long
        dwTotalPageFile As Long
        dwAvailPageFile As Long
        dwTotalVirtual As Long
        dwAvailVirtual As
    Long
End Type

Private Sub
Form_Load()
Timer1.Interval = 1000

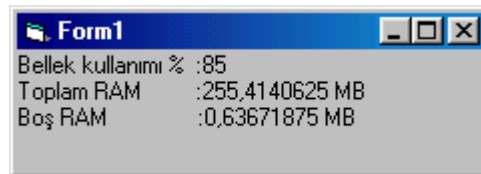
End Sub

Private Sub
Timer1_Timer()

Cls

    Dim m As MEMORYSTATUS
    GlobalMemoryStatus m
    Print "Bellek kullanımı % : " , m.dwMemoryLoad
    Print "Toplam RAM:", , m.dwTotalPhys / 1024 / 1024 & "
    MB" Print "Boş RAM:", , m.dwAvailPhys / 1024 / 1024 &
    " MB"
End Sub

```



Mikroislemci tipini ve sayisini öğrenmek

Declare Sub GetSystemInfo Lib "kernel32" (IpSystemInfo as SYSTEM_INFO)

Bu API, bilgisayardaki mikro islemcinin özelliklerini asagidaki SYSTEM_INFO yapısında verir.

```

Type SYSTEM_INFO
    dwOemID As Long
    dwPageSize As Long
    IpMinimumApplicationAddress As Long `Any

```

Microsoft Visual Basic 6.0

```
lpMaximumApplicationAddress As Long 'Any
dwActiveProcessorMask As Long
dwNumberOfProcessors As Long
dwProcessorType As Long
dwAllocationGranularity As Long
dwReserved As
Long
End Type
```

dwProcessorType:

İşlemcinin tipini belirten bir sayı geri döner. 386, 486, 586 (Pentium) işlemcilerini tanır. Diğer işlemcileri de (Cx586,686 gibi) 486 olarak tanır. Ayrıca Windows-NT için ALPHA-21064, ALPHA-21164, INTEL-386, INTEL-486, INTEL-PENTIUM-586, INTEL-860, MIPS-R2000, MIPS-R3000, MIPS-R4000, PowerPC-601, PowerPC-603, PowerPC-604, PowerPC-620 işlemcilerini tanır.

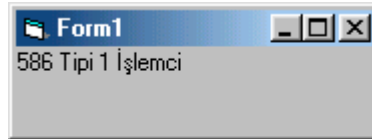
dwNumberOfProcessors:

Bilgisayardaki işlemci sayısını verir. Windows-95 birden fazla işlemciyi desteklemediği için her zaman 1 değeri döner, NT altında ise birden fazla CPU desteklendiği için işlemci sayısı bu parametre ile öğrenilir.

ÖRNEK: GetSystemInfo

```
Option Explicit
Private Declare Sub GetSystemInfo Lib "kernel32" (lpSystemInfo As
SYSTEM_INFO)
Private Type SYSTEM_INFO
    dwOemID As Long
    dwPageSize As Long
    lpMinimumApplicationAddress As Long 'Any
    lpMaximumApplicationAddress As Long 'Any
    dwActiveProcessorMask As Long
    dwNumberOfProcessors As Long
    dwProcessorType As Long
    dwAllocationGranularity As Long
    dwReserved As Long
End Type
Private Sub Form_Load()
    Show
    Cls
    Dim s As SYSTEM_INFO
```

```
GetSystemInfo s
Print s.dwProcessorType; " tipi " ; s.dwNumberOfProcessors ; "
islemci "
End Sub
```



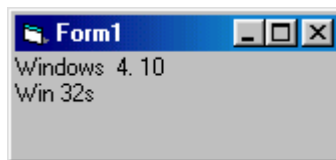
Windows versiyonunu ve ortamini öğrenmek

Declare Function GetVersion Lib "kernel32" () As Long

Bu fonksiyondan dönen deger long yani 8 byte'lik bir sayidir. Bu sayinin düşük kisimindeki iki byte Windows versiyonunu (4.0 ise 95, 4.1 ise 98), yüksek seviyeli kisimdeki en yüksek bit ise Windows ortamini verir. Bu bit 1 ise Win32s (95, 98), 0 ise Windows NT ortamidir.

ÖRNEK: GetVersion

```
Option Explicit
Private Declare Function GetVersion Lib "kernel32" () As Long
Private Sub Form_Load()
    Show
    Dim a
    a = GetVersion()
    Print "Windows " + Str(a And &HFF) + " . " + Str((a / &H100 And &HFF))
    If a And &H7FFFFFFF Then
        Print "Win 32s" Else
        Print "Windows NT" End
    If
End Sub
```



Klavye kod sayfasini öğrenmek

Microsoft Visual Basic 6.0

```
Declare Function GetKBCodePage Lib "User32" () As Integer
```

Bu API'den geri dönen deger Windows'a tanitilmiş klavye kod sayfasidir.

ÖRNEK: GetKBCodePage

```
Option Explicit
Private Declare Function GetKBCodePage Lib "User32" () As Integer

Private Sub Form_Load ()
Show
Print GetKBCodePage()
End Sub
```

Klavye tipini öğrenmek

```
Declare Function GetKeyboardType Lib "User32" (ByVal nTypeFlag As Integer) As Integer
```

nTypeFlag parametresine 0 verildiginde geri dönen deger klavye tip numarasi-dir. 2 verildiginde geri dönen deger klavyedeki fonksiyon tuslarinin sayisidir.

Klavye tip numaralari ve anlamlari söyledir:

1. XT tipi 83 tuslu,
2. Olivetti tipi 102
3. tuslu, AT tipi 84
4. tuslu, 101 veya
5. 102 tuslu, Nokia
6. 1050 tipi, Nokia
7. 9140 tipi, Japon tipi

ÖRNEK: GetKeyboardType

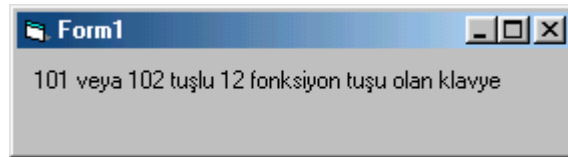
```
Option Explicit
Private Declare Function GetKeyboardType Lib "User32" (ByVal nTypeFlag As Integer) As Integer
Private Sub Form_Load()
Show
Dim a, str1 a =
GetKeyboardType(0) Select Case
a
Case 1: str1 = 'XT tipi 83 tuslu'
```



```

Case 2: str1 = 'Olivetti tipi 102 tuslu"
Case 3: str1 = 'AT tipi 84 tuslu"
Case 4: str1 = '101 veya 102 tuslu"
Case 5: str1 = 'Nokia 1050 tipi"
Case 6: str1 = 'Nokia 9140 tipi"
Case 7: str1 = 'Japon tipi"
Case Else: str1 = "Bilinmeyen bir tip" End
Select
Print str1 + Str(GetKeyboardType( 2 ) ) + " fonksiyon tusu olan
klavye"
End Sub

```



Modern Inkey\$

İlk iki kusak BASIC'ten bildiğimiz Inkey\$ fonksiyonu oldukça kullanışlı bir fonksiyondur ve bir işi yaparken kullanıcının bir tusa basıp basmadığını öğrenebiliyorduk. Mesela bir işlemi, kullanıcı bir tusa basıncaya kadar devam ettirmek için VB' de kullanabileceğiniz bir komut yoktur. Bu iş için kodunuzu bir kontrolün key veya mouse olaylarından birine yazmanız gerekecektir. Ancak GetInputState API'sini Inkey fonksiyonundan daha gelişmiş bir fonksiyon olarak kullanabiliriz. Çünkü bu API klavye haricinde mouse'un da tiklanıp tiklanmadığını bildirir.

```
Declare Function GetInputState Lib "User32" () As Integer
```

Fonksiyondan geri dönen değer 0 değilse klavye basılmış veya mouse ile kliklenmiştir. Ancak bu aktif olan program hangisi olursa olsun o olayı algılayacaktır. Ayrıca klavyedeki bütün tuşları algılar.

Aşağıdaki örnekte kullanıcı bir tusa veya mouse'a basıncaya kadar döngü içinde kalacaktır.

```

Private Declare Function GetInputState Lib "User32" () As Integer
Private Sub Timer1_Timer()
Dim i
Print "basladi"
While GetInputState () = 0
Wend
Print "ok"
End Sub

```

Windows dizinini öğrenmek

Declare Function GetWindowsDirectory Lib "Kernel32" Alias
"GetWindowsDirectoryA" (ByVal lpBuffer As String, ByVal nSize As Integer)
As Integer

Bu API ile WIN.COM dosyasının bulunduğu yolu öğrenebilirsiniz.

LpBuffer: Windows yolu bu değiskende geri döner. String olarak tanımlanmış ve içi herhangi bir karakterle doldurulmuş olması gerekir.

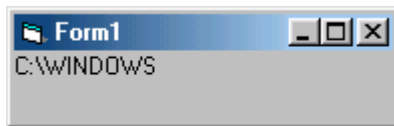
nSize: LpBuffer'in boyutudur.

Fonksiyondan geri dönen değer Windows yolunun uzunluğudur.

ÖRNEK: GetWindowsDirectory

```
Option Explicit
Private Declare Function GetWindowsDirectory Lib "Kernel32"
Alias
"GetWindowsDirectoryA" (ByVal lpBuffer As String, ByVal
nSize As
Integer) As Integer

Private Sub Form_Load()
    Show
    Dim WYol As String, uz
    WYol = String (145, Chr(32))
    uz = GetWindowsDirectory(WYol, Len(WYol))
    Print Left(WYol, uz)
End Sub
```



System dizinini öğrenmek

Declare Function GetSystemDirectory Lib "Kernel32" Alias
GetSystemDirectoryA"(ByVal lpBuffer As String, ByVal nSize As Integer)
As Integer

Yukarıdaki API'den tek farkı Windows yolunu değil Sistem dizininin yolunu verir. Kullanımı yukarıdaki ile aynıdır.

Windows Ayarları Hakkında Bilgi

GetSystemMetrics apisi Windows'un bir çok ayarıyla ilgili bilgiler verir.

```
Private Declare Function GetSystemMetrics Lib "user32" (ByVal  
nIndex As Long) As Long
```

nIndex parametresine aşağıdaki sayılardan biri verilerek o şey hakkında bilgi alınabilir.

- 0 Ekranın yatay genişliği
- 1 Ekranın dikey genişliği
- 2 Dikey kaydırma çubuğundaki okun boyutu
- 3 Yatay kaydırma çubuğundaki okun boyutu
- 4 Pencere başlığının yüksekliği
- 5 Boyutlandırılabilen pencerelerin yatay kenarlık kalınlığı
- 6 Boyutlandırılabilen pencerelerin dikey kenarlık kalınlığı
- 7 Diyalog kutularının yatay kenarlık kalınlığı
- 8 Diyalog kutularının dikey kenarlık kalınlığı
- 9 Yatay kaydırma çubuklarındaki kaydırma kutusunun yüksekliği
- 10 Yatay kaydırma çubuklarındaki kaydırma kutusunun genişliği
- 11 Standart ikon genişliği
- 12 Standart ikon yüksekliği
- 13 Standart kursor genişliği
- 14 Standart kursor yüksekliği
- 15 Menü yüksekliği
- 16 Ekranı kaplamış durumdaki pencerenin iç genişliği
- 17 Ekranı kaplamış durumdaki pencerenin iç yüksekliği
- 19 Bilgisayar bağlı bir farenin olup olmadığı
- 20 Dikey kaydırma çubuğundaki okun yüksekliği
- 21 Dikey kaydırma çubuğundaki okun genişliği
- 22 Windows'un debug modunda çalışıp çalışmadığı
- 23 Fare düğmelerinin yer değiştirilip değiştirilmediği
- 28 Pencerenin olabilecek minimum genişliği
- 29 Pencerenin olabilecek minimum yüksekliği
- 30 Başlık çubuğundaki resmin genişliği
- 31 Başlık çubuğundaki resmin yüksekliği
- 36 Çift tıklama için farenin yatay sapma genişliği
- 37 Çift tıklama için farenin yatay sapma yüksekliği
- 38 Masa üstü ikonlarının genişliği
- 39 Masa üstü ikonlarının yüksekliği

Microsoft Visual Basic 6.0

40 Popup menülerin sag tarafa dogru açilip açilmayacagi
41 Kalem yüklenmisse onun handle numarasi
42 Windows'un karakterler için iki byte (Japonca gibi) kullanip kullanmadigi
43 Farede bulunan düğme sayisi
51 Basliklarin genisligi
61 Ekranı kaplamis durumdaki pencerenin genisligi
63 Yüklü bir ag olup olmadigi
67 Windows açilis modu. 0 = normal, 1 = Güvenli kip, 2 = Ag destegiyle
Güvenli kip
71 Menülerdeki check isaretinin genisligi
72 Menülerdeki check isaretinin yüksekligi
73 Bilgisayarın çok yavas olup olmadigi
74 Arapça Windows

GetSystemMetrics apisi yukaridaki degerlerden biri ile çağrilip o özelligin degeri veya aktif olup olmadigi öğrenilebilir.

ÖRNEK: Windowsun bütün ayarlarini bildirecek bir program yapalim. Örnegimiz için formunuza bir ListBox yerlestirin.

```
Option Explicit
Private Declare Function GetSystemMetrics Lib "user32" (ByVal
nIndex As Long) As Long
Sub Form_Load ()
List1.AddItem "0 Ekranın yatay genisligi=" &
GetSystemMetrics(0)
List1.AddItem "1 Ekranın dikey genisligi=" &
GetSystemMetrics(1)
List1.AddItem "2 Dikey kaydırma çubugundaki okun boyutu=" &
GetSystemMetrics(2)
List1.AddItem "3 Yatay kaydırma çubugundaki okun boyutu=" &
GetSystemMetrics(3)
List1.AddItem "4 Pencere basliginin yüksekligi=" &
GetSystemMetrics(4)
List1.AddItem "5 Boyutlandırılabilen pencerelerin yatay
kenarlık kalınlığı=" & GetSystemMetrics(5)
List1.AddItem "6 Boyutlandırılabilen pencerelerin dikey
kenarlık kalınlığı=" & GetSystemMetrics(6)
List1.AddItem "7 Dialog kutularının yatay kenarlık kalınlığı="
& GetSystemMetrics(7)
List1.AddItem "8 Dialog kutularının dikey kenarlık kalınlığı="
& GetSystemMetrics(8)
List1.AddItem "9 Yatay kaydırma çubuklarındaki kaydırma
kutusunun yüksekligi=" & GetSystemMetrics(9)
List1.AddItem "10 Yatay kaydırma çubuklarındaki kaydırma
kutusunun genisligi=" & GetSystemMetrics(10)
```

```
List1.AddItem "11 Standart ikon genisligi=" &
GetSystemMetrics(11)
List1.AddItem "12 Standart ikon yüksekligi=" &
GetSystemMetrics(12)
List1.AddItem "13 Standart kursor genisligi=" &
GetSystemMetrics(13)
List1.AddItem "14 Standart kursor yüksekligi=" &
GetSystemMetrics(14)
List1.AddItem "15 Menü yüksekligi=" & GetSystemMetrics(15)
List1.AddItem "16 Ekrani kaplamis durumdaki pencerenin iç
genisligi=" & GetSystemMetrics(16)
List1.AddItem "17 Ekrani kaplamis durumdaki pencerenin iç
yüksekligi=" & GetSystemMetrics(17)
If GetSystemMetrics(19) Then
List1.AddItem "19 Bilgisayar bagli bir fare var"
Else: List1.AddItem "19 Bilgisayar bagli bir fare yok"
End If
List1.AddItem "20 Dikey kaydirma çubugundaki okun yüksekligi^"
& GetSystemMetrics(20)
List1.AddItem "21 Dikey kaydirma çubugundaki okun genisligi=" &
GetSystemMetrics(21)
If GetSystemMetrics(22) Then
List1.AddItem "22 Windows debug modunda çalışiyor"
Else: List1.AddItem "22 Windowsun debug modunda çalışmıyor"
End If
If GetSystemMetrics(23) Then
List1.AddItem "23 Fare düğmeleri yer degistirilmis"
Else: List1.AddItem "23 Fare düğmeleri yer degistirilmemis"
End If
List1.AddItem "28 Pencerenin olabilecek minimum genisligi=" &
GetSystemMetrics(28)
List1.AddItem "29 Pencerenin olabilecek minimum yüksekligi=" &
GetSystemMetrics(29)
List1.AddItem "30 Baslik çubugundaki resmin genisligi=" &
GetSystemMetrics(30)
List1.AddItem "31 Baslik çubugundaki resmin yükseligi=" &
GetSystemMetrics(31)
List1.AddItem "36 Çift tiklama için farenin yatay sapma
genisligi=" & GetSystemMetrics(36)
List1.AddItem "37 Çift tiklama için farenin yatay sapma
yüksekligi=" & GetSystemMetrics(37)
List1.AddItem "38 Masa üstü ikonlariinin genisligi=" &
GetSystemMetrics(38)
List1.AddItem "39 Masa üstü ikonlariinin yüksekligi=" &
GetSystemMetrics(39)
If GetSystemMetrics(40) Then
List1.AddItem "40 Popup menüleri sag tarafa dogru açiliyor"
Else
List1.AddItem "40 Popup menüleri sag tarafa dogru açilmiyor"
```

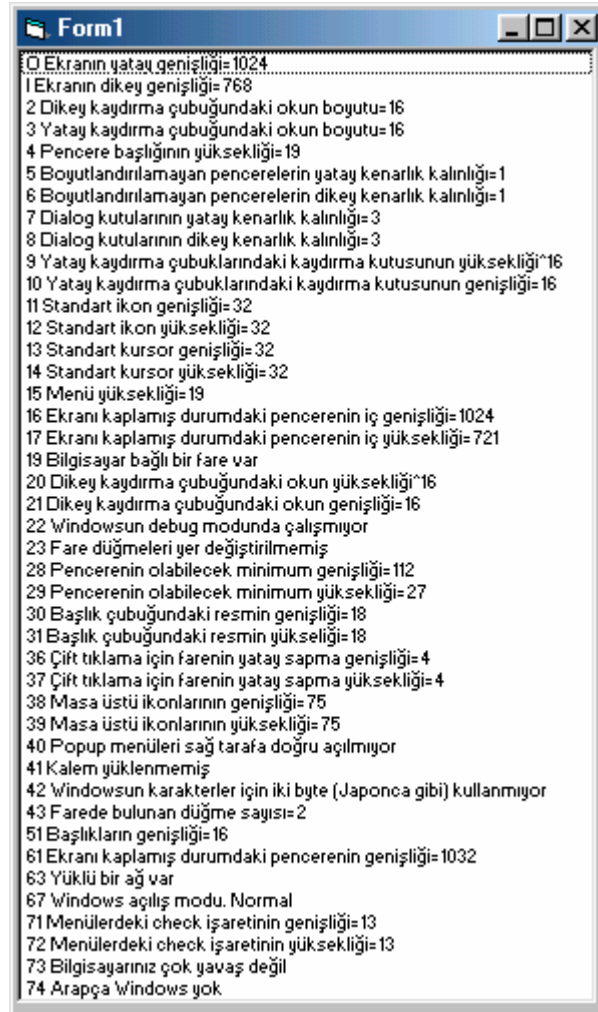
Microsoft Visual Basic 6.0

```
End If
If GetSystemMetrics(41) Then
List1.AddItem "41 Kalem yüklenmiş onun handle numarası==" &
GetSystemMetrics(41)
Else
List1.AddItem "41 Kalem yüklenmemiş"
End If
If GetSystemMetrics(42) Then
List1.AddItem "42 Windowsun karakterler için iki byte (Japonca
gibi) kullanıyor"
Else
List1.AddItem "42 Windowsun karakterler için iki byte (Japonca
gibi) kullanmıyor"
End If
List1.AddItem "43 Farede bulunan düğme sayısı=" &
GetSystemMetrics(43)
List1.AddItem "51 Başlıkların genişliği=" &
GetSystemMetrics(51)
List1.AddItem "61 Ekranı kaplamış durumdaki pencerenin
genişliği=" & GetSystemMetrics(61)
If GetSystemMetrics(63) Then
List1.AddItem "63 Yüklü bir ağ var"
Else
List1.AddItem "63 Yüklü bir ağ yok"
End If

Select Case GetSystemMetrics(67)
Case 0: List1.AddItem "67 Windows açılış modu. Normal"
Case 1: List1.AddItem "67 Windows açılış modu. Güvenli kip"
Case 2: List1.AddItem "67 Windows açılış modu. Ağ desteğiyle
Güvenli kip"
End Select

List1.AddItem "71 Menülerdeki check işaretinin genişliği=" &
GetSystemMetrics(71)
List1.AddItem "72 Menülerdeki check işaretinin yüksekliği=" &
GetSystemMetrics(72)
If GetSystemMetrics(73) Then
List1.AddItem "73 Bilgisayarınız çok yavaş"
Else
List1.AddItem "73 Bilgisayarınız çok yavaş değil"
End If
If GetSystemMetrics(74) Then
List1.AddItem "74 Arapça Windows var"
Else
List1.AddItem "74 Arapça Windows yok"
End If

End Sub
```



Disk Sürücüler Hakkında Bilgi Veren Apiler

Systemdeki disk sürücülerini öğrenmek

**Declare Function GetDriveType Lib "Kernel32" (ByVal
lpRootPathName As String) As Integer**

Bu fonksiyonla sistemdeki sürücü tiplerini ve dolaylı olarak ta sistemdeki mevcut sürücülerini öğrenmek mümkündür. LpRootPathName parametresi ile verilen sürücünün tipini geriye bir sayı ile gönderir. Bu fonksiyondan dönen değerler ve anlamlarıda şöyledir: 0

Microsoft Visual Basic 6.0

0:Yolu verilen sürücü sistemde yok,
2 :Yolu verilen sürücü bir disket sürücü (veya taşınabilir disk),
3 :Yolu verilen sürücü bir harddisk sürücü (veya taşınamaz disk),
4 :Yolu verilen sürücü bir uzak sürücü (Network sürücüsü gibi),
5 :Yolu verilen sürücü bir CDROM sürücü
6 :Yolu verilen sürücü bir RAM disk (Windows 98 başlangıç disketinin oluşturduğu sürücü gibi)

ÖRNEK: Örnek olarak sistemdeki bütün sürücüleri ve tiplerini gösterecek bir program yazalım. Örneğimizde sürücü tipini öğrenmek için bu API'yi kullanacağız. Ayrıca sistemde kaç sürücü olduğunu da bu API aracılığı ile dolaylı olarak bulabiliriz. A harfinden Z harfine kadar olan bütün harfleri bu API'ye parametre olarak bir döngü içinde verirsek, verdiğimiz harfe uyan sürücüler hakkındaki bilgi geri dönecek, o sürücü yoksa da 0 değeri dönecektir.

ÖRNEK : GetDriveType

Option Explicit

```
Private Declare Function GetDriveType Lib "Kernel32" Alias  
"GetDriveTypeA" (ByVal IpRootPathName As String) As Integer
```

```
Private Sub Form_Load()
```

```
Show
```

```
Dim i As Integer, a As Integer
```

```
Dim s2, s3, s4, s5, s6
```

```
For i = 0 To 27 'Bütün harfleri dene
```

```
a = GetDriveType (Chr(65 + i) + " : \ " ) 'rakami harfe çevir
```

```
Select Case a
```

```
Case 0: 'i numarali sürücü yok
```

```
Case 2: 'disket sürücü
```

```
s2 = s2 + " " + Chr(65 + i) + ": " Case 3: 'hard disk sürücü
```

```
s3 = s3 + " " + Chr(65 + i) + ": " Case 4: 'Uzak sürücü network  
gibi
```

```
s4 = s4 + " " + Chr(65 + i) + ": " Case 5: 'CDROM sürücü
```

```
s5 = s5 + " " + Chr(65 + i) + ": " Case 6: 'RAM disk
```

```
s6 = s6 + " " + Chr(65 + i) + ": "
```

```
End Select
```

```
Next
```

```
Print "Disket Sürücüler:";
```

```
If s2 = "" Then Print " Yok " Else Print s2
```

```
Print "Harddisk Sürücüler:"; If s3 = "" Then
```

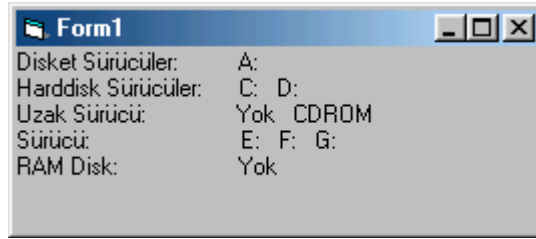
```
Print " Yok " Else Print s3 Print "Uzak sürücü:";
```

```
If s4 = "" Then Print " Yok " Else Print s4 Print "CDROM sürücü:";
```

```
If s5 = "" Then Print " Yok " Else Print s5 Print "RAM disk:";
```

```
If s6 = "" Then Print " Yok " Else Print s6
```


End Sub



Volume Bilgisini Öğrenme

GetVolumeInformation Api'si aracılığı ile bir sürücüdeki volümün adını, seri numarasını, dosya adında desteklediği harf sayısını ve sıkıştırılmış olup olmadığını öğrenmek mümkündür.

```
Declare Function GetVolumeInformation Lib "kernel32" Alias
"GetVolumeInformationA" (ByVal IpRootPathName As String, ByVal
IpVolumeNameBuffer As String, ByVal nVolumeNameSize As Long,
IpVolumeSerialNumber As Long, IpMaximumComponentLength As Long,
IpFileSystemFlags As Long, ByVal IpFileSystemNameBuffer As String,
ByVal nFileSystemNameSize As Long) As Long
```

IpRootPathName:

Volüm bilgisi öğrenilecek sürücünün ana dizinini gösteren string. Örneğin "c:\\" veya "d:\".

IpVolumeNameBuffer:

Volüm adı bu parametreye verilen değişkenle geri döner. Bu parametreye en az 14 harf alabilecek bir string verilmelidir.

nVolumeNameSize:

IpVolumeNameBuffer parametresine verilen değişkenin uzunluğu bu parametreye verilir.

IpVolumeSerialNumber:

Volüm seri numarası bu parametreye verilen Long türündeki değişkenle geri döner. Sonuç HEX fonksiyonu ile hexadesimale çevrilirse seri numara öğrenilmiş olur.

IpMaximumComponentLength:

Microsoft Visual Basic 6.0

Volümün, dosya ve dizin isimlerinde desteklediği maximum harf sayısı bu parametreye verilen değişkenle döner. Örneğin dos için bu değer 11, Windows 9x için 255'tir.

IpFileSystemFlags:

Volüm hakkındaki bazı bilgiler bu parametreye verilen değişkenle geri döner. Geri dönen değer aşağıdaki sayılarla And işlemine tabi tutularak ilgili özelliği destekleyip desteklemediği öğrenilebilir.

&H1: Dosya sistemi büyük küçük harf ayırımı destekler. Örneğin "abc" ile "ABC" farklı dosyalardır.

&H10: Dosya sistemi dosya tabanlı sıkıstırmayı destekliyor.

&H8000: Dosya sistemi volume tabanlı sıkıstırma kullanıyor. (DoubleSpace veya DrvSpace)

IpFile System Name Buffer:

Dosya sisteminin ismi (Fat, Fat32, Ntfs, Cdfs vb) bu parametreye verilen değişkenle döner.

nFileNameSize:

IpFileNameBuffer parametresi için kullanılan değişkenin uzunluğu. Eğer fonksiyonda geriye 0 değeri dönerse işlem başarısızdır.

ÖRNEK: Örnek olarak sistemden seçilen sürücü hakkında bilgi veren bir program yazalım. Örneğimiz için formunuza bir DriveListBox yerleştirin. Kullanıcı bu listeden bir sürücü seçtiğinde ona ait bilgiyi forma yazdıralım.

'ÖRNEK: GetVolumeInformation

Option Explicit

```
Private Declare Function GetVolumeInformation Lib "kernel32" Alias  
"GetVolumeInformationA" (ByVal IpRootPathName As String  
ByVal IpVolumeNameBuffer As String, ByVal nVolumeNameSize As  
Long, IpVolumeSerialNumber As Long, IpMaximumComponentLength As  
Long, IpFileSystemFlags As Long, ByVal IpFileNameBuffer  
As String, ByVal nFileNameSize As Long) As Long
```

```
Private Sub Form_Load()  
Show  
Drive1_Change End Sub
```

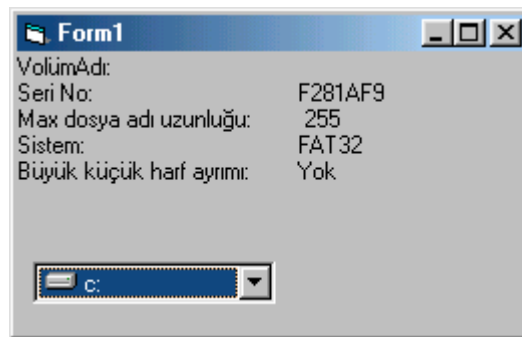
```
Private Sub Drive1_Change()  
Dim r As Long  
Dim pos As Integer  
Dim VolumeSN As Long  
Dim MaxFNLen As Long  
Dim FsName As String
```

```

Dim maxuz As Long
Dim flags As Long
Dim VolumeName As String
Dim surucu As String
VolumeName = Space(14)
FsName = Space(32)
surucu = Left(Drive1.Drive, 2) & "\"
r = GetVolumeInformation(surucu, VolumeName, Len(VolumeName),
VolumeSN&, maxuz, flags, FsName, Len(FsName))
If r = 0 Then Exit Sub
pos = InStr(VolumeName, Chr(0))
If pos Then VolumeName = Left(VolumeName, pos - 1)
Print "VolümAdı:", , VolumeName
Print "Seri No:", , Hex(VolumeSN)
Print "Max dosya adı uzunluğu:", maxuz

pos = InStr(FsName, Chr(0))
If pos Then FsName = Left(FsName, pos - 1)
Print "Sistem:", , FsName
If flags And &H8000 Then Print "Sikistirma:", , "DblSpace veya
DrvSpace"
If flags And &H10 Then Print "Sikistirma:", , "Dosya tabanlı"
Print "Büyük küçük harf ayrımı:",
If flags And 1 Then Print "Var" Else Print "Yok"
Print " ----- "
End Sub

```



Sürücü Boyutu

Windows 95 OSR2 versiyonundan sonra geçerli olan GetDiskFreeSpaceEx Api'si aracılığı ile sistemdeki bir sürücüde bulunan toplam alan, bos alan ve bir ag söz konusu ise aktif kullanıcının kullanabilecegi bos alan öğrenilebilir.

Microsoft Visual Basic 6.0

Bu api normalde 64 bitlik tam sayıları kullanır. Çünkü 32 bitlik bir sayı ile en çok 4GB ifade edilebilmektedir. VB'de 64 bitlik sayıları işleyecek değişkenler olmadığı için de iki tane 32 bit (long) değişkenden oluşan yeni bir tip tanımlayacağız.

```
Private Type uzunsayi
    l As Long 'Düşük kısım
    h As Long 'Yüksek
kısım
End Type
```

Bu tipi tanımladıktan sonra Api içerisindeki 64 bitlik tamsayıları bu tipten tanımlayabiliriz. 64 bitlik sonucu bulmak için ise, yüksek seviyeli kısım 2^{32} ile çarptıktan sonra düşük seviyeli kısım ile toplamamız gerekir.

```
Private Declare Function GetDiskFreeSpaceEx Lib "kernel32" Alias
"GetDiskFreeSpaceExA" (ByVal IpRootPathName As String,
IpFreeBytesAvailableToCaller As uzunsayi, IpTotalNumberOfBytes As
uzunsayi, IpTotalNumberOfFreeBytes As uzunsayi) As Long
IpRootPathName:
Boyutu öğrenilecek sürücünün ana dizini. Örneğin
If C:\ " IpFreeBytesAvailableToCaller:
```

Bu parametreye verilen değişkenle o anki kullanıcının kullanabileceği alandaki boş miktar byt olarak bildirilir. Network ortamlarında kullanıcıların disk alanları sınırlanabildiği için diskteki boş alanla kullanıcının kullanabileceği boş alan aynı olmayabilmektedir. Bu parametre o anki kullanıcının boş alanını gösterir.

IpTotalNumberOfBytes:

Diskin toplam kapasitesi bu parametreye verilen değişkenle döner.

IpTotalNumberOfFreeBytes:

Diskteki boş alan bu parametreye verilen değişkenle döner.

ÖRNEK: Yukarıda yaptığımız örneği biraz daha geliştirerek seçilen sürücü boyutları hakkında da bilgi verilmesini sağlayalım.

```
ÖRNEK: GetDiskFreeSpaceEx
Option Explicit
```

```
Option Explicit
Private Type uzunsayi
```

```
l As Long
h As Long
End Type
```

```
Public Declare Function GetDiskFreeSpace Lib "kernel32" Alias
"GetDiskFreeSpaceA" (ByVal lpRootPathName As String,
lpSectorsPerCluster As Long, lpBytesPerSector As Long,
lpNumberOfFreeClusters As Long, lpTotalNumberOfClusters As
Long) As Long
```

```
Private Declare Function GetVolumeInformation Lib "kernel32"
Alias "GetVolumeInformationA" (ByVal lpRootPathName As String,
ByVal lpVolumeNameBuffer As String, ByVal nVolumeNameSize As
Long, lpVolumeSerialNumber As Long, lpMaximumComponentLength As
Long, lpFileSystemFlags As Long, ByVal lpFileSystemNameBuffer
As String, ByVal nFileSystemNameSize As Long) As Long
```

```
Private Sub Drivel_Change()
```

```
Cls
Dim r As Long
Dim pos As Integer
Dim VolumeSN As Long
Dim MaxFNLen As Long
Dim FsName As String
Dim maxuz As Long
Dim flags As Long
Dim VolumeName As String
Dim surucu As String
VolumeName = Space(14)
FsName = Space(32)
surucu = Left(Drive1.Drive, 2) & "\"
Dim afb As uzunsayi, tb As uzunsayi, fb As uzunsayi
Call GetDiskFreeSpaceEx(surucu, afb, tb, fb)
Print "Toplam kapasite MB:", , (tb.h * 2 ^ 32 + tb.l) / 1024 /
1024
Print "Toplam bos alan MB:", , (fb.h * 2 ^ 32 + fb.l) / 1024 /
1024
Print "Aktif kullanıcı için bos alan MB:", (afb.h * 2 ^ 32 +
afb.l) / 1024 / 1024
r = GetVolumeInformation(sürücü, VolumeName, Len(VolumeName),
VolumeSN&, maxuz, flags, FsName, Len(FsName))
If r = 0 Then Exit Sub
pos = InStr(VolumeName, Chr(0))
If pos Then VolumeName = Left(VolumeName, pos - 1)
Print "Volüm Adı:", , VolumeName
Print "Seri No:", , Hex(VolumeSN)
Print "Max dosya adı uzunluğu:", maxuz
pos = InStr(FsName, Chr(0))
```

Microsoft Visual Basic 6.0

```
If pos Then FsName = Left(FsName, pos - 1)
Print "Sistem:", , FsName
If flags And &H8000 Then Print "Sikistirma:", , "DblSpace veya
DrvSpace"
If flags And &H10 Then Print "Sikistirma:", , "Dosya tabanlı"
Print "Büyük küçük harf ayrımı:",
If flags And 1 Then Print "Var"
Else: Print "Yok"
Print " "

End Sub

Private Sub Form_Load()
Show
Drive1_Change
End Sub
```

Windows Altında Çalışan Bütün Formlara Hükmetmek

VB'nin sunduğu kendi imkanlarıyla Windows altında çalışan diğer programların formlarını değiştirmek neredeyse imkansızdır. App.Activate ve SendKeys yöntemleriyle bu işler ancak ilkel olarak yapılabilir. Halbuki şimdi vereceğimiz A-Pillerle o anda Windows altında çalışan bütün formlar üzerinde kendi formu-nuzmuş gibi rahatlıkla işlem yapabilirsiniz. Mesela çalışması gereken ancak görüntüsü hosunuza gitmeyen bir formu istediğiniz anda gizleyebilir ve tekrar gösterebilirsiniz, veya minimize düğmesi unutulmuş(ı) bir formu minimize hale getirebilirsiniz. Hayal gücünüzle ve programcılık kabiliyetinizle kendinize yeni bir Task Manager bile yapabilirsiniz.

Windows altında formlarla ilgili API leri kullanabilmek için gerekli olan ilk şey o formun handle numarasıdır (hWnd). Bu numarayı öğrendikten sonra o form artık sizin emrinize amadedir. Kendi formunuzun handle numarasına formadi(hWnd) özelliğiyle ulaşabileceğinizi biliyorsunuz. Ya sizin programınıza ait olmayan bir formun handle numarasını? İşte bu numarayı öğrenmek için kullanacağımız API GetWindow .

Çalışan formların handle numaralarını bulmak

```
Declare Function GetWindow Lib "User32" (ByVal hWnd As Integer,
ByVal wCmd As Integer) As Integer
```

Bu API kullanılarak sistemde o anda çalışan formların (gizli dahi olsa) handle numarasını öğrenmek mümkündür. Windows çalışan bütün formları

bir listede tutar. Bu API ile o listedeki formlarin handle numaralari öğrenilir.

HWnd: Referans olarak alınacak orijinal formun handle numarasidir.

WCmd: O verilerek en üstteki formun handle numarası, 2 verilerek bir sonraki formun handle numarası öğrenilmek istendigi bildirilir.

Fonksiyondan geri dönen deger O ise ya listede baska form kalmamistir yada hatali komut girilmistir. Döner deger O degilse bu deger istenen formun handle numarasidir.

O halde biz bu API'yi kullanarak o anda çalışan bütün formlarin handle numaralarini (hWnd) öğrenebiliriz.

```
`Ilk formun handle numarasini al
Aktif_pencere_hwnd = GetWindow (Form1.hWnd, 0)
While Aktif_pencere_hwnd <> 0
`Sonrakinin handle numarasini al
Aktif_pencere_hwnd = GetWindow (Aktif_pencere_hwnd, 2) `döner
deger O ise liste bitti.
Wend
```

Böyle bir yapıyla bütün formlarin handle numaralarini öğrenmek mümkündür. Handle numarasini ne yapacagiz diye bir soru aklınıza gelebilir. Daha önce de bahsettigimiz gibi Windows kontrolleri o kontrole verdigi handle numarası ile tanir. Bir çok API' de bu numara ile isleme girer. Simdi bu numarayı kullanarak formun basligini bulacak ve degistirecek API'leri verelim.

Çalışan formlarin basliklarini degistirmek

```
Declare Sub SetWindowText Lib "User32" Alias "SetWindowTextA"
(ByVal hWnd As Integer, ByVal IpString As String)
```

GetWindows API'siyle handle numarasini öğrendiginiz formun basligini **SetWindowText** API'si ile degistirebilirsiniz.

hWnd: Basligi degistirilecek formun handle numarası IpString: Yeni baslik metni.

Çalışan formlarin basliklarini bulmak

VB' deki App.Activate metodu ile pencere basligini verdiginiz bir programi aktif hale getirebilecegimizi biliyorsunuz. Ancak bu metotta pencere basligini sizin vermeniz gerekir ve verdiginiz baslikla aktif hale getirmek istediginiz basligin tamamen ayni olmasi gerekir. Ayrica bu metotla aktif hale getireceginiz program o anda çalışiyor olmalıdır. Bu metoddaki eksiklikleri su iki API ile gidermek mümkündür.

Microsoft Visual Basic 6.0

```
Declare Function GetWindowText Lib "User32" Alias "GetWindowTextA"  
(ByVal hWnd As Integer, ByVal lpString As String, ByVal aint As  
Integer) As Integer
```

hWnd: Basligi öğrenilecek formun handle numarası

lpString: Basligin geri dönecegi degisken.

Aint: lpString degiskenine kopyalanacak karakter sayisi. Bu parametreye basligin tam uzunlugunun bir fazlasi verilerek basligin tamamının lpString degiskenine kopyalanmasini saglamalısınız. Basligin tam olarak uzunlugunu ise GetWindowTextLength API'siyle öğrenebilirsiniz.

Fonksiyondan geri dönen deger O ise ya fonksiyona girilen handle numarali bir form yoktur yada formun basligi bostur. Aksi takdirde dönen deger kopyalanan karakter sayisini verir, yani aint parametresiyle girdiginiz degerin bir eksigi.

```
Declare Function GetWindowTextLength Lib "User32" Alias  
"GetWindowTextLengthA" (ByVal hWnd As Integer) As Integer
```

Bu API ile de handle numarasini verdiginiz formun basliginin uzunlugunu öğrenebilirsiniz. Yukarida da bahsettigimiz gibi buradan dönen degerin bir fazlasini GetWindowText API'sinin aint parametresine girmeniz gerekir. Degerin bir fazlasini girmemizin sebebi ise String degiskenlerin sonunda ASCII kodu 0 olan karakterin bulunmasi mecburiyetidir. Bu fonksiyondan geri dönen deger 0 ise handle numarası verilen form sistemde mevcut degildir.

ÖRNEK: Simdi bu üç API'yi kullanarak sistemdeki bütün formlarin basliklerini bir ListBox kontrolüne ekleyecek kodu yazalım.

Bu üç API'yi kullanarak Task Manager'in (Görev Yöneticisi'nin) daha gelismisini yapabilirsiniz. Formunuzun General-Declarations kismina yukaridaki üç API'nin tanımlarini yerlestirin. Ve form üzerine bir liste kutusu yerlestirin.

ÖRNEK : GetWindowText, GetWindowTextLength, GetWindow
Option Explicit

```
Private Declare Function GetWindowTextLength Lib "User32" Alias  
"GetWindowTextLengthA" (ByVal hWnd As Integer) As Integer
```

```
Private Declare Function GetWindowText Lib "User32" Alias  
"GetWindowTextA" (ByVal hWnd As Integer, ByVal lpString As
```



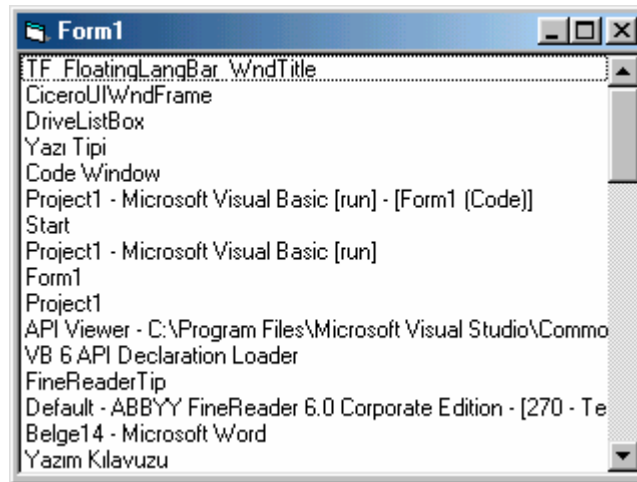
```
String, ByVal aint As Integer) As Integer

Private Declare Function GetWindow Lib "User32" (ByVal hWnd As
Integer, ByVal wCmd As Integer) As Integer

Private Sub Form_Load()
Show
Dim Aktif_pencere_hwnd, uz As Integer, baslik As String 'Ilk formun
handle numarasini al Aktif_pencere_hwnd = GetWindow(Form1.hwnd, 0)
While Aktif_pencere_hwnd <> 0
uz = GetWindowTextLength(Aktif_pencere_hwnd)
baslik = Space(uz + 1)
uz = GetWindowText (Aktif_pencere_hwnd, baslik, uz + 1)
If uz > 0 Then 'islem tamam ise
List1.AddItem baslik 'Ekrana; yaz
End If

'Sonrakinin handle numarasini al
Aktif_pencere_hwnd = GetWindow(Aktif_pencere_hwnd, 2)
'dönen deger 0 ise liste bitti.
Wend

End Sub
```



Bu kodla o anda sistemde çalışan bütün formların başlıklarını bir listeye yazdırılmaktadır. Listenin DbClick olayına App.Activate List1.Text satırını yazarak ilgili formu aktif hale de getirebilirsiniz. Ancak her form aktif hale getirilemediği (gizli formlar gibi) için bundan önce bir hata yakalayıcı da kullanmanız gerekir.

```
On Local Error Resume Next
App.Activate baslik
```

Kullanıcının listeden seçtiği bir formu aktif hale getirmek için de Liste kutusunun Doubleclick olayına su kodu yazın

```
Private Sub List1_DblClick ()  
On Local Error Resume Next  
AppActivate List1.Text  
SendKeys "% " 'Alt boşluk tusu ile kontrol kutusunu aç  
SendKeys "{ENTER}" 'Önceki boyut seçenegi  
End Sub
```

App.Activate ile aktif hale getirilen formun pencere seklinde bir değişiklik olmaz, yani form minimize ise aktif hale gelir ancak yine minimize konumundadır. Formu ekranda göstermek için aktif hale getirilen uygulamaya Alt-Bosluk ve Enter tuslarını göndererek "Önceki Boyut" seçeneğinin seçilmesini ve formun ekranda gösterilmesini sağlıyoruz.

Görüldüğü gibi App.Activate metoduyla bu işi yapmak oldukça ilkel ve yavaş. Tahmin ettiğiniz gibi bu işi de yapacak bir API var. ShowWindow.

Çalışan formların durumunu değiştirmek

```
Declare Function ShowWindow Lib "User32" (ByVal hWnd As Integer,  
ByVal nCmdShow As Integer) As Integer
```

Bu API ile hWnd numarası verdiğiniz formun nCmdShow parametresi ile belirlenen özelliklerini değiştirebilirsiniz.

NCmdShow:

0: Formu gizle (Visible = False)
5: Formu Göster (Visible = True)
9: Önceki Boyut
2: Minimize
3: Maximize

Bu parametrenin diğer değerlerini help dosyasında bulabilirsiniz. ShowWindow AP'si formun durumunu değiştirirken, formun durumu hakkında bilgi «verecek A-Piller de mevcuttur.

Form simge durumunda mı?

```
Declare Function IsIconic Lib "User32" (ByVal hWnd As Integer) As  
Integer
```

Handle numarasini verdiginiz form minimize ise geri dönen deger sifirdan farklıdır, aksi takdirde sifirdır.

Form ekranı kaplamış mı?

```
Declare Function IsZoomed Lib "User32" (ByVal hWnd As Integer) As Integer
```

Handle numarasini verdiginiz form maximize ise geri dönen deger sifirdan farklıdır, aksi takdirde sifirdır.

Form gizli mi?

```
Declare Function IsWindowVisible Lib "User32" (ByVal hWnd As Integer) As Integer
```

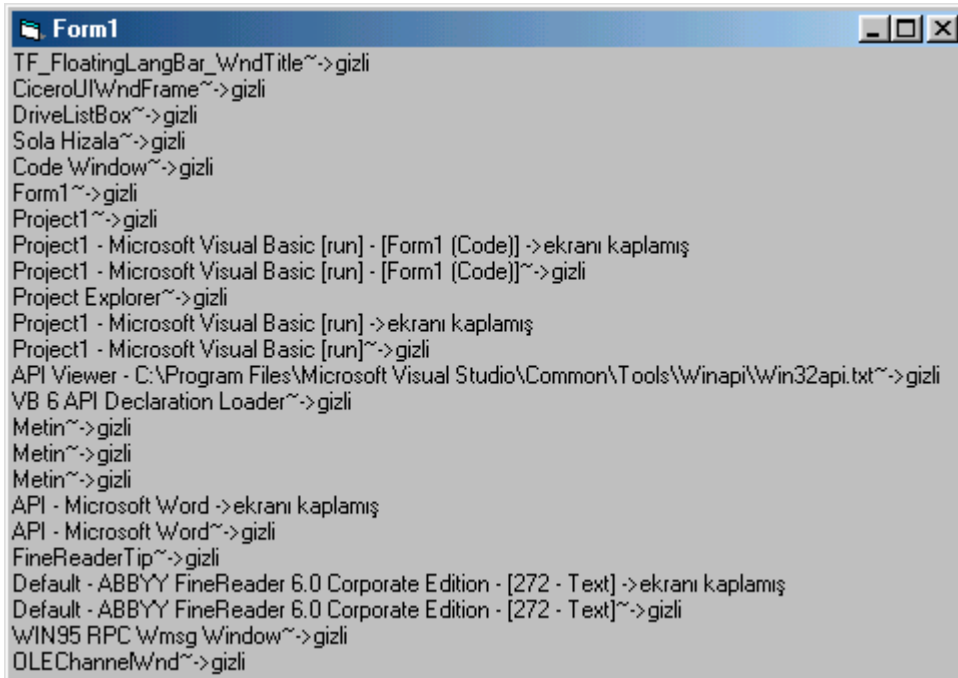
Handle numarasini verdiginiz form gizli ise geri dönen deger sıfır, aksi takdirde sifirdan farklıdır.

ÖRNEK: Yukarıdaki bir örnekte, Windows altında çalışan bütün uygulamaları listeleyecek bir program yazmistik. Şimdi aynı mantıkla bütün formların handle numarasını öğrendikten sonra onları durumları hakkında (simge durumu! gizli vb) bilgi verecek bir program yazalım. Örneğimiz için bir Timer yerleştirin ve Interval özelliğine 1000 verin.

```
Public Declare Function IsIconic Lib "user32" (ByVal hwnd As Long) As Long
Public Declare Function IsZoomed Lib "user32" (ByVal hwnd As Long) As Long
Public Declare Function IsWindowVisible Lib "user32" (ByVal hwnd As Long) As Long
Public Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" (ByVal hwnd As Long, ByVal lpString As String, ByVal cch As Long) As Long
Public Declare Function GetWindowTextLength Lib "user32" Alias "GetWindowTextLengthA" (ByVal hwnd As Long) As Long
Public Declare Function GetWindow Lib "user32" (ByVal hwnd As Long, ByVal wCmd As Long) As Long
Option Explicit
Private Sub Timer1_Timer()
    Cls
    Dim Aktif_pencere_hwnd, uz As Integer, baslik As String
    'İlk formun handle numarasını al
    Aktif_pencere_hwnd = GetWindow(Form1.hwnd, 0)
    While Aktif_pencere_hwnd <> 0
        'Print Aktifpencere.hwnd
```

Microsoft Visual Basic 6.0

```
uz = GetWindowTextLength(Aktif_pencere_hwnd)
baslik = Space(uz + 1)
uz = GetWindowText(Aktif_pencere_hwnd, baslik, uz + 1)
If uz > 0 Then
baslik = Left(baslik, uz)
If IsIconic(Aktif_pencere_hwnd) Then
Print baslik & "->simge durumunda"
End If
If IsZoomed(Aktif_pencere_hwnd) Then
Print baslik & " ->ekrani kaplamis"
End If
If Not IsWindowVisible(Aktif_pencere_hwnd) Then
Print baslik & "~>gizli"
End If
End If
Aktif_pencere_hwnd = GetWindow(Aktif_pencere_hwnd, 2)
'döneren deger 0 ise liste bitti.
Wend
End Sub
```



Masa Üstünün handle numarası

```
Declare Function GetDesktopWindow Lib "User32" () As Integer
```

Herhangi bir giriş parametresi olmayan bu API masa üstünün handle numarasını geri gönderir. Masa üstünde işlem yapmak istediğinizde (örneğin ekran görüntüsünü almak gibi) bu API ile masa üstüne ait handle numarasını öğrenebilirsiniz.

Formu en üstte tutmak ve gizlemek

```
Declare Sub SetWindowPos Lib "User32" (ByVal hWnd As Integer,
ByVal hWndInsertAfter As Integer, ByVal x As Integer, ByVal y As
Integer, ByVal ex As Integer, ByVal cy As Integer, ByVal wFlags As
Integer)
```

Handle numarası verilen formun özelliklerini değiştirmek için kullanılır.

hWnd: Üzerinde işlem yapılacak formun handle numarası.

hWndInsertAfter: Form üzerinde yapılacak sıralama işlemi bu parametreye verilecek aşağıdaki değerlerle belirlenir.

- O:** Form diğer formların önüne alınır
- 1:** Form diğer formların arkasına alınır.
- 1:** Form sürekli olarak diğer formların önünde kalır.
- 2:** Form u n en üstte kalma özelliği kaldırılır.
- x,y:** Formun yeni koordinatları
- cx,cy:** Formun yeni boyutları
- wFlags:** Bu parametreye verilebilecek değerler şunlardır.
- &H01:** Formun koordinatları değiştirilmez. Yani x ve y parametreleri göz ardı edilir.
- &H02:** Formun boyutları değiştirilmez. Yani cx ve ey parametreleri göz ardı edilir.
- &H80:** Form gizlenir.
- &H40:** Form gösterilir.
- &H10:** Form aktif hale getirilmez.

Örneğin form1 formunu en üstte tutmak için:

```
SetWindowPos form1.hWnd, -1, 0, 0, 0,0, &H10 Or &H40 Or
&H1 Or &H2
```

En üstte kalma özelliğini kaldırmak için de:

```
SetWindowPos form1.hWnd, -2, 0, 0, 0, 0, &H10 Or &H40 Or
&H1 Or &H2
```

Bir formu gizlemek için:

```
SetWindowPos Form1.hWnd, 0, 0, 0, 100, 100, &H80 Or &H1 Or
```

81H2 göstermek için ise:

```
SetWindowPos Form1.hWnd, 0, 0, 0, 100, 100, &H40 Or &H1 Or
&H2
```

Bu API'yi kullanarak sizin programınıza ait olmayan bir formu bile sürekli en üstte kalmasını sağlayabilirsiniz.

Form Başlığının rengini değiştirmek

```
Declare Function FlashWindow Lib "User32" (ByVal hWnd As Integer, ByVal bInvert As Integer) As Integer
```

Form başlığı, formun aktif veya pasif olmasına göre rengi değişir. Ancak bu rengi FlashWindow API'si ile sizde değiştirebilirsiniz.

hWnd parametresi ile handle numarası verilen formun, bInvert parametresine True verilirse Formun şu andaki başlık rengi d\$tr duruma geçer. False ise formun başlığı orjinal konumuna döner. Yani aktifte aktif renge, pasif ise pasif renge.

Burada kastedilen formun aktifliği ve pasifliği formun Enabled özeliği ile ilgili değildir. Aktiflikle o anda ekranda kontrolü elinde bulunduran form, pasiflikle de kontrolü elinde bulundurmayan diğer formlar kastedilmektedir.

Bu API kullanılarak bir timer kontrolüyle formunuzun yanıp sönmesini sağlayarak dikkat çekmesini sağlayabilirsiniz.

```
Private Declare Function FlashWindow Lib "User32" (ByVal hWnd As Integer, ByVal bInvert As Integer) As Integer
```

```
Sub Timer1_Timer()  
Call FlashWindow(Form1.hWnd, True)  
End Sub
```

Windowsu Kapatmak

```
Declare Function ExitWindowsEx Lib "user32" (ByVal uFlags As Long, ByVal dwReserved As Long) As Long
```

Windowsu kapatmak veya bilgisayarı yeniden başlatmak için bu API kullanılabilir. uFlags parametresi ile ne yapılmak istendiği bildirilir. Bu parametreye aşağıdaki değerlerden biri verilebilir.

- 0:** Aktif kullanıcıyı kapatır ve yeniden başlar.
- 1:** Windowsu kapatır ve Bilgisayarınızı Kapatabilirsiniz ekranı çıkar.
- 2:** Bilgisayarı yeniden başlatır.
- 3:** 1 değeri gibidir. Farklı olarak bilgisayar destekliyse, bilgisayarın gücünde keser.

4: Sisteme cevap vermeyen uygulamalar varsa bunların sonlandırılması için kullanıcıdan onay ister.

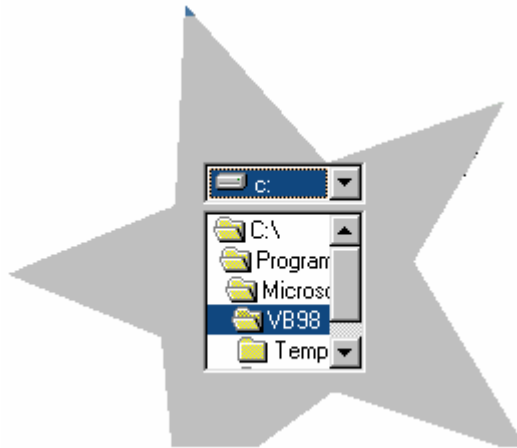
Geriye dönen değer true ise işlem başarılmıştır. False ise kapatılma işlemi bir program tarafından iptal edilmiştir.

ÖRNEK: ExitWindowsEx

```
Private Declare Function ExitWindowsEx Lib "user32" (ByVal uFlags
As Long, ByVal dwReserved As Long) As Long
Sub Command1_Click()
Dim c
c = MsgBox("Windows Yeniden Baslatılacak. Onaylıyormusunuz ?",
vbYesNo, "Restart")
If c = vbYes Then c =
ExitWindowsEx(42, 0)
End If
End Sub
```

Formlara Şekil Vermek

Formlarınız klasik dikdörtgen olmak zorunda değildir. İsterseniz formlarınızı daire, elips, çokgen vb şekillerde de hazırlayabilirsiniz.



Yapmanız gereken öncelikle CreateEllipticRgn, CreatePolyPolygonRgn, CreateRectRgn gibi bir API ile şekli oluşturmak ve buradan elde edeceğiniz şeklin handle numarasını SetWindowRgn API'sinde kullanmanız.

```
Private Declare Function CreateEllipticRgn Lib "gdi32" Alias
"CreateEllipticRgn" (ByVal X1 As Long, ByVal Y1 As Long, ByVal
X2 As Long, ByVal Y2 As Long) As Long
```

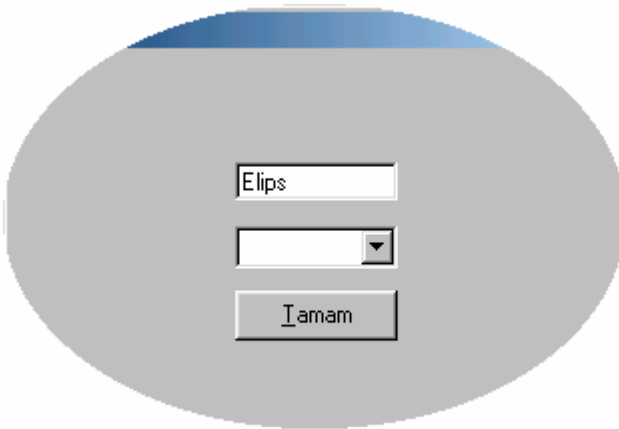
Bu API'deki parametreler oluşturulacak elipsin içine yerleştirileceği dikdörtgenin koordinatlarını belirler. İşlem başarılıysa oluşan şeklin handle numarası geri dönecektir.

Oluşturulan bu şekli forma uygulayan API ise SetWindowRgn'dir.

```
Private Declare Function SetWindowRgn Lib "user32" (ByVal hwn  
As Long, ByVal hrgn As Long, ByVal bdraw As Boolean) As  
Integer
```

Buradaki hwn parametresi şekli değiştirilecek formun handle numarasını, hrgn parametresi uygulanacak şeklin handle numarasını belirler, bdraw parametresi ise şeklin uygulanması için true olmalıdır.

```
Option Explicit  
Private Declare Function CreateEllipticRgn Lib "gdi32" (ByVal  
X1  
As Long, ByVal Y1 As Long, ByVal X2 As Long, ByVal Y2 As Long) As  
Long  
Private Declare Function SetWindowRgn Lib "user32" (ByVal hwn  
As  
Long, ByVal hrgn As Long, ByVal bdraw As Boolean) As Integer  
  
Private Sub Form_Load()  
Dim şekilHandle As Long  
Show  
ScaleMode = 3  
şekilHandle = CreateEllipticRgn(0, 0, Form1.ScaleWidth,  
Form1.ScaleHeight)  
Call SetWindowRgn(Form1.hWnd, şekilHandle, True)  
End Sub
```



CreatePolygonRgn apisi aracılığıyla çokgenler oluşturulup formlar bu sekle sokulabilir.

```
Private Declare Function CreatePolygonRgn Lib "gdi32" (IpPoint As  
POINTAPI, ByVal nCount As Long, ByVal nPolyFillMode As Long) As  
Long
```

Buradaki **IpPoint** parametresi aşağıdaki **POINTAPI** tipinden tanımlanmıştır ve oluşturulacak şeklin köşe koordinatlarını belirleyen bir dizidir. **nCount** parametresi ise köşe sayısını bildirir.

```
Private Type POINTAPI  
x As Long  
y As  
Long End  
Type
```

ÖRNEK:

```
Option Explicit  
Private Declare Function CreateEllipticRgn Lib "gdi32" (ByVal XI  
As Long, ByVal Y1 As Long, ByVal X2 As Long, ByVal Y2 As Long) As  
Long  
Private Declare Function SetWindowRgn Lib "user32" (ByVal hwn As  
Long, ByVal hrgn As Long, ByVal bredraw As Boolean) As Integer  
Private Declare Function CreatePolygonRgn Lib "gdi32" (IpPoint As  
POINTAPI, ByVal nCount As Long, ByVal nPolyFillMode As Long) As  
Long
```

```
Private Type POINTAPI  
x As Long  
y As Long End Type
```

```
Private Sub Form_Load()  
Dim sekilHandle As Long  
Dim p(1 To 9) As  
POINTAPI  
Show  
ScaleMode = 3
```

```
p(1).x = 100: p(1).y = 100  
p(2).x = 250: p(2).y = 0  
p(3).x = 350: p(3).y = 0  
p(4).x = 550: p(4).y = 100  
p(5).x = 550: p(5).y = 200  
p(6).x = 350: p(6).y = 300
```

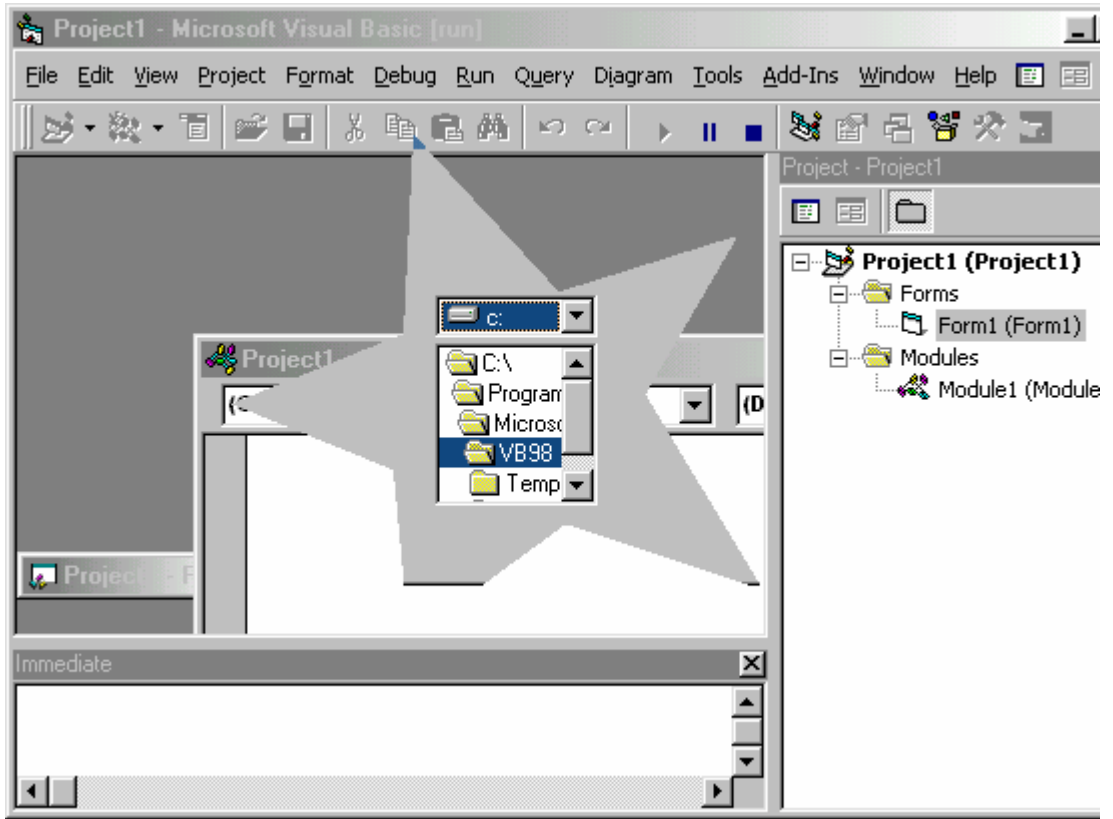
Microsoft Visual Basic 6.0

```
p(7).x = 250: p(7).y = 300
p(8).x = 100: p(8).y = 200
p(9).x = 100: p(9).y = 100 'basi ve sonu birlestirin
sekilHandle = CreatePolygonRgn(p(1), 9, 0)
Call SetWindowRgn(Form1.hWnd, sekilHandle, True)
End Sub
```

ÖRNEK: Yıldız seklinde bir form

```
Option Explicit
Private Declare Function CreateEllipticRgn Lib "gdi32" (ByVal XI
As Long, ByVal Y1 As Long, ByVal X2 As Long, ByVal Y2 As Long) As
Long
Private Declare Function SetWindowRgn Lib "user32" (ByVal hwn
As Long, ByVal hrgn As Long, ByVal bredraw As Boolean) As
Integer
Private Declare Function CreatePolygonRgn Lib "gdi32" (IpPoint As
POINTAPI, ByVal nCount As Long, ByVal nPolyFillMode As Long) As
Long

Private Type POINTAPI
x As Long
y As Long End Type
Private Sub Form_Load()
Dim sekilHandle As Long
Dim p (1 To 11) As POINTAPI
Show
ScaleMode = 3
p(1).x = 14: p(1).y = 149
p(2).x = 100: p(2).y = 114
p(3).x = 105: p(3).y = 14
p(4).x = 178: p(4).y = 93
p(5).x = 267: p(5).y = 65
p(6).x = 221: p(6).y = 144
p(7).x = 280: p(7).y = 242
p(8).x = 181: p(8).y = 209
p(9).x = 103: p(9).y = 266
p(10).x = 95: p(10).y = 182
p(11).x = 14: p(11).y = 149
sekilHandle = CreatePolygonRgn(p(1), 11, 0)
Call SetWindowRgn(Form1.hWnd, sekilHandle, True)
End Sub
```



Bir timer olayına yazacağınız kodla formun şeklini sürekli değiştirip animasyonlar da yaptırabilirsiniz.

Formu Taskbarda Gizleme

VB'de oluşturduğunuz formların simgesi görev çubuğuna yerleşir. Özellikle programınızda çok sayıda form varsa bu durum istenmez. Programınızdaki formların taskbarda görülmesini istemiyorsanız

GetWindowLong apisi aracılığıyla formun özelliklerini öğrenip **Set Window Log** apisi ile bu özellikleri değiştirerek formun taskbarda görülmemesini sağlayabilirsiniz.

```
Private Declare Function GetWindowLong Lib "user32" Alias
"GetWindowLongA" (ByVal hwnd As Long, ByVal nIndex As Long) As
Long
Private Declare Function SetWindowLong Lib "user32" Alias
"SetWindowLongA" (ByVal hwnd As Long, ByVal nIndex As Long,
ByVal dwNewLong As Long) As Long
Option Explicit
```

Microsoft Visual Basic 6.0

```
Private Declare Function SetWindowLong Lib "user32"
Alias "SetWindowLongA" (ByVal hwnd As Long, ByVal nIndex As
Long, ByVal dwNewLong As Long) As Long
Private Declare Function GetWindowLong Lib "user32"
Alias "GetWindowLongA" (ByVal hwnd As Long, ByVal nIndex As
Long) As Long
Private Const WS_EX_APPWINDOW = &H40000 Private Const GWL
EXSTYLE = (-20)
Private Sub Command1_Click ()
Dim eski, yeni As Long
eski = GetWindowLong(hwnd, GWL_EXSTYLE)
yeni = eski And (Not WS_EX_APPWINDOW)
Hide
SetWindowLong hwnd, GWL_EXSTYLE, yeni
Show
End Sub
```

Menülerle ilgili API'ler

Windows API'lerini kullanarak menüleri de çok daha kullanışlı bir şekilde getirebilirsiniz. Örneğin menülerinize resim ekleyebilirsiniz. Bu kısımda da menülerle işlem yapacak bazı API'leri vereceğiz.

Her Windows kontrolünde olduğu gibi menülerin de bir handle numarası vardır ve menülere bu handle numaraları vasıtasıyla ulaşılır. Bir form üzerinde bir tane menü olmayacağı için menülere verilen handle numaraları biraz daha farklıdır. Öncelikle formun menü çubuğu için bir handle verilir. Bu handle numarası ile ana menülere, bu ana menülere verilen handle numarasıyla da alt menülere ulaşılır.

Ayrıca formların sol üst köşesinde bulunan System menüsünün (buna kontrol menüsü de denir) handle numarası da vardır.

Kontrol-System Menüsü



Eğer formunuzun bir kontrol kutusu var ise bu kutuya yeni seçenekler ekleyebilirsiniz. Bu is menü çubugundaki menülere eleman eklemekten daha karisiktir. Windows kontrol menusunu (buna API'lerde System menuyu de deniyor) menü çubugundaki menülerden ayrı olarak görür. Ve bu menüdeki seçeneklerin tanımlanmış işlemleri vardır. Şimdi vereceğimiz API'lerle bu menüye yeni elemanlar ekleyebileceğiniz gibi var olan standart seçeneklerin işlevlerini de değiştirebilirsiniz. Diğerlerinde olduğu gibi system menuyu üzerinde de işlem yapabilmek için bu menünün handle numarasını öğrenmemiz gerekir.

System Menusunun Handle Numarasını Öğrenmek

```
Declare Function GetSystemMenu Lib "User32" (ByVal hWnd As Integer, ByVal bRevert As Integer) As Integer
```

hWnd: System menuyu handle numarası öğrenilecek formun handle numarası

bRevert: Bu parametreye False verilirse şu andaki system menuyunun handle numarası verilir, True verilirse system menuyu orjinal haline getirilir.

Aşağıdaki gibi kullanılarak Form1'e ait system menuyunun handle numarası öğrenilebilir.

```
Sysmenuh = GetSystemMenu (Form1.hWnd, False)
```

Artık, bu handle numarasını kullanarak menü üzerinde işlem yapabiliriz. Bu numarayı kullanarak menüye yeni elemanlar ekleyelim.

System Menüüne Yeni Seçenekler Ekleme

Microsoft Visual Basic 6.0

Declare Function AppendMenu Lib "User32" Alias "AppendMenuA" (ByVal hMenu As Integer, ByVal wFlags As Integer, ByVal wIDNewItem As Integer, ByVal lpNewItem As Any) As Integer

hMenu:Yeni seçeneklerin ekleneceği menünün handle numarası

wFlags:Bu parametrenin alabileceği bir kaç değer var ve bu değerler sonraki iki parametrenin kullanımını etkiler. Menüye eleman eklemek için bu parametreye 0 (MF_STRING) değerini vereceğiz.

WIDNewItem: Eklenecek yeni seçeneğin tanıtıcı numarası. Buraya vereceğimiz değerle ileride menüyü taniyacağız. Bu parametreye verebileceğiniz değerler &H10 ile &HFOOO arasında son basamağı 0 olan hexadecimal sayılar olabilir. Son basamağı 0 vermezseniz Windows son basamağı dikkate almayacaktır.

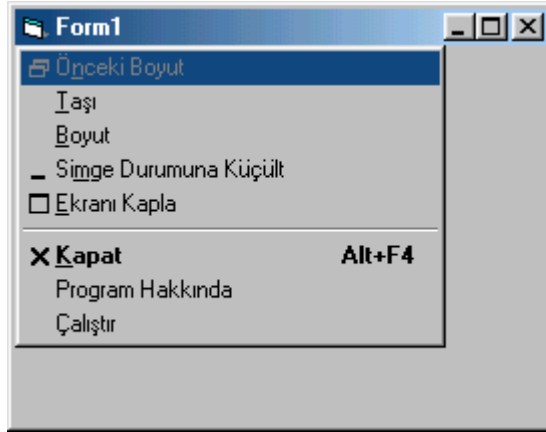
LpNewItem: wFlags parametresine bağlı olarak bu parametreye menü seçeneğinin textini yazacağız.

Fonksiyondan geri dönen değer 0 ise işlem başarısızdır.

ÖRNEK: GetMenu, AppendMenu
Option Explicit

```
Private Declare Function AppendMenu Lib "User32" Alias
"AppendMenuA" (ByVal hMenu As Integer, ByVal wFlags As Integer,
ByVal wIDNewItem As Integer, ByVal lpNewItem As Any) As Integer
Private Declare Function GetSystemMenu Lib "User32" (ByVal hWnd
As Integer, ByVal bRevert As Integer) As Integer
Dim sh, x, tanitici1, tanitici2
Private Sub Form_Load()
tanitici1 = 10
tanitici2 = 20
sh = GetSystemMenu(Form1.hWnd, False)
x = AppendMenu(sh, 0, tanitici1, "Program Hakkında")
x = AppendMenu(sh, 0, tanitici2, "Çalıştır")
End Sub
```

Bu satırlarla system menüsü aşağıdaki gibi olacaktır. Burada verdiğimiz tanitici1 ve tanitici2 numaralarını daha sonra bu menünün seçildiğini anlamak için kullanacağız. Bu yüzden bu sayıları Global olarak tanımlamanız gerekebilir.



Evet buraya kadar hiç bir problem yok. Simdi programi alistirin ve kontrol penceresinin yukaridaki hale geldigini grn. Ekledigimiz menlerden birini seerseniz hi bir aksiyon gremeyeceksiniz. nk henz olay alt programini yazmadik.

Yazalim. Ama nereye ? Bu seenekleri VB saglamadigi iin bu seeneklere ait olay alt programlarini da saglamayacaktır. O halde biz srekli olarak mennn seilip seilmedigini kontrol etmemiz gerekiyor. Dogal olarak bu kontrol yapmamizi saglayacak komutlari da VB saglamiyor. Is yine basa (API'lere) dsyor. nce Windows bir olay oldugunda nasil davranir onu anlamaya alisalim.

Mesaj Kuyrugundaki Mesaji Almak

Bir olay meydana geldiginde Windows bu olayi, olay hangi programa aitse o programa bildirir. Bu bildirme ilemi programin mesaj kuyruguna mesajin eklenmesiyle olur. Program bu mesaj kuyrugundaki mesajlari alarak ileme sokar. Normalde VB bu ii bizim yerimize yapar. Yani mesaj kuyrugundan aldigi mesajlari yorumlayarak mesaj hangi kontroln, hangi olayina aitse o olay alt programini agirir. Yukarida da belirttigimiz gibi System mensne eklenen seenekler VB tarafından saglanmadigi iin VB bu mesajı isleyemeyecektir. Bu durumda mesajlari srekli olarak kontrol ederek system mensne ait olan mesajlari bizim yorumlamamiz gerekiyor. Mesaj kuyrugundan bir mesajı almak iin GetMessage API'sini kullanacagiz.

Microsoft Visual Basic 6.0

```
Declare Function GetMessage Lib "User32" Alias "GetMessageA"  
(IpMsg As MSG, ByVal hWnd As Integer, ByVal wParam As Integer,  
ByVal lParam As Integer) As Integer
```

hWnd: Mesaj alınacak formun handle numarası

**wParamFilterMin,
wParamFilterMax:**

Bu iki parametre vasıtasıyla mesajlar filtrelemeye tabi tutulabilir. Parametrelere 0 değeri verilirse bütün mesajlar alınabilir.

IpMsg:

Siradaki mesaj MSG yapısında tanımlanmış bu parametre ile geri döner. Mesaj yapısı aşağıda verilmiştir.

Fonksiyondan geri dönen değer 0 ise WM_QUIT mesajı alınmıştır.
Bu mesaj ile birlikte program sona erer.

```
Type POINT  
    x As Integer  
    y As Integer  
End Type
```

```
Type MSG  
    hwnd As Long  
    message As Long  
    wParam As Long  
    lParam As Long  
    time As Long  
    pt As POINT  
End Type
```

hWnd: Mesajın ait olduğu pencerenin handle numarası
time: Mesajın alındığı zaman
message: Alınan mesajın numarası. Bu numaraya bakılarak mesajın hangi sınıfa ait olduğu öğrenilebilir. Bu değer &H112 ise bu bir system menu mesajıdır. Yani system menu ile ilgili bir işlem yapılmıştır.

Wparam, lparam: Message parametresi mesajın sınıfı hakkında bilgi verir ancak işlem hakkında bilgi vermez. Mesajla ilgili işlem bu iki parametre ile öğrenilir. Bu iki parametrenin alacağı değerin anlamı tamamen Message parametresine bağlıdır.

Mesaj bir system menu mesajı ise (message = &H112 ise) lparam parametresinin düşük seviyeli iki byte'ı mouse'un x koordinatı, yüksek

seviyeli iki byte'ida y koordinatini verir. İşlem mouse ile yapılmamışsa bu parametrenin bir anlamı yoktur.

Wparam parametresi ise yapılan işlemin kodunu verir. Bu kod:

&HF060: Kapat seçeneği
&HF030: Ekranı kapla seçeneği
&HF020: Simge durumuna küçült seçeneği
&HF010: Tasi seçeneği
&HF120: Önceki boyut seçeneği
&HFOOO: Boyut seçeneği
&HF130: Geçiş Yap seçeneği

TanıtıcıNo: Eğer seçilen seçenek standart bir menü seçeneği değilse yani bizim eklediğimiz bir seçenek ise geri dönen değer AppendMenu API'sinin WIDNewItem parametresine verdiğimiz değerdir.

```
Dim a, msg1 As MSG
```

```
a = GetMessage(msg1, form1.hWnd, 0, 0)  
If msg1.message = &H112 Then 'WM_SYSCOMMAND ise
```

```
Select Case msg1.wParam  
Case &H10: MsgBox ("Program Hakkında Seçildi")  
Case &H20: MsgBox ("Çalıştır seçildi")  
End Select
```

```
End If
```

Bu şekilde bir kodla mesajı alıp işleyebiliriz. Ancak bu kod eksiktir. Çünkü mesajı kuyruktan alıyoruz ancak sadece kendi isimize yarıyorsa kullanıyoruz, yaramıyorsa yani system menüsü ile ilgili bir seçenek değilse hiçbir şey yaptırmıyoruz. Bu da diğer mesajların VB tarafından iletilmesini önleyecektir. Mesaj system menüsüne ait bir mesaj değilse DispatchMessage API'si ile mesajı işlenmesi için forma göndermemiz gerekir.

```
Declare Function DispatchMessage Lib "User32" Alias  
"DispatchMessageA" (IpMsg As MSG) As Long  
IpMsg: Gönderilecek mesaj. Bu mesaj GetMessage  
API'si ile aldığımız  
mesaj olmalıdır.
```

Geri dönen değer mesajı işleyen prosedürün gönderdiği değerdir ve genellikle bir anlamı yoktur.

Örneğimizi bu şekilde yeniden düzenleyelim.

```
Dim a, msg1 As MSG
```

Microsoft Visual Basic 6.0

```
a = GetMessage(msgl, form1.hWnd, 0, 0) If  
msgl.message = &H112 Then 'WM_SYSCOMMAND ise  
Select  
Case msgl.wParam  
Case &H10: MsgBox ("Program Hakkında Seçildi")  
Case &H20: MsgBox ("Çalıştır seçildi")  
Case Else : a = DispatchMessage(Msgl)  
End Select Else  
a = DispatchMessage(Msgl)  
End If
```

Böylece mesaj bir system menusu mesajı ve bizim ekledigimiz seçeneklere ait bir olayı yazdigimiz kod çalışıyor, değilse mesajı islenmek üzere pencereye gönderiyoruz.

Kodumuz bu hali ile sadece bir defa kontrol ediyor. Halbuki ekledigimiz seçeneğin ne zaman seçileceği kullanıcıya bağlıdır. Dolayısı ile mesajları sürekli olarak kontrol etmemiz gerekir. Bu işlem için bir modül içindeki Startup özelliği ile belirlenen bir alt programı kullanabiliriz. Bunun için bir modül içine Main isimli bir altprogram oluşturun ve Option/Project menüsündeki Start Up Form seçeneğini Sub Main olarak registirin. Böylece program buradan çalışmaya başlayacaktır. Burada bir While-Wend döngüsü içinde yukarıdaki kontrolü yapabiliriz. Ne zamana kadar, GetMessage API'sinden 0 yani quit mesajı gelinceye kadar.

Örneğimiz için form üzerine bir Common Dialog kontrolü yerleştirin ve Project menüsünden yeni bir modül ekleyin. Ve Project-Project Properties menüleri ile açılan pencereden Startup Object olarak Sub Main seçin.

ÖRNEK : System Menusu 'Bu kısım Modüle içine yazılacak.

Option Explicit

```
Type POINT  
    X As Integer  
    y As Integer  
End Type  
  
Type MSG  
    hwnd As Long  
    message As Long  
    wParam As Long  
    lParam As Long  
    time As Long  
    pt As POINT  
End Type
```

```
Declare Function GetMessage Lib "user32" Alias "GetMessageA"
(IpMsg As MSG, ByVal hwnd As Integer, ByVal wParamFilterMin As
Integer, ByVal wParamFilterMax As Integer) As Integer
```

```
Declare Function AppendMenu Lib "user32" Alias "AppendMenuA"
(ByVal hMenu As Integer, ByVal wFlags As Integer, ByVal wIDNewItem
As Integer, ByVal lpNewItem As Any) As Integer
```

```
Declare Function TranslateMessage Lib "user32" Alias
"TranslateMessageA" (IpMsg As MSG) As Integer
```

```
Declare Function DispatchMessage Lib "user32" Alias
"DispatchMessageA" (IpMsg As MSG) As Long
Declare Function DefWindowProc Lib "user32" Alias
"DefWindowProcA" (ByVal hwnd As Integer, ByVal wParam As Integer,
ByVal lParam As Integer, lParam As Any) As Long
Declare Function GetSystemMenu Lib "user32" (ByVal hwnd As
Integer, ByVal bRevert As Integer) As Integer Dim msgl As MSG
```

```
Sub main()
Dim a, hwnd, X
Form1.Show
hwnd = GetSystemMenu(Form1.hwnd, False)
X = AppendMenu(hwnd, 0, &H10, "Hakkında")
X = AppendMenu(hwnd, 0, &H20, "Çalıştır")
'bu &H10 ve &H20 sayılarını daha sonra kullanacağız
While GetMessage(msgl, Form1.hwnd, 0, 0) <> 0
If msgl.message = &H112 Then 'WM_SYSCOMMAND ise
menu Else
a = DispatchMessage(msgl) 'Mesajı işle
End If Wend
End Sub

Sub menu()
*
Dim a
Select Case msgl.wParam-Case &H10:
a = MsgBox("System Menusu Örneği. " + Chr(10) + Chr(13) +
"Cihan CAN Nisan 2003, Hendek/SAKARYA", 64, "Visual Basic")
Case &H20:
Form1.CommonDialog1.Action = 1
If Form1.CommonDialog1.filename <> "" Then
a = Shell(Form1.CommonDialog1.filename, 1) End If Case &HF020:
Form1.WindowState = 1 DoEvents
a = DefWindowProc(Form1.hwnd, msgl.message, msgl.wParam,
msgl.lParam)

Case Else: a = DispatchMessage(msgl)
```

```
End Select
```

```
End Sub
```

Menu çubugunun handle numarasini öğrenmek

```
Declare Function GetMenu Lib "User32" (ByVal hWnd As Integer) As Integer
```

hWnd parametresi ile verilen formun menu handle numarasini verir.

Fonksiyondan geri dönen deger 0 ise ya handle numarasini verilen form sistemde yüklü degildir yada menü çubuğu yoktur, yani formun menüleri yoktur.

```
Dim MnHandle
MnHandle = GetMenu(Form1.hWnd)
If MnHandle Then
    Print "Menu çubugunun handle numarasini: " ; GetMenu(Form1.hWnd)
Else
    Print "Formun menusu yok"
End If
```

Burada formun handle numarasini yerine kendi programımıza ait bir formun handle numarasini verdik. Ancak formlarla ilgili verdigimiz API'leri kullanarak kendi programınıza ait olmayan formlarin menülerine de burada verecegimiz API'leri uygulayabilirsiniz.

AltMenülerin handle numarasini öğrenmek

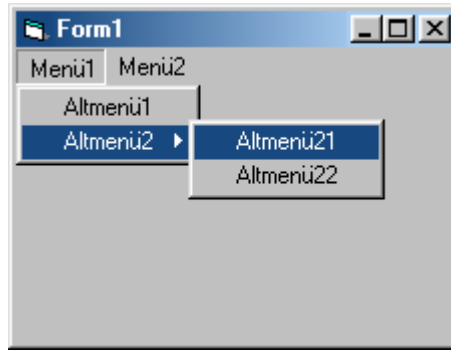
```
Declare Function GetSubMenu Lib "User32" (ByVal hMenu As Integer, ByVal nPos As Integer) As Integer
```

hMenu: Altmenüsünün handle numarasini öğrenilecek menünün, handle numarasini.

nPos: Alt menünün menu içindeki sirasi. İlk alt menü için 0.

Geri dönen deger 0 ise handle numarasini verilen menü veya pozisyonu verilen menü yoktur. Aksi takdirde geri dönen deger menünün handle numarasidir.

Bu API ile herhangi bir menünün alt menülerinin handle numarasini öğrenebiliriz. Bir menü asagidaki durumlardan birinde olabilir.



1-Menü çubugu üzerindeki menüler (Menü1, Menü2) formunuzun alt menüleridir. Dolayısıyla bu menülerin handle numarasini öğrenmek için hMenu yerine menü çubugunun handle numarasini vermeniz gerekir.

```
Dim MnCHandle, MnHandle
MnCHandle = GetMenu (Form1.hWnd) 'Menu çubugunun hwnd si
Print "Menu Çubugunun handle numarası"; MnCHandle
MnHandle = GetSubMenu (MnCHandle, 0)
Print "Menü1'in alt handle numarası"; MnHandle
MnHandle = GetSubMenu (MnCHandle, 1)
Print "Menu2'nin handle numarası"; MnHandle
```

Bir menünün alt menüleri yoksa onun handle numarası olmaz. O menüye bulunduğu yerin handle numarası ile ulaşılır. Yani Menü2'nin handle numarası onun alt menülerine ulaşmak için kullanılır. MenüZ'ye ulaşmak için ise menü çubugunun handle numarası kullanılır.

2-Eğer handle numarası öğrenilecek menu, menu çubugu üzerindeki menünün alt menuyu ise (AltMenu1, AltMenu2) hMenu yerine menü çubugu üzerindeki menünün (Menu1) handle numarasini vermemiz gerekir.

```
Dim MnCHandle, AMnHandle, MnHandle
MnCHandle = GetMenu(Form1.hWnd) 'Menu çubugunun hwnd si
Print "Menu Çubugunun handle numarası"; MnCHandle
MnHandle = GetSubMenu (MnCHandle, 0)
Print "Menü1'in handle numarası"; MnHandle
AMnHandle = GetSubMenu(MnHandle, 1)
Print "AltMenu2'nin handle numarası"; AMnHandle
```

3-Eğer menü bu menünün de bir alt menuyu ise 2. Basamaktaki işlem tekrar uygulanır.

Görüldüğü gibi bir altmenünün handle numarasini öğrenmek için ondan önceki menülerin handle numaraları öğrenilebilir. Burada menülerin alt menü sayılarini bildigimiz için nPos yerine gerekli degeri verebildik.

Ancak bir menünün alt menusu olmayabilir veya sayısı belli olmayabilir. Bu sayıyı öğrenmek içinde GetMenuItemCount API'si! bulunmaktadır.

Menülerin alt menü sayisini öğrenmek

```
Declare Function GetMenuItemCount Lib "User32" (ByVal hMenu As Integer) As Integer
hMenu: Altmenü sayısı öğrenilecek menünün handle numarası.

Geri dönen deger -1 ise handle numarası verilen menü mevcut değildir. Aksi taktirde geri dönen deger alt menü sayisidir. Dolayisiyla geri dönen deger 0 ise menünün alt menusu yoktur.
Print "Menu çubugu üzerindeki menu sayısı"; Print
GetMenuItemCount( GetMenu(Form1.hWnd))
```

Bu API'lerle menülerin handle numarasini ve alt menu sayilarini nasıl öğreneceğimizi biliyoruz. Bunlar menülerle ilgili API'leri kullanabilmek için gerekli idi. Şimdi bu API'leri kullanarak menülerle ilgili bazı işler yapan API'leri verelim.

Menülere resim eklemek

```
Declare Function GetMenuCheckMarkDimensions Lib "User" () As Long
```

Menülerinize resim ekleyerek menülerinizin daha güzel görülmesini sağlayabilirsiniz. Aslında bu API'yi kullanmasanız dahi Windows menüler için bir resim kullanır. Bahsettiğimiz resim menü işaretli iken gösterilen ^ işaretidir. Bir menu işaretli ise (Checked = True) bu resim menü isminin başında gösterilir. İşaretli değilse boş bir resim gösterilir. İşte bu resimleri istediğiniz resimlerle değiştirebilirsiniz. Ancak resmin büyüklüğü belirli sınırlar içinde olmalıdır. GetMenuCheckMarkDimensions API'si bu boyutları verir.

Bu fonksiyondan geri dönen degerin düşük seviyeli iki byte'i resmin genişliğini, yüksek seviyeli byte'i yüksekliğini pixel olarak verir. Bu deger 640x480 standart VGA çözünürlüğü için 14x14'dür.

```
Dim x
X = GetMenuCheckMarkDimensions ()
Print "Menu için kullanilabilecek resimin"
Print " genişliği   :", x And &HFFFF&
Print " yüksekliği:", (x And &HFFFF0000) / 65535)
```

```
Declare Function SetMenuItemBitmaps Lib "User" (ByVal hMenu As Integer, ByVal nPosition As Integer, ByVal wFlags As Integer, ByVal hBitmapUnchecked As Integer, ByVal hBitmapChecked As Integer) As Integer
```

Resmin olabilecek boyutlarini öğrendikten sonra bu resmi degistirelim. Bir menü iki durumdan birinde olabilir. Isaretli veya isaretsiz. Ve bu her iki durum için ayri bir resim yükleyebilirsiniz. Bu resimleri degistirecek API SetMenuitemBitMaps API'sidir. Bu API ile bir menünün alt menülerinin isaretli ve isaretsiz durumları için iki ayri resim yükleyebilirsiniz. Dolayisi ile her menünün alt menüleri için farkli resimler yükleyebilirsiniz.

hMenu: Menünün handle numarasi.

nPosition: Altmenünün menü içindeki sirasi. İlk menü için bu deger O'dir.

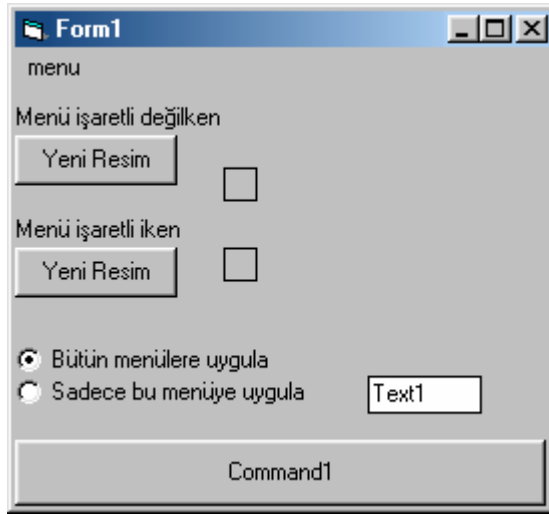
Wflags: Bu parametre O veya &H400 degerlerinden birini alarak nPosition parametresinin nasil kullanacagi belirlenir.

HBitmapUnchecked: Menu isaretsiz iken kullanılacak resim.

HBitmapChecked: Menü isaretli iken kullanılacak resim. Bu degerler yerine O verilirse Windows orjinal resimleri kullanir.

Geri dönen deger O degilse islem basarilmistir.

ÖRNEK: Örnek olarak menülere resim ekleyecek bir program yazalım. Örneğimiz için asagidaki formu ve menüleri hazirlayin. Ayrica iki Image kontrolü içine de uygun resimleri yerlestirin. "Yeni Resim" baslikli düğmeleri ve alt menüleri birer dizi olarak tasarlayin.



ÖRNEK : Menu API

Option Explicit

```
Private Declare Function SetMenuItemBitmaps Lib "User32" (ByVal  
hMenu As Integer, ByVal nPosition As Integer, ByVal wFlags As  
Integer, ByVal hBitmapUnchecked As Integer, ByVal hBitmapChecked As  
Integer) As Integer
```

```
Private Declare Function GetMenu Lib "User32" (ByVal hWnd As  
Integer) As Integer
```

```
Private Declare Function GetMenuCheckMarkDimensions Lib "User32" (  
As Long
```

```
Private Declare Function GetMenuItemID Lib "User32" (ByVal hMenu As  
Integer, ByVal nPos As Integer) As Integer
```

```
Private Declare Function GetMenuItemCount Lib "User32" (ByVal hMenu  
As Integer) As Integer
```

```
Private Declare Function SetMenu Lib "User32" (ByVal hWnd As  
Integer, ByVal hMenu As Integer) As Integer
```

```
Private Declare Function CheckMenuItem Lib "User32" (ByVal hMenu As  
Integer, ByVal wIDCheckItem As Integer, ByVal wCheck As Integer) As  
Integer
```

```
Private Declare Function GetSubMenu Lib "User32" (ByVal hMenu As  
Integer, ByVal nPos As Integer) As Integer
```

```
Private Sub coramand1_click()
```

```
Dim anamenuhandle, menuhandle Dim x, menusay, bas, son, i
```

```
anamenuhandle = GetMenu(Form1.hWnd) menuhandle =
```

```
GetSubMenu(anamenuhandle, 0) menusay =
```

```
GetMenuItemCount(menuhandle) If Option1 Then
```

```
bas = 0
```

```
son = menusay - 1 Else
```

```
bas = Val(Text1) - 1
```



```
son = bas End If If Command1.Caption =  
"Resimli" Then  
For i = bas To son  
x = SetMenuItemBitmaps(menuhandle, i, &H400,  
Image2.Picture, Image1.Picture)  
Next  
Command1.Caption = "Orj inal" Else  
For i = bas To son  
x = SetMenuItemBitmaps(menuhandle, i, &H400, 0, 0)  
Next  
Command1.Caption = "Resimli" End If  
End Sub  
  
Private Sub Command2_Click(index As Integer)  
CommonDialog1.Action = 1  
If CommonDialog1.filename = "" Then Exit Sub 'iptal seçildi ise If  
index = 1 Then  
Image2.Picture = LoadPicture(CommonDialog1.filename)  
Else  
Image1.Picture = LoadPicture(CommonDialog1.filename)  
End If  
  
If Command1.Caption = "Orj inal" Then 'Menu resimli  
durumda ise yeni resimleri aktif yap  
Command1.Caption = "Resimli"  
command1_click  
End If  
End Sub  
  
Private Sub Form_Load()  
Dim size, x  
Command1.Caption = "Resimli"  
size = GetMenuCheckMarkDimensions()  
x = size End Sub  
  
Private Sub menu1_Click(index As Integer)  
menu1(index).Checked = Not menu1(index).Checked End  
Sub  
  
Private Sub Option1_Click()  
Text1.Visible = False  
End Sub  
  
Private Sub Option2_Click()  
Text1.Visible = True  
Text1 = 1  
End Sub  
Private Sub Text1_LostFocus()  
Dim anamenuhandle, menuhandle
```

Microsoft Visual Basic 6.0

```
Dim menusey
anamenuhandle = GetMenu(Form1.hWnd)
menuhandle = GetSubMenu(anamenuhandle, 0)
menusey = GetMenuItemCount(menuhandle)
If Val(Text1) > menusey Then
MsgBox ("Menu numarası 1 ile " & menusey & " arasında
olabilir")
Text1.SetFocus
End If
End Sub
```

Menülere resim eklemenin bir yoluda elemanın tamamen resim olmasıdır. Bu işlem AppendMenü API'si ile yapılabilir.

```
Declare Function AppendMenü Lib "User32" Alias "AppendMenuA"
(ByVal hMenu As Integer, ByVal wFlags As Integer, ByVal
wIDNewItem As Integer, ByVal lpNewItem As Any) As Integer
```

hMenu: Yeni seçeneklerin ekleneceği menünün handle numarası

wFlags: Bu parametrenin alabileceği bir kaç değer var ve bu değerler sonraki iki parametrenin kullanımını etkiler. Menüye resimli eleman eklemek için bu parametreye 4 (MF_BITMAPS) değerini vereceğiz.

WIDNewItem: Eklenecek yeni seçeneğin tanıtıcı numarası. Buraya vereceğimiz değerle ileride menüyü tanıyacağız. Bu parametreye verebileceğiniz değerler &H10 ile &HFOOO arasında son basamağı 0 olan hexadesimal sayılar olabilir. Son basamağı 0 vermezseniz Windows son basamağı dikkate almayacaktır.

LpNewItem: wFlags parametresine bağlı olarak bu parametreye menü seçeneği için kullanılacak resmin Picture özelliğinin düşük seviyeli iki baytını vereceğiz.

Fonksiyondan geri dönen değer 0 ise işlem başarısızdır.

h = GetMenu(hWnd) 'formun menüsünü al

h = GetSubMenu(h, 0)

x = AppendMenu(h, &H4, 10, Picture1.Picture And &HFFFF)

Kapat Düğmesini Pasif Yapma

Windows 95 ile birlikte pencerelerin sağ üst köşesine x düğmesinin eklendiğini biliyorsunuz. Bunu pasif yaparak kullanıcının programı buradan sonlandıramamasını sağlayabilirsiniz. Aslında bunu pasif yapmak demek system menüsündeki kapat seçeneğini kaldırmak demektir.

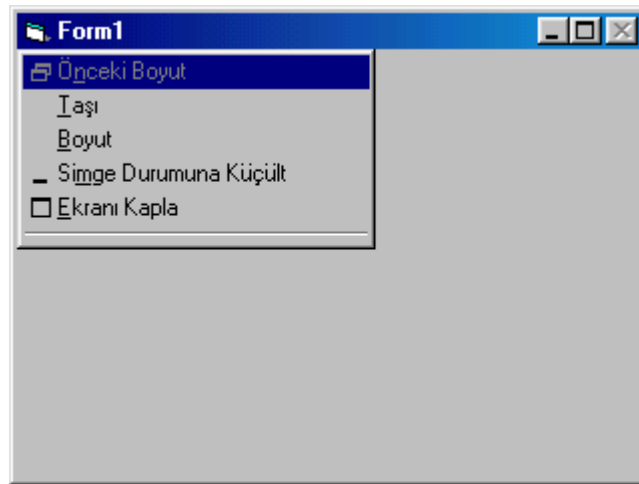
RemoveMenu apisi ile istediğiniz bir menüdeki seçeneği kaldırabilirsiniz. **GetSystemMenu** apisi ile system menüsünün handle numarasını öğrendikten sonra bu menüdeki 6. sırada olan kapat menüsünü **RemoveMenu** apisi ile kaldırabilirsiniz.

```
Private Declare Function GetSystemMenu Lib "User32" (ByVal hWnd As Integer, ByVal bRevert As Integer) As Integer

Private Declare Function RemoveMenu Lib "User32" (ByVal hMenu As Integer, ByVal nPosition As Integer, ByVal wFlags As Integer) As Integer
Private Const MF_BYPOSITION = &H400

Private Sub Form_Load()
    dim mnhandle
    mnhandle = GetSystemMenu(hWnd, 0)
    call RemoveMenu(mnhandle, 6, MF_BYPOSITION)
End Sub
```

Böylece x düğmesi pasif olan bir formunuz olacaktır.



Sıkıştırılmış Dosyalarla İlgili API'ler

Microsoft'un ürünlerini sıkıştırmak ve açmak için kullandığı DOS altında çalışan COMPRESS.EXE ve EXPAND.EXE programlarını kullanır. Windows altında da COMPRESS.EXE programı ile sıkıştırılmış dosyaları açmak için LZEXPAND.DLL kütüphanesi Windows'la birlikte gelmektedir. Böylece özellikle kurulum programlarında standart olarak bir sıkıştırma tekniğini kolayca kullanabilme imkanı verilmektedir. Ancak LZEXPAND.DLL ile dosyaları sıkıştıramıyor sadece sıkıştırılmış dosyaları açabiliyorsunuz. Dosyaları sıkıştırmak için DOS altında çalışan COMPRESS.EXE kullanılabilir.

Sıkıştırılmış bir dosyayı açmak için şu adımlar uygulanmalıdır.

1. LzOpenFile ile sıkıştırılmış dosyayı açmak(Open)
2. LzOpenFile ile açılacak dosyayı oluşturmak

3. LzCopy ile sıkıştırılmış dosyayı açarak (UnCompress) 2. Adımda oluşturulan dosyaya kopyalamak
4. LzClose ile, açılmış olan iki dosyayı da kapatmak.

Bu adımlarda görüldüğü gibi bir dosyayı açmak için gerekli üç API var. Bu API'leri sirasi ile verelim.

Sıkıştırılmış Dosyayı Açmak ve Hedef Dosyayı Oluşturmak

```
Declare Function LZOpenFile Lib "\windows\system\lz32.dll"  
Alias "LZOpenFileA" (ByVal IpszFile As String, IpOf As  
ofstruct, ByVal style As Long) As Long  
IpszFile:    Açılacak veya oluşturulacak dosyanın adı
```

IpOf: Açılan dosya hakkında bilginin kopyalandığı Ofstruct yapısında bir değişken. Aşağıda bu yapı verilmistir.

Style: Dosya üzerinde yapılacak işlemi belirleyen aşağıdaki sayılardan ÖR işlemine tabi tutularak elde edilen bir değer.

Style	Anlami
&HO	Dosya sadece okunmak için açılacak
&H1	Dosya sadece yazmak için açılacak
&H2	Dosya okunmak ve yazmak için açılacak
&H1000	Dosya oluşturulacak
&H200	Dosya silinecek
&H2000	Dosya bulunamazsa kutusu gösterilecek. Dosyanın bulunamadığına dair bir diyalog kutusu gösterilecek.

Geri dönen değer **-1** ise hata olmuştur. Bu hatanın ne olduğu ise Ipof parametresinin nErrCode alt değişkeni ile öğrenilir. Aksi takdirde geri dönen değer açılan veya oluşturulan dosyanın Handle numarasıdır. Ve bu değer dosyayla ilgili işlem yaparken kullanılır.

Ofstruct yapısı oluşturulan dosya hakkındaki bazı bilgileri içerir. Yapı bir modül içerisinde şu şekilde tanımlanmalıdır.

```
Type OFSTRUCT  
    cBytes As Integer  
    fFixedDisk As Integer  
    nErrCode As Long  
    Reserved1 As Long
```

```

        Reserved2 As Long
        szPathName As String *
128
End Type

```

cBytes: Bu yapinin byte olarak uzunlugunu bildirir.
Asc(ofs.cBytes) seklinde öğrenilebilir.

FFixedDisk: Buradan dönen deger 0 ise dosya
harddisktedir. (Asc(ofs.FFixedDisk))

reserved: Su an için bu parametrenin bir anlami yok. Ilerisi için
ayrilmis

szPathName: Açılan veya olusturulan dosyanin yolu ve sürücüsü dahil
adi.

nErrCode: LzOpenFile fonksiyonundan dönen deger -1 ise bu
parametre ile olusan hata kodu öğrenilebilir. Bu degiskenin alabilecegi
bazi degerler ve anlamları söyledir.

NerrCode	Anlami
&H0002	Dosya bulunamadi
&H0003	Yol bulunamadi
&H0004	Çok fazla sayida açık dosya var. (Bir kaç dosyanin kapatilmasi veya Config.Sys dosyasindaki FILES sayisinin
&H0005	Erisim engellendi (Readonly dosya olabilir)
&H0008	Bellek yetersiz
&H000B	Geçersiz Format
&H000F	Geçersiz sürücü harfi
&H0013	Write protect hatasi
&H0015	Sürücü hazır degil
&H0017	CRC hatasi
&H001B	Sector bulunamadi

NerrCode	Anlami
&H0020	Dosya baska bir uygulama tarafindan violation) kullaniliyor(Sharing
&H0036	Network okunamiyor
&H0041	Networka erisim engelleniyor
&H0050	Olusturulmak istenen dosya zaten var

```

Dim kaynakh
Dim ofs As ofstruct
'Dosyayi aç, dosya bulunamazsa hata mesaji ver

```

Microsoft Visual Basic 6.0

```
kaynakh = LZOpenFile ("sound.ba_", ofs, 0 Or &H2000)
'Hata olustu ise hata kodunu ver
If kaynakh = -1 Then
MsgBox ("Hata kodu:" & ofs.nErrCode)
End If
```

Sikistirilmis Dosyayi Hedef Dosyaya Açmak

```
Declare Function LZCopy Lib "\windows\system\lz32" (ByVal
hfSource As Long, ByVal hfDest As Long) As Long
```

hfSource: Kaynak (sikistirilmis) dosyanin handle numarası

hfDest: Hedef dosyanin handle numarası. Bu iki handle numarası da LZOpenFile ile açılan ve oluşturulan dosyaların handle numaralarıdır.

Bu API kaynak dosyayı açarak hedef dosyaya kopyalar. Fonksiyondan geri dönen değer dosyanın uzunluğudur. Dönen değer 0 dan küçük ise hata oluşmuştur. Ve olabilecek hata kodlarından bazıları şöyledir.

Hata kodu	Anlamı
-1	Kaynak dosya handle numarası yanlış
-2	Hedef dosya handle numarası yanlış
-3	Kaynak dosya formatı geçerli değil
-5	Bellek yetersiz
-8	Kaynak dosyanın sıkıştırılması için kullanılan algoritma

Dosyaları Kapatmak

```
Declare Sub LZClose Lib "\windows\system\lz32" (ByVal hfFile As
Long)
```

hfFile: Kapatılacak dosyanın handle numarası

ÖRNEK: Sıkıştırılmış bir dosyayı açmak için gerekli olan bu üç API'yi verdikten sonra sıkıştırılmış bir dosyayı açmak için bir örnek program yapalım.Örneğimiz i-çin formunuzun üzerine bir CommonDialog ve bir komut düğmesi yerleştirin.

'ÖRNEK : LZOpenFile, LZCopy, LZClose

Option Explicit

```
Private Declare Function LZOpenFile Lib
"\windows\system\lz32.dll" Alias "LZOpenFileA" (ByVal IpszFile As
String, IpOf As OFSTRUCT, ByVal style As Long) As Long
```

```
Private Declare Function LZCopy Lib "\windows\system\lz32" (ByVal
hfSource As Long, ByVal hfDest As Long) As Long
```

```
Private Declare Sub LZClose Lib "\windows\system\lz32" (ByVal
hfFile As Long)
```

```
Private Type OFSTRUCT
    cBytes As Integer
    fFixedDisk As Integer
    nErrCode As Long
    Reserved1 As Long
    Reserved2 As Long
    szPathName As String * 128
End Type
```

```
Private Sub Command1_Click ()
Dim kaynakh, hedefh, size As Long
Dim dl, d2
Dim ofs As OFSTRUCT, ofsh As OFSTRUCT
CommonDialog1.filename = "*.??_"
CommonDialog1.DialogTitle = "Sikistirilmis dosyayi seçin"
CommonDialog1.ShowOpen
dl = CommonDialog1.filename
CommonDialog1.filename = dl
CommonDialog1.DialogTitle = "Yeni ismi girin"
CommonDialog1.ShowSave
d2 = CommonDialog1.filename
'Sikistirilmis dosyayi aç
kaynakh = LZOpenFile(dl, ofs, 0 Or &H2000)
1 Hedef dosyayi olustur
hedefh = LzOpenFile(d2, ofsh, &H1000)
'Kaynak dosyayi hedef dosyaya aç
size = LZCopy(kaynakh, hedefh)
Print "Toplam "; size; "byte"
'Dosyalari kapat
LZClose (kaynakh)
LZClose (hedefh)
End Sub
```

Ses kartinin API'lerle kullanilmasi

VB ses kartınız için (Genel olarak multimedya donanımlarınız için) MCI kontrolünü sunar. Ancak bu kontrol her zaman ise yaramaz. Bu kontrol yeterince hızlı değildir. Ayrıca programınızdan sadece sesleri kullanmak için MCI kontrolü için bir sürü kod yazmanız da gerekmez. MCI kontrolünü kullanarak ses kartının a-yarlarını da yapamazsınız. İşte şimdi vereceğimiz API'lerle bütün bu olumsuzlukları asabilirsiniz.

Bu API'leri vermeden önce seslerle ilgili bu özellikleri hatırlatalım:

Ses kartlarıyla gerçek hayattaki sesler örneklenip saklanabilir. Bu işlem için ses kartının mikrofona veya line-in girişleri kullanılır. Örneklenen bu sesler daha sonra çalınabilir. Bu işlem için de hoparlör veya line-out çıkışı kullanılır.

Gerçek hayattan örneklenen sesler oldukça fazla yer kaplarlar. Örneğin piyanodan çalınan 5 dakikalık bir müziği 8 bit 22 kHz ile mono olarak örneklenirse bu ses dosyası yaklaşık olarak $22000 * 5 * 60 = 6.3$ MB yer kaplar. Halbuki, piyanodan çalınabilen ses notalarıyla oluşturulduğundan, nota bilgileri kaydedilip bu sesler dijital olarak oluşturulabilir. Bunun içinde MIDI standartları geliştirilmiştir. Aynı müziği MIDI olarak kaydederseniz 30-100 KB yer kaplayacaktır. Aynı şey her ses için geçerli değildir. Örneğin insan sesi bu yöntemlerle oluşturulamaz. Genel olarak bu tip sesler örneklenerek elde edilir ve oldukça fazla yer kaplar. Notalarla ifade edilebilen veya düzgün bir frekans genlik yapısına sahip sesler MIDI olarak kaydedilebilir. Bu da diğerine göre oldukça az yer kaplayacaktır.

Wav dosyalarını çalabilmek için sistemde ses kartının bulunması şart değildir. Windows'a kurulu bir PC Speaker sürücüsü varsa Wav dosyaları bilgisayarınızın hoparlöründen de çalınabilir. Aşağıda vereceğimiz API'lerde sistemdeki ses kartı veya Speaker sürücüsü fonksiyonları üzerinde işlem yapılmasını sağlar.

Wav dosyalarını çalmak

```
Declare Function sndplaysound Lib "winmm" Alias "sndPlaySoundA"  
(ByVal IpszSoundName As String, ByVal wFlags As Long) As Long
```

IpszSoundName: Çalınacak Wav dosyası

wFlags: Bu parametrenin alabileceği değerler şunlardır.

0: Çalınan dosya bitinceye kadar kontrol Windows'a bırakılmaz

1: Çalınan dosya artanda çalınır. Yani parça çalınırken Windows altındaki diğer programlar da çalışmaya devam eder. Bilgisayarınızın hızı parçayı çalmaya yetmiyorsa dosyanın kesintisiz olarak çalınabilmesi için bu bit 0 yapmalısınız.

8: Bir sonraki parçaya kadar sürekli çalar. Bu değeri çalacağınız parçayı fon müziği olarak kullanacaksanız verebilirsiniz. Bu değeri verdiginizde programınız çalışmayı bitirse dahi dosya tekrar-tekrar çalmaya devam edecektir. Programdan çıkarken bu değeri pasif hale getirmeniz gerekir.

Fonksiyondan dönen değer -1 (True) ise parça çalınmıştır, aksi takdirde çalina-mamistir veya çalman parça henüz bitmemistir.

```
Private Declare Function sndplaysound Lib "winmm" Alias  
"sndPlaySoundA" (ByVal IpszSoundName As String, ByVal wFlags As  
Long) As Long
```

```
Sub Form_Load()  
x = sndplaysound("music.wav", 1)  
End Sub
```

Wav dosyalarını çalacak donanım var mı?

```
Declare Function waveOutGetNumDevs Lib "winmm" () As Long
```

Sistemde Wav dosyalarını çalacak bir donanım olup olmadığını veya varsa kaç tane olduğunu waveOutGetNumDevs API'si ile öğrenebiliriz.

Bu fonksiyondan dönen değer 0 ise sistemde Wav dosyalarını çalacak bir donanım (veya windowsa tanıtılmış sürücüsü) yoktur.

```
Private Declare Function waveOutGetNumDevs Lib "winmm" () As Long  
Sub Form_Load() If WaveOutGetNumDevs() = 0 Then  
MsgBox("Sistemimde sesleri çalacak donanım yok") Else  
MsgBox ("Sisteminizde " & waveOutGetNumDevs() & " tane ses  
sürücüsü var")  
End If End Sub
```

Wav dosyalarını çalacak donanımın kapasitesi nedir?

```
Declare Function waveOutGetDevCaps Lib "winmm" Alias  
"waveOutGetDevCapsA" (ByVal uDeviceid As Long, IpCaps As  
WAVEOUTCAPS, ByVal uSize As Long) As Long
```

Sistemde bulunan ses donanımının stereo mu, mono mu olduğunu, ses ayarı yapılıp yapılamadığı gibi bazı özellikleri waveOutGetDevCaps API'si ile öğrenilebilir.

udeviceid: Kapasitesi öğrenilecek birimin numarası. Bu parametrenin alacağı değerler 0 ile sistemde yüklü ses sürücüsü sayısının bir eksiği

Microsoft Visual Basic 6.0

arasındadır. Sisteminizde bir ses kartı ve bir sürücüsü varsa bu parametreye 0, iki tane ise ikinci sürücü için 1 gibi değerler vermelisiniz.

LpCaps: Bu parametre WAVEOUTCAPS kullanıcı tanımlı tipinden bir değişkendir. Ve ses sürücüsünün kapasitesi bu tipte tanımlanmış değişken ile geri döner.

usize: WAVEOUTCAPS tipinden tanımlanmış değişkenin uzunluğu.

Global Const MAXPNAMELEN = 32 ' Ürün isminin maksimum uzunluğu

Type WAVEOUTCAPS

```
wMid As Integer ' Üretici ID numarası
wpid As Integer ' Ürün ID numarası
vdriverversion As Long ' Sürücü versiyonu
szpname As String * MAXPNAMELEN ' Ürün ismi
dwFormats As Long ' Desteklediği formatlar
wchannels As Long ' Desteklediği kanal sayısı (stereo,mono)
dwSupport As Long ' Sürücü tarafından desteklenen
```

özellikler

End Type

WaveOutCaps yapısındaki dwFormats parametresinin alabileceği değerler ve anlamları şunlardır. Bir sürücü bunlardan bir çoğunu aynı anda destekleyebileceği için bu değerler AND işlemine tabi tutularak hangilerinin desteklendiği öğrenilebilir.

&H1:	11.025 kHz, Mono, 8-bit
&H2:	11.025 kHz, Stereo, 8-bit
&H4:	11.025 kHz, Mono, 16-bit
&H8:	11.025 kHz, Stereo, 16-bit
&H10:	22.05 kHz, Mono, 8-bit
&H20:	22.05 kHz, Stereo, 8-bit
&H40:	22.05 kHz, Mono, 16-bit
&H80:	22.05 kHz, Stereo, 8-bit
&H100:	44.1 kHz, Mono, 8-bit
&H200:	44.1 kHz, Stereo, 8-bit
&H400:	44.1 kHz, Mono, 16-bit
&H800:	44.1 kHz, Stereo, 16-bit

WaveOutCaps yapısındaki wChannels parametresinin alabileceği değerler ve anlamları şunlardır.

0	: Mono
1	: Stereo

WaveOutCaps yapısındaki dwSupport parametresinin alabileceği değerler ve anlamları şunlardır. Bir sürücü bunlardan bir çoğunu aynı anda destekleyebileceği için AND işlemine tabi tutularak hangilerinin desteklendiği öğrenilebilir.

&H1: Pitch seviyesi ayarlanabilir.
&H2: PlaybackRate ayari yapilabilir.
&H4: Ses ayari yapilabilir.
&H8: Sol ve sag hoparlörlerin ses seviyeleri ayri ayri ayarlanabilir.
 Fonksiyondan dönen degerlerin anlamlari söylerdir.

0: Hata yok

2: udeviceid parametresi ile verilen sürücü yok. Sistemdeki sürücü sayisindan daha büyük bir numara vermissiniz.

6: Sistemde yüklü sürücü yok

ÖRNEK: Örnek olarak sistemdeki ilk ses sürücüsü hakkında detayli bilgi verecek bir program yazalim.

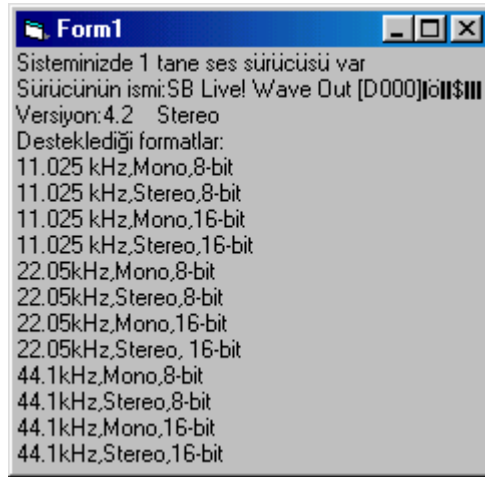
```
ÖRNEK : WaveOutGetNumDevs, WaveOutGetDevCaps
Option Explicit
Const MAXPNAMELEN = 32
Private Type WAVEOUTCAPS
wMid As Integer 'Üretici ID numarası
wPId As Integer 'Ürün ID numarası
vDriverVersion As Long 'Sürücü versiyonu
szPName As String * MAXPNAMELEN 'Ürün ismi
dwFormats As Long 'Desteklediği formatlar
wChannels As Long 'Desteklediği kanal sayısı (stereo, mono)
dwSupport As Long 'Sürücü tarafından desteklenen özellikler
End Type
Private Declare Function waveOutGetNumDevs Lib "winmm" () As Long
Private Declare Function waveOutGetDevCaps Lib "winmm" Alias
waveOutGetDevCapsA" (ByVal uDeviceId As Long, lpCaps As
WAVEOUTCAPS, ByVal uSize As Long) As Long

Private Sub Form_Load ()

Dim Ipc As WAVEOUTCAPS Show If
waveOutGetNumDevs () = 0 Then
Print "Sisteminde sesleri çalışacak donanım yok"
Exit Sub
End If
Print "Sisteminizde " & waveOutGetNumDevs() & " tane ses
sürücüsü var"
Call waveOutGetDevCaps(0, Ipc, Len(lpc))
Print "Sürücünün ismi:"; Ipc.szPName
Print "Versiyon:" & (Ipc.vDriverVersion And &HFF00) \ 256 & " . " &
(Ipc.vDriverVersion And &HFF), If Ipc.wChannels = 1 Then
Print "Mono" Else
Print "Stereo" End
If
Print "Desteklediği formatlar:"
If Ipc.dwFormats And 1 Then Print "11.025 kHz, Mono, 8-bit"
```

Microsoft Visual Basic 6.0

```
If Ipc.dwFormats And 2 Then Print "11.025 kHz,Stereo,8-bit"  
If Ipc.dwFormats And 4 Then Print "11.025 kHz,Mono,16-bit" If  
Ipc.dwFormats And 8 Then Print "11.025 kHz,Stereo,16-bit" If  
Ipc.dwFormats And &H10 Then Print "22.05kHz,Mono,8-bit"  
If Ipc.dwFormats And &H20 Then Print "22.05kHz,Stereo,8-bit"  
If Ipc.dwFormats And &H40 Then Print "22.05kHz,Mono,16-bit"  
If Ipc.dwFormats And &H80 Then Print "22.05kHz,Stereo, 16-bit"  
If Ipc.dwFormats And &H100 Then Print "44.1kHz,Mono,8-bit"  
If Ipc.dwFormats And &H200 Then Print "44.1kHz,Stereo,8-bit"  
If Ipc.dwFormats And &H400 Then Print "44.1kHz,Mono,16-bit"  
If Ipc.dwFormats And &H800 Then Print "44.1kHz,Stereo,16-bit"  
  
End Sub
```



Bu örnekte ilk ses sürücüsü hakkında bilgi aldık. Sisteminizde birden fazla ses çıkışı için kullanılabilir birim olabilir. Genelde modemlerin de ses çıkış sürücüleridir. Bunlar hakkında da bilgi almak için aşağıdaki satırdaki 0 sayısını 1,2,3 şeklinde değiştirebilirsiniz.

```
Call waveOutGetDevCaps(0, Ipc, Len(Ipc))
```

Ses ayarı yapmak

Ses donanımınız yazılımla ses ayarı yapılmasına olanak tanıyorsa sol ve sağ ses seviyelerini öğrenmek ve ayarlamak WaveOutGetVolume ve WaveOutSetVolume API'leriyle yapılabilir.

```
'Ses seviyesini öğrenmek için  
Declare Function waveOutGetVolume Lib "winmm" (ByVal udeviceid  
As Long, lpdwvolume As Long) As Long  
'Ses seviyesini değiştirmek için  
Declare Function waveOutSetVolume Lib "winmm" (ByVal udeviceid
```

As Long, ByVal dwVolume As Long) As Long

Udeviceid: Ses seviyesi ayarlanacak veya öğrenilecek sürücünün numarası.

Lpdwvolume: Ses seviyesi bu 32 bitlik degisken ile ayarlanır. Bu parametrenin yüksek seviyeli 2 bayti sag hoparlörün ses seviyesini, düşük seviyeli 2 bayti sol hoparlörün ses seviyesini ayarlar. Bu degerlerin &HFFFF olmasi sesin en yüksek seviyesini 0 olmasi en düşük seviyesini gösterir. Eger ses karti her iki hoparlörün ses ayarini ayri ayri yapamiyorsa yüksek seviyeli 2 byte dikkate alınmaz, düşük seviyeli 2 byte her iki hoparlörün de çıkis seviyesini ayarlar.

Burada dikkat edilmesi gereken bir konu var. Lpdwvolume parametresi gerçekte 32 byte'lik isaretsiz bir uzun tamsayi tipidir. Ancak VB'de böyle bir tip olmadigi için bu parametre mecburen isaretli olarak tanımlanmistir. Yani bu sayinin 32. Biti isaret biti olarak kabul edilecektir. Burada yapmanız gereken aritmetik islemler olacaktır. Hexadesimal sayilari kullanabilirsiniz.

Öneğin iki hoparlörü de maximum seviyeye getirmek için 32 bitin tümünün 1 olması gerekir. Bunun decimal karsiligi 4294967295'dir. Bu sayiyi bir long tipine atayamazsiniz çünkü o tipin sinirlarini asar. Ancak &HFFFFFFF ataması yaparsanız problem çıkmaz. Atadığınız degiskenin degerine bakarsanız pozitif bir sayi atamis olmamıza ragmen sonucun negatif olduğunu görürsünüz. Çünkü atadığımız sayinin en yüksek seviyeli biti 1'dir ve bu VB tarafından negatiflik isareti olarak kabul edilecektir. Genelde problem &H7FFFFFFF sayisindan daha büyük sayılarda çıkar. Çünkü bu degerden daha büyük degerlerde 32. Bit 1'dir ve bu degerden daha büyük degerler negatif degermis gibi isleme girer.

ÖRNEK: Örnek olarak iki Slider kontrolü ile ses kartinin sol ve sag ses ayarlarini yapacak bir program yazalım.

ÖRNEK : WaveOutGetVolume, WaveOutSetVolume

Option Explicit

Const MAXPNAMELEN = 32

Private Type WAVEOUTCAPS

wMid As Integer ' Üretici ID numarası

wpid As Integer ' Ürün ID numarası

vdrierversion As Long ' Sürücü versiyonu

'szpname As String * MAXPNAMELEN ' Ürün ismi

dwFormats As Long ' Desteklediği formatlar

wchannels As Long ' Desteklediği kanal sayısı (stereo,mono)

dwSupport As Long ' Sürücü tarafından desteklenen özellikler

End Type

Private Declare Function waveOutGetNumDevs Lib "winmm" () As Long

Private Declare Function waveOutGetDevCaps Lib "winmm" Alias

"waveOutGetDevCapsA" (ByVal udeviceid As Long, IpCaps As

WAVEOUTCAPS, ByVal uSize As Long) As Long

Microsoft Visual Basic 6.0

```
Private Declare Function waveOutGetVolume Lib "winmm" (ByVal
uDeviceID As Long, lpdwVolume As Long) As Long
Private Declare Function waveOutSetVolume Lib "winmm" (ByVal
uDeviceID As Long, ByVal dwVolume As Long) As Long

Private Sub Form_Load()
Dim lpc As WAVEOUTCAPS
If waveOutGetNumDevs() = 0 Then
    Print "Sisteminde sesleri çalacak donanim yok"
    End
End If
Call waveOutGetDevCaps(0, lpc, Len(lpc))
If lpc.wchannels = 0 Then
    Slider2.Visible = False 'mono ise birini gizle
End If
If (lpc.dwSupport And 4) = 0 Then
    'Ses ayarini desteklemiyorsa ikisininide gizle
    Slider1.Visible = False
    Slider2.Visible = False
End If
If (lpc.dwSupport And 8) = 0 Then
    'Sol sag ses ayarini desteklemiyorsa birini gizle
    Slider2.Visible = False
End If
Slider1.Min = 0
Slider1.Max = &HFFFF&
Slider1.TickFrequency = &HFFFF& / 10
Slider2.Min = 0
Slider2.Max = &HFFFF&
Slider2.TickFrequency = &HFFFF& / 10
'Su anki seviyeyi göster
Dim x, sol, sag, st
Call waveOutGetVolume(0, x)
sol = x And &HFFFF& 'düşük seviyeli iki byte
st = Hex(x And &HFFFF0000)
If Len(st) > 4 Then
    st = Mid(st, 1, Len(st) - 4) 'Yüksek seviyeli iki bayti al
Else
    st = "0"
End If
sag = CDbl("&h" & st)
Slider1.Value = sol
Slider2.Value = sag
End Sub

Sub sesayar()
Dim x, sol, sag, s
sol = Slider1.Value
sag = Slider2.Value
s = Val("&h" & Hex(sag) & String(4 - Len(Hex(sol)), "0") &
Hex(sol) & "&")
Call waveOutSetVolume(0, s)
End Sub

Private Sub Slider1_Click()
sesayar
End Sub
```

```

Private Sub Slider2__Click()
sesayar
End Sub

Private Sub Slider1_Scroll()
sesayar
End Sub

Private Sub Slider2_Scroll()
sesayar
End Sub

```

Ses Girişi İle İlgili API'ler

Ses kartinizin ses kaydı yapma desteği varsa ve sürücüsü Windows'a kurulu ise ses kayıt ayarlarını aşağıdaki API'lerle yapabilirsiniz.

Ses kaydı yapacak donanım var mı?

```
Declare Function waveinGetNumDevs Lib "winmm" () As Long
```

Sistemde ses kaydı yapacak donanım kontrolünü ve varsa kaç tane olduğunu WaveInGetNumDevs API'si ile öğrenebiliriz.

Bu fonksiyondan dönen değer 0 ise sistemde kayıt yapacak bir donanım yok veya Windows'a tanıtılmamış olabilir.

```

Private Declare Function waveinGetNumDevs Lib "winmm" () As Long

Private Sub Form_Load()
If waveinGetNumDevs() = 0 Then
    MsgBox ("Sisteminizde ses kaydı yapacak donanım yok")
Else
    MsgBox ("Sisteminizde " & waveinGetNumDevs() & " tane ses kaydı yapacak donanım var")
End If
End Sub

```

Ses kaydı yapacak donanımın kapasitesi nedir?

```
Declare Function waveinGetDevCaps Lib "winmm" (ByVal udeviceid As Long, IpCaps As WAVEINCAPS, ByVal uSize As Long) As Long
```

Sistemde bulunan ses kayıt donanımının bazı özellikleri WaveInGetDevCaps A-PI'si ile öğrenilebilir.

udeviceid: Kapasitesi öğrenilecek birimin numarası. Bu parametrenin alacağı değerler 0 ile sistemde yüklü ses sürücüsü sayısının bir eksigi arasındadır.

uSize: WAVEINCAPS tipinden tanımlanmış değişkenin uzunluğu.

Microsoft Visual Basic 6.0

LpCaps: Bu parametre WAVEINCAPS kullanıcı tanımlı tipinden bir değişkendir. Ses kayıt sürücüsünün kapasitesi bu tipte tanımlanmış değişken ile geri döner. WAVEINCAPS yapısını general-declarations kısmında şöyle tanımlamalısınız.

Global Const MAXNAMELEN=32 'Ürün isminin maximum uzunluğu

```
Type WAVEINCAPS
    wMid As Integer 'Üretici ID Numarası
    wpid As Integer 'Ürün ID Numarası
    vdriverversion As Long 'Sürücü versiyonu
    szpname As String 'Ürün ismi
    dwFormats As Long 'Desteklediği formatlar
    wchannels As Long 'Desteklediği formatlar kanal
sayisi(stero,mono)
End Type
```

Yapı içindeki wMid ve Wpid alt değişkenleri sürücünün üretici ve ürün numaralarıdır. Programın çalışması açısından bir önemli değildir. Ancak özel bir ses kartının kendine has bazı özelliklerini kullanacaksanız bu değerlerle kartı tanıyabilirsiniz. Vdriverversion, szpname, dwFormats ve Wchannels parametrelerinin anlamları WAVEOUTCAPS yapısında olduğu gibidir.

Fonksiyondan dönen değerlerin anlamları şöyledir.

0: Hata yok

2: udeviceid parametresi ile verilen sürücü yok. Sistemdeki sürücü sayısından daha büyük bir numara vermissiniz.

6: Sistemde yüklü sürücü yok

MIDI Çıkışı İle İlgili API'ler

Eğer ses kartınızın MIDI desteği varsa ve gerekli sürücü Windows'a kurulu ise MIDI çıkış ayarlarını aşağıdaki API'lerle yapabilirsiniz.

MIDI formatındaki dosyaları çalışacak donanım var mı?

```
Declare Function MidiOutGetNumDevs Lib "winmm" () As Long
```

Sistemde MIDI çıkış donanımı olup olmadığını veya varsa kaç tane olduğunu MidiOutGetNumDevs API'si ile öğrenebiliriz.

MIDI çıkış kapasitesi nedir?


```
Declare Function midiOutGetDevCaps Lib "winmm" (ByVal udeviceid As Long, IpCaps As MIDIOUTCAPS, ByVal uSize As Long) As Long
```

Sistemde bulunan MIDI çıkış donanımının bazı özellikler MidiOutGetDevCaps API'si ile öğrenilebilir.

Udeviceid :Kapasitesi öğrenilecek birimin numarası. Bu parametrenin alacağı değerler 0 ile sistemde yüklü ses sürücüsü sayısının bir eksiği arasındadır.

Usize :MIDIOUTCAPS tipinden tanımlanmış değişkenin uzunluğu.

LpCaps :Bu parametre MIDIOUTCAPS kullanıcı tanımlı tipinden bir değişkendir. Ve ses kayıt sürücüsünün kapasitesi bu tipte tanımlanmış değişken ile geri döner, MIDIOUTCAPS yapısını general-declarations kısmında şöyle tanımlamalısınız.

```
Type MIDIOUTCAPS
```

```
    wMid As Integer 'Üretici ID numarası
```

```
    wPid As Integer 'Ürün ID numarası
```

```
    vDriverVersion As Long 'Sürücü versiyonu
```

```
    szPname As String * MAKPNAMLEN 'Ürün ismi
```

```
    wTechnology As Long 'Sesleri oluşturmak için kullanılan yöntem
```

```
    wVoices As Integer 'Ses sayısı
```

```
    wNotes As Long 'Nota sayısı
```

```
    wChannelMask As Long 'Kullanılan kanallar
```

```
    dwSupport As Long 'Desteklenen fonksiyonlar
```

```
End Type
```

Yapı içindeki wMid, wPid, vDriverVersion ve szPname parametrelerinin anlamları WAVEOUTCAPS yapısında olduğu gibidir.

wVoices :Desteklenen ses sayısı bu değişkende geri döner.

wNotes :Desteklenen nota sayısı bu değişkende geri döner.

dwSupport :Bu değişkenden dönen değer ilk üç bitinin anlamı vardır. Bu bitler ve anlamları şöyledir.

1:Ses ayarı yapılabilir

2: Sol-sag ses ayarı ayrı ayrı yapılabilir.

4: Patch-Caching desteği var.

wTechnology :Bu değişkenden dönen değer alacağı değerler ve anlamları şöyledir.

1: MIDI dosyaları için çıkış portu var

3: Sesler kare dalga sentezi yöntemiyle oluşturuluyor.

4: Sesler FM sentezi ile oluşturuluyor.

5: MIDI haritalayıcısı yöntemiyle oluşturuluyor.

Bu degiskenden dönen degerin 1 olmasi seslerin özel bir donanim tarafından olusturulduğunu gösterir ve bu durumda diger 4 parametrenin (WVoices, WNotes, WChannel, dWSupport) bir anlami yoktur.

Fonksiyondan dönen degerlerin anlamlari

söyledir.

0: Hata yok

2: udeviceid parametresi ile verilen sürücü yok. Sistemdeki sürücü sayisindan daha büyük bir numara vermissiniz.

6: Sistemde yüklü sürücü yok

MIDI çıkis ses seviyesini öğrenmek ve ayarlamak

Eger MIDI donanimi yazilimla ses ayari yapılmasına imkan taniyorsa sol ve sag ses seviyelerini öğrenmek ve ayarlamak MidiOutGetVolume ve MidiOutSetVolume API'leriyle yapılabilir.

Ses seviyesini öğrenmek için:

```
Declare Function MidiOutGetVolume Lib "winmm" (ByVal udeviceid As Long, lpdwvolume As Long) As Long
```

Ses seviyesini degistirmek için:

```
Declare Function MidiOutsetVolume Lib "winmm" (ByVal udeviceid As Long, ByVal dwVolume As Long) As Long
```

Bu iki API'nin kullanimi ses çıkis seviyesi için verdigimiz WaveOutSetVolume ve WaveOutGetVolume API'leriyle ayni.

MIDI giris donanimi var mi?

```
Declare Function MidiinGetNurnDevs Lib "winmm" () As Long
```

Bu fonksiyondan dönen deger 0 ise sistemde MIDI kaydi yapacak bir donanim (veya windowsa tanitilmis sürücüsü) yoktur.

```
Declare Function MidiinGetNurnDevs Lib "winmm" () As Long
```

```
Private Sub Form_Load()  
If MidiinGetNurnDevs() = 0 Then  
    MsgBox ("Sisteminde MIDI giris donanimi yok")  
End If  
End Sub
```

Ses kartinin sundugu diger sundugu fonksiyonlarla ilgili API'ler

Ses kartiniz ses ve midi dosyalarinin ayarlan haricinde kendine özgü diger bazi ayarlarin yapilmasini da sagliyor olabilir. Ses kartinizdan; bir CD'nin veya speaker'in ses ayarlarini bu özellikleri kullanarak yapabilirsiniz. Ses kartinizin sagladigi bu diger fonksiyonlari Auxlary olarak adlandirilan API'lerle kullanabilirsiniz.

Yardimci Fonksiyonlari Sayisi?

```
Declare Function auxGetNumDevs Lib "winmm" () As Long
```

Bu fonksiyondan dönen deger 0 ise donanimin sundugu yardimci islemler yoktur. Aksi takdirde dönen deger desteklenen yardima islemlerin sayisini verir.

Desteklenen Yardimci Fonksiyonlari sitesi nedir?

```
Declare Function auxGetDevCaps Lib "winmm" (ByVal udeviceid As Long, IpCaps As AUXCAPS, ByVal uSize As Long) As Long
```

Udeviceid :Kapasitesi öğrenilecek yardimci birimin numarası, İlk birim için bu numaranin degeri 0 dir.

LpCaps :Bu parametre AUXCAPS kullanıcı tanımlı tipinden bir degiskendir, AUXCAPS yapisini general-declarations kısmında söyle tanımlamalısınız.

```
Global Const MAXPNAMELEN=32
```

```
Type AUXCAPS
```

```
    wMid As Integer 'Üretici ID numarası
```

```
    wpid As Integer 'Ürün ID numarası
```

```
    vdriverversion As Long ' Sürücü versiyonu
```

```
    szpname As String * MAXPNAMELEN 'Ürün ismi
```

```
    wTechnology As Long
```

```
    dwSupport As Long 'Desteklenen fonksiyonlar
```

```
End Type
```

Yapi içindeki wMid, wpid, Vdriverversion ve szpname parametrelerinin anlamları WAVEOUTCAPS yapısında olduğu gibidir.

DWSupport: Bu degiskenden dönen degerin ilk iki bitinin anlamı vardır.

Bu bitler ve anlamları şöyledir.

1: Ses ayarı yapılabiliyor

2: Sol-sag ses ayarı ayrı ayrı yapılabiliyor.

Fonksiyondan dönen degerlerin anlamları şöyledir.

0: Hata yok

2: udeviceid parametresi ile verilen sürücü yok. Sistemdeki sürücü sayısından daha büyük bir numara vermissiniz.

6: Sistemde yüklü sürücü yok

Yardımcı birimlerin ses seviyesini öğrenmek ve ayarlamak

Eğer desteklenen yardımcı birimin ses ayarı yazılımla yapılabiliyorsa bu birimin ses seviyesini öğrenmek ve ayarlamak AuxGetVolume ve AuxSetVolume API'leriyle yapılabilir.

Ses seviyesini öğrenmek için:

```
Declare Function AuxGetVolume Lib "winmm" (ByVal udeviceid As Long, lpdwvolume As Long) As Long
```

Ses seviyesini değiştirmek için:

```
Declare Function AuxSetVolume Lib "winmm" (ByVal udeviceid As Long, ByVal dwVolume As Long) As Long
```

Bu iki API'ninde kullanımı ses çıkış seviyesi için verdiğimiz WaveOutSetVolume ve WaveOutGetVolume API'leriyle aynıdır.

Resim İşleme

VB'de resimleri göstermek için Picture ve Image kontrolleri, parçalara ayırarak işlemek içinse PicClip nesnesi bulunmaktadır. Örneğin bir resmin negatif görüntüsünü oluşturmak veya bir resim nesnesi üzerinde birden fazla resmi aynı anda göstermek için PaintPicture metodu vardır. Bu metod aslında BitBlt Apisinin Vb'deki halidir. BitBlt Apisini kullanarak sadece kendi formunuzda değil, ekranın tamamında işlemler yapabilirsiniz.

```
Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal XSrc As Long, ByVal YSrc As Long, ByVal dwRop As Long) As Long
```

Parametrelerin anlamları ve alabilecekleri değerler ise şöyledir:

hSrcDc : İşlenecek resmin bulunduğu kontrolün hDC numarası.

hDestDc : İşlenen resmin gösterileceği kontrolün hDC numarası

X, Y : Resmin gösterileceği kontrol içindeki koordinatlar.

nWidth, nHeight:İslenecek resmin genişliği ve yüksekliği.

XSrc,YSrc : İslenecek resmin bulunduğu kontrol içindeki resim parçasının başlangıç koordinatları.

dwRop : Resim kaynaktan hedefe kopyalanırken uygulanacak işlem.

Bu işlemlerden bazıları şunlardır.

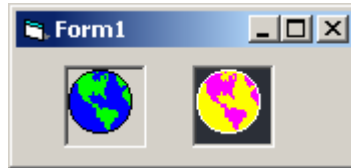
- &H550009: Hedef resmin tersi alınır.

Örneğin picture1 içindeki resim tiklanınca tersi renge girmesi için

```
Private Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal dwRop As Long) As Long
```

```
Private Sub Form_Load()  
    Picture2.Picture = Picture1.Picture  
End Sub
```

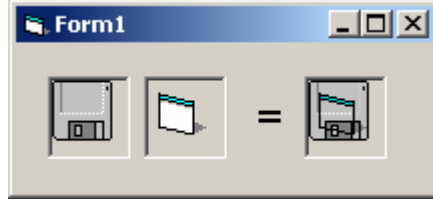
```
Private Sub Picture2_Click()  
    Dim t  
    Picture2.ScaleMode = 3 'Pixel  
    t = BitBlt(Picture2.hDC, 0, 0, Picture2.ScaleWidth, Picture2.ScaleHeight, Picture2.hDC, 0, 0, &H550009)  
End Sub
```



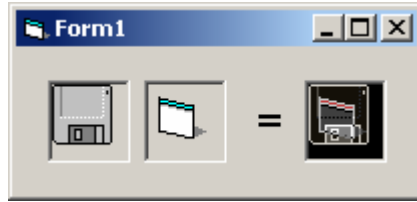
- &H330008: Kaynak resmin tersi alınarak hedefe kopyalanır.
- &H1100A6: Kaynak resim ile hedef resim OR işlemine tabi tutulur ve sonucun tersi (NOT işlemi) alınarak hedefte gösterilir.
- &H8800C6: Kaynak resim ile Hedef resim AND işlemine tabi tutularak hedefte gösterilir.

```
Sub Picture2_Click()  
    Dim t  
    Picture1.ScaleMode = 3 'Pixel  
    t = BitBlt(Picture1.hDC, 0, 0, Picture1.ScaleWidth, _  
        Picture1.ScaleHeight, Picture2.hDC, 0, 0, &H8800C6)
```

End Sub



- &HCC0020: Kaynak resim hiçbir isleme tabi tutulmadan hedefte gösterilir.
- &H660046: Kaynak resim ile hedef resim XOR işlemine tabi tutularak hedefte gösterilir.



Bu API kullanılırken kaynak resmin AutoRedraw özelliği True yapılmalıdır. Aksi takdirde kaynak resim ekranda görülmediği sürece doğru şekil üzerinde işlem yapılmayacaktır.

Bu API kullanarak resimler üzerinde yukarıdaki işlemler gerçekleştirilebilirken bir kontrol üzerinde birden çok resim göstermek içinde kullanılabilir.

ÖRNEK: Örneğin ICON gösteren bir programımız olsun. Seçilen dizindeki iconları form üzerinde gösterebilir. Bir form üzerinde en az elli tane icon gösterilebilir. Form üzerine 50 tane PictureBox yerleştirip iconları bunların içinde göstermek sistem kaynakları açısından hiçde uygun bir yöntem değildir. Bu noktada BitBlt API'sini kullanarak bir PictureBox kontrolü içinde istediğimiz kadar resmi gösterebiliriz. Örneğimiz için aşağıdaki formu oluşturun. Form üzerine biri büyükçe (Picture2) diğeri daha küçük (Picture1) iki PictureBox'da yerleştirin. Küçük olan PictureBox'un (Picture1) AutoreDraw özelliğinde True yapın.

Option Explicit

```
Public Declare Function BitBlt Lib "gdi32" _  
    (ByVal hDestDC As Long, ByVal X As Long, ByVal Y As Long, _  
    ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As  
    Long, _
```

```

ByVal xSrc As Long, ByVal ySrc As Long, ByVal dwRop As Long) As
Long

Sub Form_Load()
    Show
    Picture2.ScaleMode = 3
    Picture1.ScaleMode = 3
    Picture2.FontBold = False
    If File1.ListCount > 0 Then
        'icon varsa Click olayini meydana getir
        File1.ListIndex = 0
    End If
End Sub

Sub Drive1_Change()
    ChDrive Drive1.Drive
    Dir1.Path = Drive1.Drive
End Sub

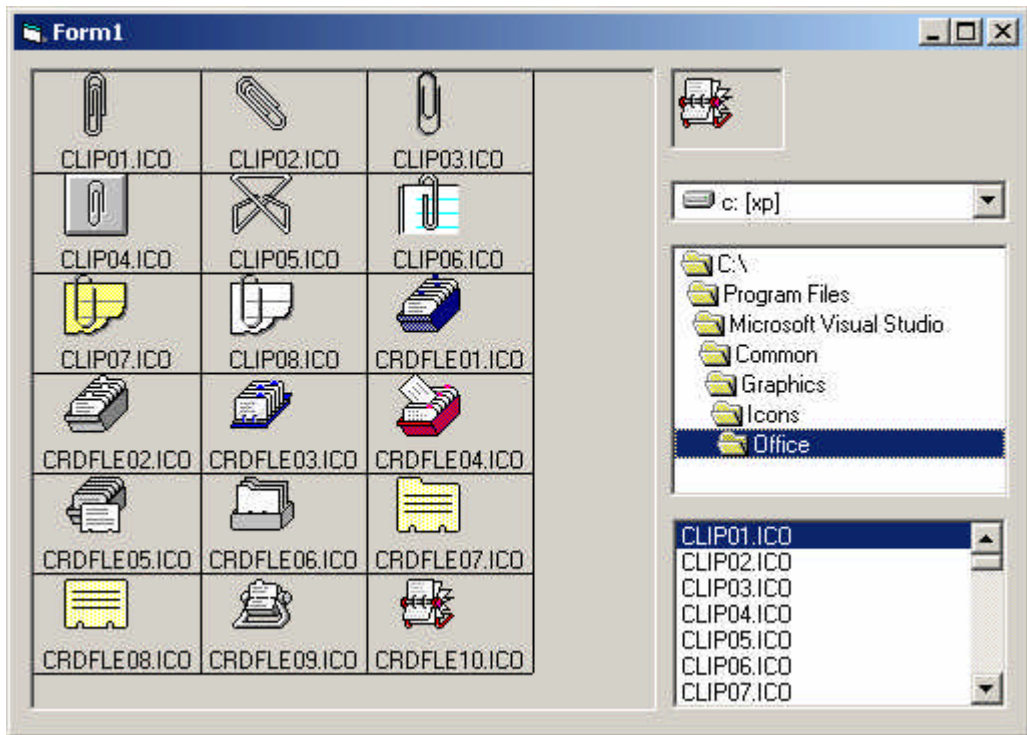
Sub Dir1_Change()
    File1.Path = Dir1.Path
    ChDir Dir1.Path
    If File1.ListCount > 0 Then
        'icon varsa Click olayini meydana getir
        File1.ListIndex = 0
    End If
End Sub

Sub File1_Click()
    On Local Error GoTo hata
    Dim iconpersutun, iconpersatir, t
    Dim X, Y, i, j, r, g, w, h, k, ha
    Static b, s, eskidizin
    If eskidizin <> File1.Path Then 'dizin degismisse
        eskidizin = File1.Path
    Else
        If File1.ListIndex >= b And File1.ListIndex <= s
Then 'Seçilen dosya ekranda halen varsa
            Picture1.Picture = LoadPicture(File1.FileName)
            Exit Sub
        End If
        w = Picture1.ScaleWidth
        h = Picture1.ScaleHeight
        iconpersutun = Int(Picture2.ScaleWidth / (w + 30))
        iconpersatir = Int(Picture2.ScaleHeight / (h +
Picture2.TextHeight("X")))
        b = File1.TopIndex
        s = iconpersutun * iconpersatir + b - 1
        If s > File1.ListCount - 1 Then
            s = File1.ListCount - 1
        End If
        Picture2.Cls
        X = 15

```

Microsoft Visual Basic 6.0

```
Y = 0
k = b
For i = b To s
    Picture1.Picture = LoadPicture(File1.List(i))
    If ((i - b) Mod iconpersutun) = 0 And i <> b Then
        X = 15
        k = i
        Y = Y + Picture1.ScaleHeight +
Picture2.TextHeight("X")
        End If
        X = (i - k) * (w + 30) + 15
        If ha = 1 Then 'hatali resim
            ha = 0
            Picture2.CurrentX = X
            Picture2.CurrentY = Y + h / 2
            Picture2.Print "HATALI"
        Else
            t = BitBlt(Picture2.hDC, X, Y, w, h,
Picture1.hDC, 0, 0, &HCC0020)
            End If
            Picture2.CurrentX = X - 15 + ((w + 30) -
Picture2.TextWidth(File1.List(i))) / 2 'ortayi bul
            Picture2.CurrentY = Y + Picture1.ScaleHeight
            Picture2.Print File1.List(i)
            DoEvents
        Next
        X = 0
        For i = 0 To Picture2.ScaleWidth
            X = X + w + 30
            Picture2.Line (X, 0)-(X, (h +
Picture2.TextHeight("X")) * iconpersatir)
        Next
        Y = 0
        For i = 0 To Picture2.ScaleHeight
            Y = Y + h + Picture2.TextHeight("X")
            Picture2.Line (0, Y)-((w + 30) * iconpersutun, Y)
        Next
    Exit Sub
hata:
    If Err = 481 Then ha = 1 'resim hatali
    Resume Next
End Sub
```

StretchBlt

BitBlt API'si kaynak resimdeki resimin bir parçasını üzerinde işlem yaparak hedef nesneye aynı boyutta kopyalayabilir. Resim üzerinde büyültme veya küçültme yapamaz. VB'de Image nesnesinin Stretch özelliği ile resmi kontrolün boyutlarına sigdırabiliyorduk. Ancak bu resim üzerinde işlem yapamayacağımız için üzerinde küçülterek veya büyülterek işlem yapacağımız resimler için StretchBlt API'sini kullanabiliriz.

```
Public Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As Long
        nSrcWidth, nSrcHeight :Resim parçasının orjinal boyutlari
```

nWidth, nHeight : Resim parçasının hedefteki yeni boyutları

Diğer parametreler BitBlt API'sinde olduğu gibidir.

ÖRNEK: Formun PrintForm özelliği ile formunuzu yazıcıdan çıkardığınızda tam ekran bir formun yazıcıdan çok küçük çıktığını göreceksiniz, bunun sebebi yazıcı çözünürlüğünün ekran çözünürlüğünden çok büyük olmasıdır. Tam ekran bir formu yazıcıdan tam sayfa olarak çıkarmak için bu API'den faydalanabiliriz.

Formun üzerindeki görüntüyü PrintScreen tusu ile alabilirsiniz. Daha sonra bunu bir PictureBox içinde göstererek yazıcıya gönderebiliriz.

Örneğimiz için programınıza yeni bir form daha ekleyin. İlk forma istediğiniz şeyleri koyduktan sonra bir komut düğmesi daha yerleştirin. Ve ikinci forma AutoReDraw özelliği true olan bir PictureBox yerleştirin.

ÖRNEK : StretchBlt

```
'Birinci form
Sub Form_Load()
    Form1.WindowState = 2 'maximize
    Form1.Print "Formu yazıcıdan çıkarmak için PrintScreen tusuna
basınız ve komut düğmesini tıklayınız"
End Sub
Sub Command1_Click()
    Form2.Show
End Sub
```

```
'İkinci Form
Private Declare Function StretchBlt Lib "GDI32" (ByVal hDestDC As
Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long,
ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal nSrcWidth As
Long, ByVal nSrcHeight As Long, ByVal XSrc As Long, ByVal YSrc As
Long, ByVal dwRop As Long) As Long
```

```
Sub Form_Load()
    Show
    Dim X
    Printer.ScaleMode = 3 'Pixel
    Picture1.Picture = Clipboard.GetData()
    DoEvents
    Printer.Line (0, 0)-(0, 0) 'Yazdırma işleminin başlaması için
    X = StretchBlt(Printer.hdc, 0, 0, Printer.ScaleWidth,
Printer.ScaleHeight, Picture1.hdc, 0, 0, Picture1.ScaleWidth,
Picture1.ScaleHeight, &HCC0020)
    Printer.EndDoc
    Unload Me
End Sub
```

PrintScreen tusunu SendKeys komutu ile gönderemedigimiz için PrintScreen tusuna kullanıcının basması gerekiyor.

Ekran görüntüsünü API kullanarak almak mümkündür. GetDC, GetDesktopHwnd ve BitBlt apileri ile ekran görüntüsü alınabilir.

Ekran Yakalama

VB bütün nesneler için bir hDC numarası sunmaz. VB'de Form, Common Dialog, PictureBox ve Printer nesneleri için hDC özelliği ile bu numara öğrenilebilir. Ancak diğer nesneler için o nesnenin handle numarası kullanılarak GetDC API'si ile hDC numarası öğrenilebilir (hDC numarası alınabilir demek daha doğru bir terim. Formun hDC özelliğinde de anlattığımız gibi bu numara sürekli değişen bir numaradır), Böylece ListBox, TextBox, Label gibi nesnelerde resim ekleyebilirsiniz.

Nesnelerin hDC numarasını öğrenmek

```
Declare Function getdc Lib "User32" (ByVal hWnd As Long) As Long
```

- hWnd : hDC numarası alınacak kontrolün handle numarası.

Fonksiyondan geri dönen değer kontrole verilen hDC numarasıdır. Bu değer 0 ise işlem başarısızdır (Handle numarası verilen kontrol mevcut olmayabilir).

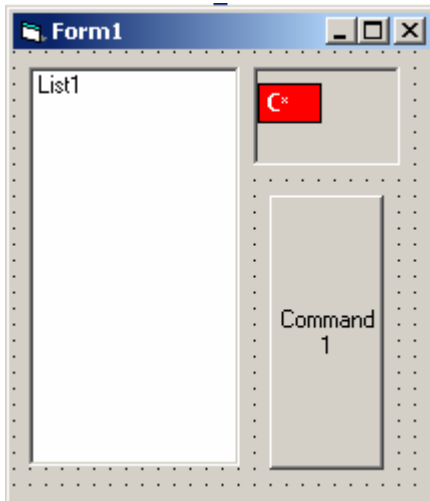
Bu fonksiyonla hDC numarası alınıp işlendikten sonra diğer uygulamalar tarafından kullanılabilmesi için ReleaseDC API'si ile serbest bırakılmalıdır.

```
Declare Function releasedc Lib "User32" (ByVal hWnd As Long, ByVal hDC As Long) As Long
```

- hWnd, hDC : Nesnenin handle ve hDC numaraları.

Geri dönen değer 1 ise işlem başarılıdır.

ÖRNEK: Örnek olarak listbox'a resim ekleyecek bir program yapalım. Örneğimiz için aşağıdaki formu oluşturun.



Option Explicit

```
Public Declare Function GetDC Lib "user32" (ByVal hwnd As Long) As Long
Public Declare Function ReleaseDC Lib "user32" (ByVal hwnd As Long, ByVal hdc As Long) As Long
Public Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As Long
```

```
Sub Form_Load()
    Dim i
    For i = 1 To 15
        List1.AddItem Space(5) & i
    Next
End Sub

Sub Command1_Click()
    Dim t, hd, i, w, h, w1, h1
    Picture1.ScaleMode = 3 'pixel
    w = Picture1.ScaleWidth
    h = Picture1.ScaleHeight
    Form1.ScaleMode = 3 'pixel
    Form1.FontSize = List1.FontSize
    h1 = Form1.TextHeight("X")
    w1 = h1 * h / w
    For i = 0 To List1.ListCount - 1
        hd = GetDC(List1.hwnd)
        t = StretchBlt(hd, 0, i * h1, w1, h1, Picture1.hdc, 0, 0, w, h, &HCC0020) 'srcCopy
        t = ReleaseDC(List1.hwnd, hd)
    Next
End Sub
```

Dikkat ederseniz örnekte listbox içindeki her elemanın yüksekliğini bulabilmek için formun TextHeight özelliğini kullandık. Bunun için önce formun fontsize özelliğini Üstenin fontsize özelliğine atadık ve formun TextHeight özelliği vasıtasıyla bu font büyüklüğündeki bir harfin yüksekliğini pixel olarak bulduk.

Örnekteki resimler yapay yollarla oluşturulduğu için Formun Paint olayında ve Listenin Click olayında resimleri tekrar gösterecek kodları da yazmanız gerekir.

Bu yöntemi kullanarak istediğiniz bütün nesnelere resimler ekleyebilirsiniz

Mouse ile seçmek

Fare ile bir resmi seçip tasımak için gerekli işlemler bir resim işleme programında bulunması gereken özelliklerdir. DrawFocusRect API'si ile bir alanı seçmek, InvertRect API'si ile de seçili kısmı ters çevirmek mümkündür. Ayrıca seçili kısmı tasımak için de BitBlt kullanılabilir.

```
Public Declare Function DrawFocusRect Lib "user32" Alias
"DrawFocusRect" (ByVal hdc As Long, lpRect As RECT) As Long
    Hdc                : Seçilecek kısmın bulunduğu kontrolün HDC numarası.

    lpRect              : Seçilecek kısmın koordinatları aşağıdaki yapı ile
                        tanımlanmış bu parametre ile belirlenir.

Type RECT
    Left As Long
    Top As Long
    Right As Long
    Bottom As Long
End Type
```

DrawFocusRect ile seçim yapıldıktan sonra, seçimin kaldırılması için tekrar aynı API çağrılmalıdır.

```
Public Declare Function InvertRect Lib "user32" Alias "InvertRect"
(ByVal hdc As Long, lpRect As RECT) As Long
```

lpRect parametresi ile belirlenmiş alanın ters renkte gösterilmesi için bu API kullanılır. İkinci kez çağrıldığında belirlenen alan orijinal renklerine döner.

ÖRNEK: Örnek olarak farenin sol tuşu ile seçip, sağ tuşu ile tasıma yapılabilecek bir program yazalım. Örneğimiz için formun Picture özelliği ile bir resim yükleyin ve aşağıdaki kodu yazın.

```
Option Explicit
Private Type RECT
    Left As Long
    Top As Long
    Right As Long
    Bottom As Long
End Type
```

Microsoft Visual Basic 6.0

```
Private Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long,
ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight
As Long, ByVal hSrcDC As Long, ByVal XSrc As Long, ByVal YSrc As
Long, ByVal dwRop As Long) As Long
Private Declare Function DrawFocusRect Lib "User32" (ByVal hdc As
Long, IpRect As RECT) As Long
Private Declare Function InvertRect Lib "User32" (ByVal hdc As Long,
IpRect As RECT) As Long
Dim MouseRect As RECT

Private Sub Form_Load()
    ScaleMode = 3 'Pixel
End Sub

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
    'Daha önce seçili olani kaldır.
    InvertRect hdc, MouseRect
    DrawFocusRect hdc, MouseRect
    If Button And 1 Then 'sol tus basildi ise
        MouseRect.Left = X
        MouseRect.Top = Y
        MouseRect.Right = X
        MouseRect.Bottom = Y
    End If
End Sub

Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As
Single, Y As Single)
    If Not (Button And 3) Then 'Sol veya sag tus bırakildi ise
        MouseRect.Right = X
        MouseRect.Bottom = Y
        InvertRect hdc, MouseRect
    End If
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As
Single, Y As Single)
    Static sonx, sony
    'Sag tus ile tasi
    If Button And 2 Then
        'Üzerinden geçtiğin yeri orjinal hale çevir.
        Call BitBlt(hdc, sonx, sony, MouseRect.Right -
MouseRect.Left, MouseRect.Bottom - MouseRect.Top, hdc,
MouseRect.Left, MouseRect.Top, &H660046) 'XOR islemine tabi tut
        sonx = X
        sony = Y
    End If
End Sub
```

```

        Call BitBlt(hdc, X, Y, MouseRect.Right - MouseRect.Left,
MouseRect.Bottom - MouseRect.Top, hdc, MouseRect.Left, MouseRect.Top,
&H660046) 'XOR
    End If
    If Button And 1 Then 'Sol tus ile seç
        DrawFocusRect hdc, MouseRect
        MouseRect.Right = X
        MouseRect.Bottom = Y
        DrawFocusRect hdc, MouseRect
    End If
End Sub

```

Program Dosyalarından ICON alma

Program dosyaları (EXE, DLL vb) içinde bir veya daha fazla sayıda ikonlar bulunabilir. VB ancak ICO uzantılı resim dosyalarındaki ikon dosyalarını alabilir. Ancak ExtractIcon apisi ile istediğiniz bir program dosyasındaki ikonları alabilir ve DrawIcon apisi ile de herhangi bir resim kontrolü içinde gösterebilirsiniz.

```

Public Declare Function ExtractIcon Lib "shell32.dll" Alias
"ExtractIconA" (ByVal hInst As Long, ByVal lpszExeFileName As String,
ByVal nIndex As Long) As Long

```

- HInst : Formunuzun handle numarasını.
- lpszExeFileName : İkonlarını almak istediğiniz dosyanın yolunu ve adını.
- nIndex : Almak istediğiniz ikonun numarasını verin.

Bir dosyada birden fazla ikon bulunabilir. İlk ikonun numarası 0'dır. Eğer numarasını verdiğiniz ikon dosyada yoksa geriye 0 değeri döner. Diğer durumda ise geriye dönen değer ikonun handle numarasıdır. Bu numarayı DrawIcon apisinde kullanarak o ikonu istediğiniz yere çizdirebilirsiniz.

```

Public Declare Function DrawIcon Lib "user32" Alias "DrawIcon"
(ByVal hdc As Long, ByVal x As Long, ByVal y As Long, ByVal hIcon
As Long) As Long

```

- Hdc : İkonu çizdireceğiniz kontrolün (form, picturebox vb) hdc numarasını.
- x ve y : Koordinatlarını (pixel olarak).
- hIcon : Çizdirmek istediğiniz ikonun handle numarasını girin.

ÖRNEK: VB6.EXE dosyasındaki ikonları bir picturebox içinde göstermek için aşağıdaki kodu kullanabiliriz.

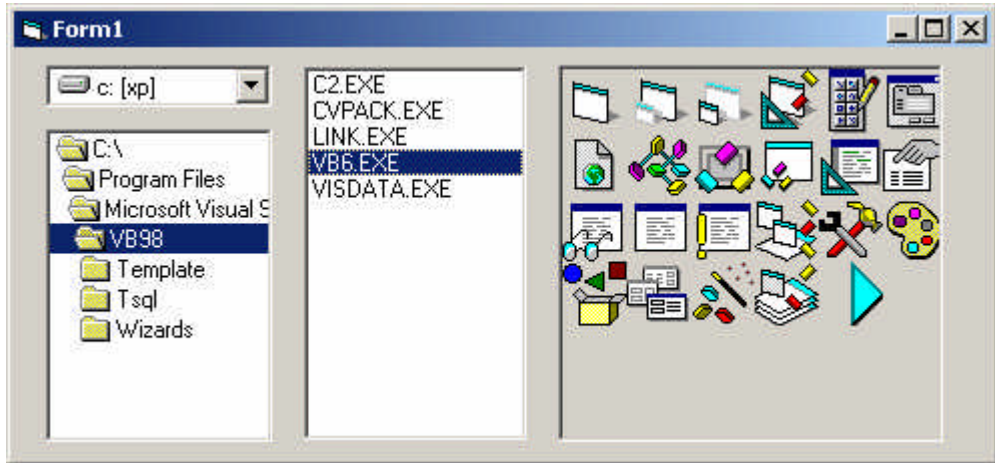
```

Option Explicit
Public Declare Function DrawIcon Lib "user32" (ByVal hdc As Long,
ByVal x As Long, ByVal y As Long, ByVal hIcon As Long) As Long
Public Declare Function ExtractIcon Lib "shell32.dll" Alias
"ExtractIconA" (ByVal hInst As Long, ByVal lpszExeFileName As
String, ByVal nIndex As Long) As Long

```

Microsoft Visual Basic 6.0

```
Private Sub Dir1_Change()  
    ChDir Dir1.Path  
    File1.Path = Dir1.Path  
End Sub  
  
Private Sub Drive1_Change()  
    ChDrive Drive1.Drive  
    Dir1.Path = Drive1.Drive  
End Sub  
  
Private Sub File1_Click()  
    Dim ih, dosya As String, i, y, x  
    If Left(File1.Path, 1) = "\" Then  
        dosya = File1.Path & File1.FileName  
    Else  
        dosya = File1.Path & "\" & File1.FileName  
    End If  
    Picture1.Picture = LoadPicture() 'önceki resimleri sil  
    Do  
        i = i + 1  
        ih = ExtractIcon(hWnd, dosya, i)  
        ih = DrawIcon(Picture1.hdc, x, y, ih)  
        x = x + 32 'her ikon 32 pixeldir  
        'eger bir satir dolarsa alt satira geç  
        If x > Picture1.ScaleWidth Then  
            x = 0  
            y = y + 32  
        End If  
    Loop While ih <> 0  
End Sub  
  
Private Sub Form_Load()  
    File1.Pattern = "*.exe;*.dil"  
    Picture1.ScaleMode = 3 'pixel  
    ScaleMode = 3 'pixel  
    'ikonlari tam sigmasi için picture1 boyutunu 32 nin kati yap  
    Picture1.Width = Picture1.Width - (Picture1.Width Mod 32)  
End Sub
```

DİĞER API'LER

Listelere Yatay Kaydırma Çubuğu

Liste kutusundan türetilmiş kontrollere (ListBox, FileListBox, DirectoryListBox) dikey kaydırma çubuğu kendiliğinden eklenir ancak bu kontrollere yatay kaydırma çubuğu eklemek için VB bir yöntem sağlamaz. SendMessage API'si ile ListBox'lara yatay scroll bar eklemek mümkündür.

```
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA"
    (ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Long,
    ByVal wMsg As Long) As Long
```

- hwnd : Mesajın gönderileceği kontrolün handle numarası
- wParam : Gönderilecek mesajın numarası. ListBoxlar için bu parametrenin &H415 olması yatay scrollbar ekleneceğini belirtir.
- lParam : Mesajla ilgili ilk parametre. Yatay kaydırma çubuğu için bu parametre kaydırma çubuğunun uzunluğunu belirler. O verilirse yatay kaydırma çubuğu varsa kaldırılır.
- lParam : Mesajla ilgili ikinci parametre.

ÖRNEK: Örnek için form üzerine bir ListBox koyun ve aşağıdaki kodu yazın.

```
'SendMessage
```

```
Option Explicit
```

```
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA"
    (ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Long,
    ByVal wMsg As Long) As Long
```

```
Private Sub Form_Load()  
    Dim x, uz  
    uz = 2 * List1.Width / Screen.TwipsPerPixelX  
    x = SendMessage(List1.hwnd, &H415, uz, 0)  
End Sub
```

Örnekte eklenen kaydırma çubuğunun boyu listboxun genişliğinin 2 katı olarak verilmiştir. Bu değer listbox içindeki elemanların uzunluğuna göre artırılıp azaltılabilir. Ancak ListBox'un genişliğine eşit bir kaydırma çubuğunun etkisi olmayacaktır.

Listede Arama

Yine SendMessage apisi aracılığıyla liste kutusundan bir metnin bulunması istenebilir. İkinci parametreye LB_FINDSTRING değeri, son parametreye de aranması istenen metin girilirse geriye elemanın listedeki numarası döner.

ÖRNEK: Kullanıcının girdiği bir elemanı listede buldurup seçmek için:

```
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA"  
(ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Long, lParam  
As Any) As Long  
Const LB_FINDSTRING = &H18F  
  
Sub Command1_Click()  
    Dim aranan  
    aranan = InputBox("Aranacak eleman")  
    List1.ListIndex = SendMessage(List1.hwnd, LB_FINDSTRING, -1,  
ByVal CStr(aranan))  
End Sub
```

ÖRNEK: Kullanıcının bir text kutusuna yazarken o elemana en yakın elemanı seçmesi için:

```
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA"  
(ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Long, lParam  
As Any) As Long  
  
Const LB_FINDSTRING = &H18F  
Private Sub Text1_Change()  
    '1 yazarken seçilmesini sağla  
    List1.ListIndex = SendMessage(List1.hwnd, LB_FINDSTRING, -1,  
ByVal CStr(Text1.Text))  
End Sub  
  
Sub Command1_Click()  
    List1.AddItem = InputBox("Eklenecek eleman")  
End Sub
```

Kendiliginden Açılan ComboBox

Bir ComboBox aktif olduğunda kendiliginden açılmasını sağlayabilirsiniz. Bu işlem için ComboBox'un handle numarasını kullanarak o kontrole CB_SHOWDROPDOWN mesajını SendMessage API'si ile göndermeniz yeterlidir.

```
Option Explicit
```

```
Public Declare Function SendMessage Lib "user32" Alias "SendMessageA"  
(ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Any) As Long
```

```
Const CB_SHOWDROPDOWN = &H14F  
Private Sub Combol_GotFocus()  
    SendMessage Combol.hWnd, CB_SHOWDROPDOWN / True, ByVal 0&  
End Sub
```

Fare Kapanı

```
Public Declare Function ClipCursor Lib "user32" Alias "ClipCursor"  
(lpRect As Any) As Long
```

Farenin hareketlerini ClipCursor API'si ile bir pencere içine hapsedebilirsiniz.

- **lpRect** :Aşağıdaki RECT yapısında tanımlanmış değişken. Bu değişkenin alt değişkenleri ile farenin hareket edebileceği pencerenin koordinatları belirlenir,

```
Type RECT  
    left As Long  
    top As Long  
    right As Long  
    bottom As Long  
End Type
```

ÖRNEK: Örnek olarak fareyi formumuza kilitleyecek ve serbest bırakacak bir program yapalım. Örneğimiz için iki tane komut düğmesini formunuzun üzerine yerleştirin.

```
Option Explicit
```

```
Public Declare Function ClipCursor Lib "user32" (lpRect As Any) As Long
```

```
Private Type rect  
    left As Long  
    top As Long  
    right As Long  
    bottom As Long  
End Type
```

Microsoft Visual Basic 6.0

```
Sub Command1_Click()  
    Dim r As rect  
    Form1.ScaleMode = 3 'pixel  
    r.left = left / Screen.TwipsPerPixelX + 5  
    r.top = top / Screen.TwipsPerPixelY + 22  
    r.bottom = (top + Height) / Screen.TwipsPerPixelY - 5  
    r.right = (left + Width) / Screen.TwipsPerPixelX - 5  
    'Fareyi Form1 içine hapset  
    ClipCursor r  
End Sub  
Private Sub Command2_Click()  
    Dim r As rect  
    r.left = 0  
    r.top = 0  
    r.bottom = Screen.Height / Screen.TwipsPerPixelX  
    r.right = Screen.Width / Screen.TwipsPerPixelX  
    'Farenin yeni hareket alaninin ekranin tamamı yap.  
    ClipCursor r  
End Sub
```

Fareyi Oynatma

SendKeys metodu ile klavyeyi taklit ederek bir programa tus gönderebileceğinizi biliyorsunuz. SetCursorPos apisini kullanarak da fareyi hareket ettirebilirsiniz.

```
Public Declare Function SetCursorPos Lib "user32" Alias  
"SetCursorPos" (ByVal x As Long, ByVal y As Long) As Long
```

Buradaki x ve y parametreleri farenin gideceği koordinatları gösterir.

```
Call SetCursorPos(1000, 500)
```

Yazici Hakkında Daha Çok Bilgi

```
Public Declare Function GetDeviceCaps Lib "gdi32" Alias  
"GetDeviceCaps" (ByVal hdc As Long, ByVal nIndex As Long) As Long
```

- **hDC** : Hakkında bilgi alınacak birimin hDC numarası. Yazıcı hakkında bilgi almak için Printer.hDC, Ekran hakkında bilgi almak için Form1.hDC kullanabilirsiniz.
- **nIndex** : Hakkında bilgi alınacak konu. Bu degere verilen sayıya göre geriye dönen degerler ve anlamları şöyledir.
- **nIndex=0** :Yazıcı sürücüsünün (veya hDCsi verilen birimin) versiyonu geri döner.

```
Dim ver  
ver = getDeviceCaps(Printer.hDC, 0)  
Print "Yazıcı versiyonu=";  
Print ((ver And &HFF00) / &H100) & "." & (ver And &HFF)
```

nIndex=2: Birimin ne olduğu hakkında bilgi verir. Bu durumda geri dönen değerler ve anlamları ise şöyledir.

- 0: Vektör Plotter
- 1: Raster Display
- 2: Raster printer
- 3: Raster Camera
- 4: Character Stream,PLP
- 5: Metafile, VDM
- 6: Display File

```
Dim tek, s
tek = getDeviceCaps(Printer.hDC, 2)
Select Case tek
    Case 0: s = "Vektör Plotter"
    Case 1: s = "Raster Display"
    Case 2: s = "Raster printer"
    Case 3: s = "Raster Camera"
    Case 4: s = "Character Stream,PLP"
    Case 5: s = "Metafile, VDM"
    Case 6: s = "Display File"
End Select
Print s

nIndex = 4 :Yatay boyut (mm)
nIndex = 6 :Dikey boyut (mm)

Print "Yatay:"; getDeviceCaps(Printer.hDC, 4); " mm"
Print "Dikey:"; getDeviceCaps(Printer.hDC, 6); " mm"

nIndex = 8 :Yatay çözünürlük (pixel)
nIndex = 10 :Dikey çözünürlük (pixel)

Print getDeviceCaps(Printer.hDC, 8) & "x" &
getDeviceCaps(Printer.hDC, 10) & " piksel"

nIndex = 12 :Her pixel için kullanılan bit sayısı

nIndex = 14 :Renk yüzeyi sayısı

Print " Renk Sayisi:"; getDeviceCaps(Printer.hDC, 14);
getDeviceCaps(Printer.hDC, 12)

nIndex = 22 :Donanimin desteklediği font sayısı

Print "Font Sayisi:"; getDeviceCaps(Printer.hDC, 22)
```

ÖRNEK: Yazıcı ve ekran hakkında bilgi verecek bir program yazalım.

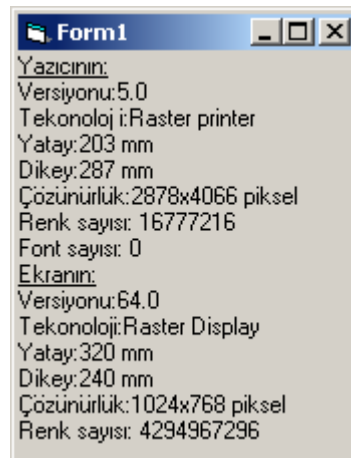
Microsoft Visual Basic 6.0

ÖRNEK : GetDeviceCaps

Option Explicit

Public Declare Function GetDeviceCaps Lib "gdi32" (ByVal hdc As Long, ByVal nIndex As Long) As Long

```
Private Sub Form_Load()  
    Show  
    Dim X As String, ver, tek  
    FontUnderline = True  
    Print "Yazicinin:"  
    FontUnderline = False  
    X = String(255, 0)  
    ver = GetDeviceCaps(Printer.hdc, 0)  
    Print "Versiyonu:"; ((ver And &HFF00) / &H100) & "." & (ver And  
&HFF)  
    Print "Teknoloji:";  
    tek = GetDeviceCaps(Printer.hdc, 2)  
    Select Case tek  
        Case 0: Print "Vektör Plotter"  
        Case 1: Print "Raster Display"  
        Case 2: Print "Raster printer"  
        Case 3: Print "Raster Camera"  
        Case 4: Print "Character Stream,PLP"  
        Case 5: Print "Metafile, VDM"  
        Case 6: Print "Display File"  
    End Select  
    Print "Yatay:"; GetDeviceCaps(Printer.hdc, 4) & " mm"  
    Print "Dikey:"; GetDeviceCaps(Printer.hdc, 6) & " mm"  
    Print "Çözünürlük:"; GetDeviceCaps(Printer.hdc, 8) & "x" &  
    GetDeviceCaps(Printer.hdc, 10) & " piksel"  
    Print "Renk sayısı:"; GetDeviceCaps(Printer.hdc, 14) * 2 ^  
    GetDeviceCaps(Printer.hdc, 12)  
    Print "Font sayısı:"; GetDeviceCaps(Printer.hdc, 22)  
    FontUnderline = True  
    Print "Ekranın:"  
    FontUnderline = False  
    X = String(255, 0)  
    ver = GetDeviceCaps(hdc, 0)  
    Print "Versiyonu:"; ((ver And &HFF00) / &H100) & "." & (ver And  
&HFF)  
    Print "Teknoloji:";  
    tek = GetDeviceCaps(hdc, 2)  
    Select Case tek  
        Case 0: Print "Vektör Plotter"  
        Case 1: Print "Raster Display"  
        Case 2: Print "Raster printer"  
        Case 3: Print "Raster Camera"  
        Case 4: Print "Character Stream, PLP"  
        Case 5: Print "Metafile, VDM"  
        Case 6: Print "Display File"  
    End Select  
    Print "Yatay:"; GetDeviceCaps(hdc, 4)  
& " mm"  
    Print "Dikey:"; GetDeviceCaps(hdc, 6)  
& " mm"
```



Form1

Yazıcının:
Versiyonu:5.0
Teknoloji:Raster printer
Yatay:203 mm
Dikey:287 mm
Çözünürlük:2878x4066 piksel
Renk sayısı: 16777216
Font sayısı: 0

Ekranın:
Versiyonu:64.0
Teknoloji:Raster Display
Yatay:320 mm
Dikey:240 mm
Çözünürlük:1024x768 piksel
Renk sayısı: 4294967296

```
Print "Çözünürlük:"; GetDeviceCaps(hdc, 8) & "x";
GetDeviceCaps(hdc, 10) & " piksel"
Print "Renk sayısı:"; GetDeviceCaps(hdc, 14) * 2 ^
GetDeviceCaps(hdc, 12)
End Sub
```

INI Dosyasından okuma

```
Public Declare Function GetPrivateProfileString Lib "kernel32" Alias
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal
lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString
As String, ByVal nSize As Long, ByVal lpFileName As String) As Long
```

INI dosyalarından bir degeri okumak için GetPrivateProfileString API'si kullanilir.

- IpFileName : Okunacak dosyanin ismi
- IpApplicationName : INI dosyasından okunacak bölümün ismi
- IpKey Name : Bu bölüm altında okunacak anahtar kelime
- IpDefault : Anahtar kelime bulunamazsa geri dönecek varsayılan deger.
- IpReturnedString : Geri dönecek degiskenin ismi
- nSize : IpReturnedString degiskeninin uzunlugu

Fonksiyondan geri dönen deger okunan karekter sayisini verir.

ÖRNEK: Win.ini dosyasının [Windows] kismindaki "Device="satiri ile geçen yazicinin ismini öğrenelim.

```
Public Declare Function GetPrivateProfileString Lib "kernel32" Alias
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal
lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString
As String, ByVal nSize As Long, ByVal lpFileName As String) As Long

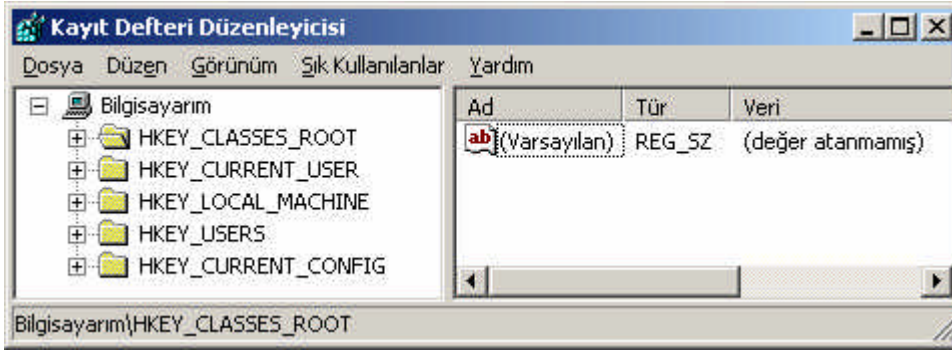
Private Sub Form_Load()
    Dim X As String, t As Integer, s
    X = String(255, 0)
    t = GetPrivateProfileString("windows", "device", "", X, Len(X),
"win.ini")
    If t = 0 Then
        s = "Bilinmiyor"
    Else
        s = Left(X, t)
    End If
    MsgBox "YAZICI:" & s
End Sub
```

Registry islemleri

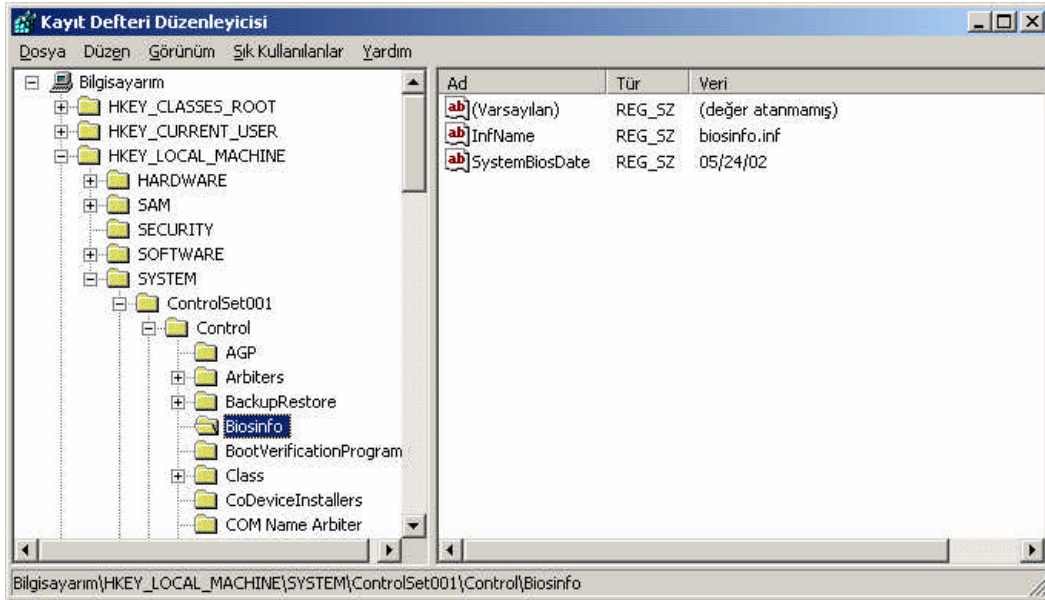
Microsoft Visual Basic 6.0

VB registry işlemleri için DeleteSetting, GetSetting, GetAllSetting ve SaveSetting fonksiyonlarını sunar ancak bu fonksiyonlarla registry dosyasındaki bütün ayarlara ulaşmak mümkün değildir. Bu noktada bazı Windows API'leri kullanılarak registry işlemleri kolayca yapılabilir.

Registry dosyası 6 ana başlıktan oluşur. Başlat-Çalıştır menüsü ile açılan pencereye **REGEDIT** yazarak Windowsun kayıt düzenleyici programını çalıştırırsanız bu ana başlıkları görürsünüz.



Bir başlığı çift tıkladığınızda ise ona ait alt başlıklar (anahtarlar) açılacaktır. Alt anahtarlar hiyerarsik bir yapıdadır ve klasör yapısını andirir.



Bu alt başlıkların altında ise bazı anahtarlar ve bu anahtarların değerleri bulunur. Seçtiğiniz alt başlık altındaki bu anahtarın değerini yukarıdaki pencerenin sağ tarafında görebilirsiniz. İşte bu değerleri okuyabilmek için bazı API'ler bulunur.

RegOpenKey apisi araciligi ile bir alt anahat açılır ve bu anahtara ait handle numarası elde edilir. Elde edilen bu handle numarası kullanılarak **RegQueryValueEx** apisi çağrılır ve bu api araciligi ile istenen anahtarın değeri öğrenilir. Son olarak RegCloseKey apisi çağrılarak açılan alt anahtar kapatılır.

```
Public Declare Function RegOpenKey Lib "advapi32.dll" Alias
"RegOpenKeyA" (ByVal hKey As Long, ByVal lpSubKey As String,
phkResult As Long) As Long
```

hKey parametresine açılacak ana anahtarın (kayıt düzenleyici programında gördüğümüz 6 ana başlıktan birinin) değeri verilir. Bu değerler şunlardır:

- Const HKEY_CLASSES_ROOT = &H80000000
- Const HKEY_CURRENT_USER = &H80000001
- Const HKEY_JDCAL.MACHINE = &H80000002
- Const HKEY_JJSERS = &H80000003

lpSubKey parametresine açılacak alt anahtarın yolu verilir. Örneğin "System\CurrentControlSet\Services\RemoteAccess"

phkResult parametresi ile açılan anahtara ait handle numarası geri döner. Bu handle kullanılarak **RegQueryValueEx** araciligi ile istenen anahtarın değeri okunur ve işlem bittikten sonra yine bu handle numarası kullanılarak **RegCloseKey** ile alt başlıklar kapatılır.

Fonksiyondan geriye dönen değer 0 ise işlem basarılmıştır.

```
Public Declare Function RegQueryValueEx Lib "advapi32.dll" Alias
"RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String,
ByVal lpReserved As Long, lpType As Long, lpData As Any, lpcbData As
Long) As Long
```

hKey parametresine **RegOpenKey** ile açılan alt başlığın handle numarası verilir. Bu numara **RegOpenKey** apisinin son parametresi ile belirlenen degiskenle geri döner.

lpValuename parametresine değeri öğrenilecek anahtarın ismi verir.

lpReserved parametresi adından da anlaşılabileceği gibi ilerisi için ayrılmış bir parametredir herhangi bir anlamı yoktur ve değeri 0 değildir.

lpType parametresine, okunacak anahtarın hangi türde bilgi içerdiğini belirleyen aşağıdaki değerlerden biri verilir.

- REG_BINARY
- REG_DWORD
- REG_SZ 'string

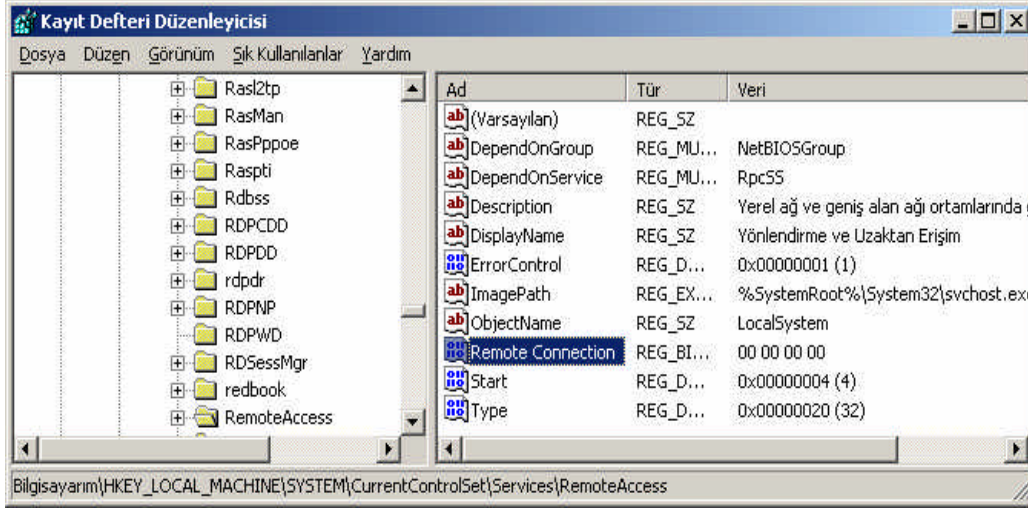
lpData parametresi ile istenen anahtarın değeri döner.

lpcbData parametresi ise dönen değerin uzunluğunu verir.

Fonksiyondan geriye dönen deger 0 ise islem basarilmistir.

```
Public Declare Function RegCloseKey Lib "advapi32.dll" Alias  
"RegCloseKey" (ByVal hKey As Long) As Long
```

RegOpenKey araciligi ile açilan bir alt anahtar bu API ile kapatilir.



ÖRNEK: HKEY_LOCAL_MACHINE ana basligi altindaki System\CurrentControlSet\Services\RemoteAccess alt basliginda bulunan Remote Connection anahtarinin degeri bilgisayarın internete bagli olup olmadigini bildirir. Eger bu anahtarın degeri 0 degilse internet baglantisi aktif demektir. Yukaridaki apileri kullanarak bu degeri registry'den okuyabiliriz.

```
Option Explicit  
Private Const HKEY_LOCAL_MACHINE = &H80000002  
Public Declare Function RegCloseKey Lib "advapi32.dll" (ByVal hKey  
As Long) As Long  
Public Declare Function RegOpenKey Lib "advapi32.dll" Alias  
"RegOpenKeyA" (ByVal hKey As Long, ByVal lpSubKey As String,  
phkResult As Long) As Long  
Public Declare Function RegQueryValueEx Lib "advapi32.dll" Alias  
"RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String,  
ByVal lpReserved As Long, lpType As Long, lpData As Any, lpcbData As  
Long) As Long ' Note that if you declare the lpData  
parameter as String, you must pass it By Value.
```

```
Private Sub Form_Load()  
    Dim hKey As Long, lpData As Long, nSizeData As Long  
    Dim alt, anahtar  
    alt = "System\CurrentControlSet\Services\RemoteAccess"  
    anahtar = "Remote Connection"
```

```
If RegOpenKey(HKEY_LOCAL_MACHINE, alt, hKey) = 0 Then
    IpData = 0&
    nSizeData = Len(IpData)
    If RegQueryValueEx(hKey, anahtar, 0&, 0&, IpData, nSizeData)
= 0 Then
        If IpData <> 0 Then
            MsgBox ("Internet baglantisi aktif")
        Else
            MsgBox ("Internet baglantisi aktif degil")
        End If
    End If
    RegCloseKey (hKey)
End If
End Sub
```

Programinizi Task Listesinden Gizleme

Programınız gizli yapsanız bile kullanıcı Ctrl+Alt+Del tuslarına bastığında açılacak listede programınız gözükecektir. Bazı API'leri kullanarak programınızın tamamen görünmez olmasını sağlayabilirsiniz.

Programınız gizlemek için formun Load olayına aşağıdaki kodu, yine çıkarken de normal hale getirmek için Formun Unload olayına aşağıdaki kodu yazmanız gerekir.

Egik Yazma

CreateFontIndirect apisini kullanarak belli bir açıyla yazı yazdırabilirsiniz.

ÖRNEK: 270 derecelik bir açıyla yukarıdan aşağıya yazdırma:

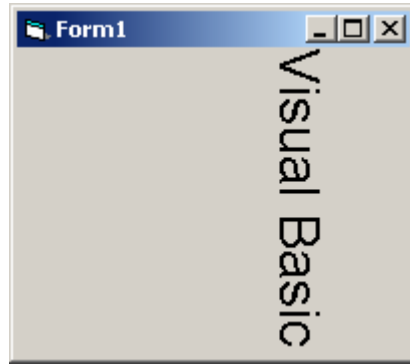
```
Private Declare Function CreateFontIndirect Lib "gdi32" Alias
"CreateFontIndirectA" (lpLogFont As LOGFONT) As Long
Private Declare Function SelectObject Lib "gdi32" (ByVal hdc As
Long, ByVal hObject As Long) As Long
```

```
Private Const LF_FACESIZE = 32
Private Type LOGFONT
    lfHeight As Long
    lfWidth As Long
    lfEscapement As Long
    lfOrientation As Long
    lfWeight As Long
    lfItalic As Byte
    lfUnderline As Byte
    lfStrikeOut As Byte
    lfCharSet As Byte
    lfOutPrecision As Byte
```

Microsoft Visual Basic 6.0

```
IfClipPrecision As Byte
lfQuality As Byte
lfPitchAndFamily As Byte
lfFaceName(LF_FACESIZE) As Byte
End Type

Private Sub Form_Load()
    Dim x As LOGFONT
    AutoRedraw = True
    açı = 270
    boyut = 20
    x.lfEscapement = açı * 10
    x.lfHeight = (boyut * -20) / Screen.TwipsPerPixelY
    rFont = CreateFontIndirect(x)
    Call SelectObject(hdc, rFont)
    CurrentX = ScaleWidth / 2
    CurrentY = 0
    Print "Visual Basic"
End Sub
```



ÖRNEK: 10'ar derece aralıklarla yazdırılmış 360 derecelik bir yazı

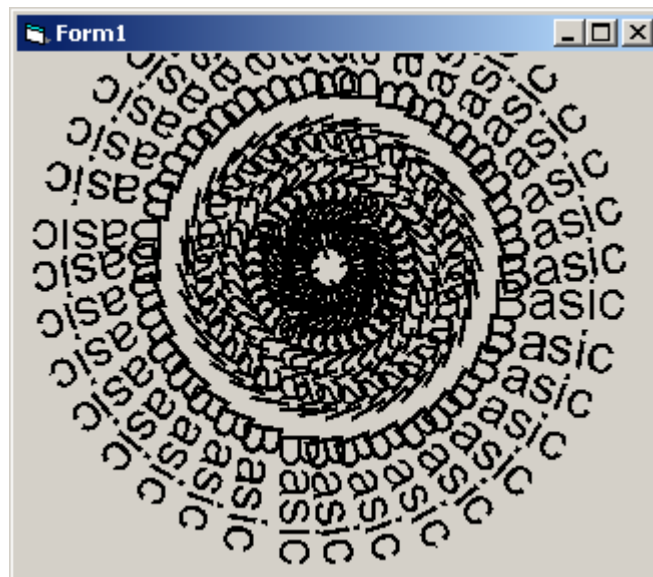
```
Private Declare Function CreateFontIndirect Lib "gdi32" Alias
"CreateFontIndirectA" (lpLogFont As LOGFONT) As Long
Private Declare Function SelectObject Lib "gdi32" (ByVal hdc As
Long, ByVal hObject As Long) As Long
```

```
Private Const LF_FACESIZE = 32
Private Type LOGFONT
    lfHeight As Long
    lfWidth As Long
    lfEscapement As Long
    lfOrientation As Long
    lfWeight As Long
    lfItalic As Byte
```

```

IfUnderline As Byte
IfStrikeOut As Byte
IfCharSet As Byte
IfOutPrecision As Byte
IfClipPrecision As Byte
lfQuality As Byte
IfPitchAndFamily As Byte
lfFaceName(LF_FACESIZE) As Byte
End Type
Private Sub Form_Load()
    Dim x As LOGFONT
    Dim açi, boyut, rfont
    ScaleMode = 3
    AutoRedraw = True
    Show
    For açi = 0 To 360 Step 10 'açi = (açi + 10) Mod 360
        boyut = 20
        x.IfEscapement = açi * 10
        x.IfHeight = (boyut * -20) / Screen.TwipsPerPixelY
        rfont = CreateFontIndirect(x)
        Call SelectObject(hdc, rfont)
        CurrentX = ScaleWidth / 2
        CurrentY = ScaleHeight / 2
        Print "Visual Basic"
    Next
End Sub

```



Bekletme

Hızlı çalışan program kodlarınızı (özellikle de döngülerde) yavaşlatmak isteyebilirsiniz. Bu işlemler için VB'de herhangi bir komut bulunmaz ancak Windowsun Sleep apisi sistemi verdiğiniz süre boyunca bekletir.

```
Public Declare Sub Sleep Lib "kernel32" Alias "Sleep" (ByVal dwMilliseconds As Long)
```

Buradaki **dwMilliseconds** parametresi ne kadar bekletileceğini belirler ve milisaniye cinsindendir.

```
Option Explicit
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Private Sub Form_Load()
    Show
    Dim i
    For i = 1 To 10
        Sleep 500
        DoEvents
        Print i
    Next
End Sub
```

Duvar Kagitini Degistirme

Windows duvar kagitini degistirmek için SystemParametersInfo apisinden yararlanilir.

```
Private Declare Function SystemParametersInfo Lib "user32" Alias "SystemParametersInfoA" (ByVal uAction As Long, ByVal uParam As Long, ByRef lpvParam As Any, ByVal fuWinIni As Long) As Long
```

Bu API bir çok ise yarar ancak duvar kagitini degistirmek için asagidaki gibi kullanilir:

```
call SystemParametersInfo (20,0,dosya,1)
```

```
Private Declare Function SystemParametersInfo Lib "user32" Alias "SystemParametersInfoA" (ByVal uAction As Long, ByVal uParam As Long, ByRef lpvParam As Any, ByVal fuWinIni As Long) As Long
```

```
Sub Form_Load()
    Call SystemParametersInfo(20, 0, "c:\windows\bulutlar.bmp", 1)
End Sub
```

Windowsun çalışma süresi

GetTickCount apisi araciligiyla windowsun ne kadar süredir açık olduğunu bulabilirsiniz.

```
Private Declare Function GetTickCount Lib "kernel32" Alias
"GetTickCount" () As Long
```

ÖRNEK: Windowsun ne kadar süredir çalıştığını sürekli gösterecek bir kod yazalım. Örneğimiz için formunuza bir Timer yerleştirin ve **Interval** özelliğini 1000 yapın.

```
Option Explicit
Private Declare Function GetTickCount Lib "kernel32" () As Long

Private Sub Timer1_Timer()
    Dim t
    t = GetTickCount&
    Caption = (t \ 1440000) & ":" 'saat
    t = t Mod (60000 * 24) 'kalan dakikayı bul
    Caption = Caption & (t \ 60000) & ":" 'dakika
    t = t Mod (60000) 'kalan saniyeyi bul
    Caption = Caption & (t \ 1000) 'saniye
    Label1.Caption = Caption
End Sub
```

Belgeler Menüsüne Doküman Ekleme

Windows 9X ve üstü versiyonlarda Baslat menusu altında Belgelerim menusu olduğunu biliyorsunuz. Bu menüde son kullandığınız belgelerin bir listesi tutulur. Eger sizin programınızın kullandığı belgelerin de bu listeye eklenmesini isterseniz SHAddToRecentDocs apisini kullanabilirsiniz. Bu api aynı zamanda Belgelerim menusunu silmek için de kullanılır.

```
Private Declare Sub SHAddToRecentDocs Lib "shell32.dll" (ByVal uFlags As Long, ByVal pv
As String)
```

uFlags parametresine 2 degeri verildikten sonra pv parametresine eklenecek dokümanın yolu ve adi girilir. Eger ikinci parametreye bos string verilirse belgeler menusu silinir.

```
Option Explicit
Private Declare Sub SHAddToRecentDocs Lib "shell32.dll" (ByVal
uFlags As Long, ByVal pv As String)

Private Sub Form_Load()
    Call SHAddToRecentDocs(2, "c:\windows\bulutlar.bmp")
End Sub
```

Dizin Seçme Penceresi

Microsoft Visual Basic 6.0

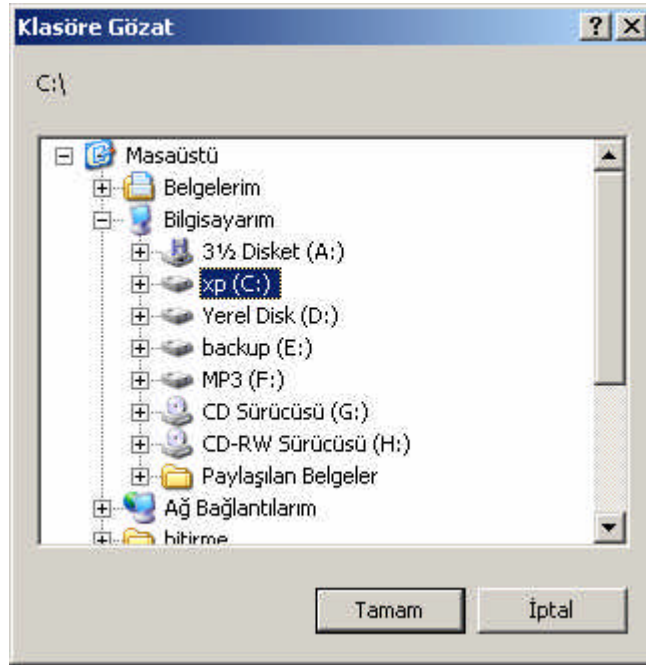
SHBrowseForFolder apisi araciligiyla Windows'un asagidaki dizin seçme penceresi gösterilebilir ve **SHGetPathFromIDList** apisi ile de seçilen dizin gösterilebilir.

```
Private Type BrowseInfo
    hWndOwner As Long
    pIDLRoot As Long
    pszDisplayName As Long
    lpszTitle As Long
    ulFlags As Long
    lpfnCallback As Long
    IParam As Long
    iImage As Long
End Type

Const BIF_RETURNONLYFSDIRS = 1
Private Declare Sub CoTaskMemFree Lib "ole32.dll" (ByVal hMem As Long)
Private Declare Function lstrcat Lib "kernel32" Alias "lstrcatA"
    (ByVal lpString1 As String, ByVal lpString2 As String) As Long
Private Declare Function SHBrowseForFolder Lib "shell32" (lpbi As BrowseInfo) As Long
Private Declare Function SHGetPathFromIDList Lib "shell32" (ByVal pidList As Long, ByVal lpBuffer As String) As Long

Private Sub Form_Load()
    Dim yer As Integer, id As Long
    Dim yol As String, inf As BrowseInfo
    inf.hWndOwner = hWnd
    inf.lpszTitle = lstrcat("C:\", "")
    inf.ulFlags = BIF_RETURNONLYFSDIRS
    id = SHBrowseForFolder(inf)
    If id Then
        yol = String(260, 0)
        SHGetPathFromIDList id, yol
        CoTaskMemFree id
        yer = InStr(yol, vbNullChar)
        If yer Then
            yol = Left$(yol, yer - 1)
        End If
    End If
    MsgBox "Seçilen dizin " & yol
```


End Sub



Bilgisayar Adını Öğrenme

Denetim Masasında Ağ simgesini çift tıklayarak açılan pencerenin Tanımlama kısmına geçerseniz bilgisayarınızın ağ üzerindeki ismini görebilirsiniz.

Bu adı öğrenebilmek için **GetComputerName** apisini kullanabilirsiniz.

```
Private Declare Function GetComputerName Lib "kernel32" Alias
"GetComputerNameA" (ByVal IpBuffer As String, nSize As Long) As Long
```

Bilgisayarın adı **IpBuffer** parametresi ile verilen degiskende döner. Bu degiskenin uzunluğu ise **nSize** parametresi ile bildirilir.

```
Private Declare Function GetComputerName Lib "kernel32" Alias
"GetComputerNameA" (ByVal IpBuffer As String, nSize As Long) As Long
```

```
Private Sub Form_Load()
    Dim s As String
    s = Space(255)
    Call GetComputerName(s, 255)
    MsgBox ("Bilgisayarın adı:" & s)
End Sub
```

System Tray Uygulamalari

Windows görev çubugunun en saginda bulunan yere System Tray denir.



Siz de yazacaginiz programlari buraya yerlestirmek isteyebilirsiniz. Bu islem için **Shell_NotifyIcon** API'si kullanilir.

```
Private Declare Function Shell_NotifyIcon Lib "shell32.dll" Alias "Shell_NotifyIconA" (ByVal dwMessage As Long, lpData As NOTIFYICONDATA) As Long
```

dwMessage

Bu parametreye verilecek asagidaki degerlerden biriyle islemin türü belirlenir.

0:Uygulama System Tray'a yerlestirilirken bu parametreye 0 degeri verilir.

1:System Tray'a yerlesmis uygulama üzerinde bazi degisiklikler yaptiginizda bu degisikliklerin etkili olabilmesi için API bu degerle yeniden çağrılır.

2:System Tray'a yerlesmis bir uygulama kaldirilmak istendiginde bu parametreye 1 verilir. Ayni zamanda programiniz sona ererken de Shell_NotifyIcon API'sini 1 degeriyle tekrar çağirarak System Tray'dan kaldirildigini bildirmeniz gerekir.

InData

NOTIFYICONDATA yapisindan tanimlanmis bu tipte, uygulamaya ait bazi özellikler belirlenir. Asagidaki NOTIFYICONDATA yapisini formunuzda tanımlamaniz ve bu tipten bir degisken kullanmanız gerekir.

```
Type NOTIFYICONDATA
    cbSize As Long
    hwnd As Long
    uID As Long
    uFlags As Long
    uCallbackMessage As Long
    hlcon As Long
    szTip As String * 64
End Type
```

Bu tipte bulunan parametrelerin anlamlari sunlardir:

cbSize :Tanimlanan degiskenin uzunlugu.

Hwnd :System Tray'a yerlestirilecek formun hWnd numarasi

uCallbackMessage :System Tray'a yerlesmis icon üzerinde kullanıcı bir işlem yaptığında çağrılacak olay. Genellikle MouseMove olayı verilerek gerekli kod bu olaya yazılır. Bu parametreye &H200 değeri verilirse MouseMove olayı bildirilmiş olur.

HIcon :System Tray'da gösterilecek Icon bu özellik ile belirlenir. Formun Icon özelliğini buraya verebileceğiniz gibi farklı ikonlar da verebilirsiniz.

szTip :Kullanıcı fare ile System Tray'daki ikon üzerinde bir müddet durduğunda gösterilecek açıklama mesajı.

- ❖ IpData parametresine yukarıdaki tipten bir değişken tanımlanarak gerekli özellikler ayarlandıktan sonra dwMessage parametresine 0 verilerek Shell_NotifyIcon API'si çağrılır. Böylece uygulama System Tray'a yerleşmiş olur.

Option Explicit

```
Private Declare Function Shell_NotifyIcon Lib "shell32" Alias "Shell_NotifyIconA" (ByVal dwMessage As Long, pnid As NOTIFYICONDATA) As Boolean
```

```
Private Type NOTIFYICONDATA
    cbSize As Long
    hwnd As Long
    uID As Long
    uFlags As Long
    uCallbackMessage As Long
    hIcon As Long
    szTip As String * 64
End Type
```

```
Dim tray As NOTIFYICONDATA 'Global Degisken
```

```
Private Sub Form_Load()
    tray.cbSize = Len(tray)
    tray.hwnd = Form1.hwnd
    tray.uID = vbNull
    tray.szTip = "Bizim Uygulama" + Chr(0)
    tray.uCallbackMessage = &H200
    tray.uFlags = 7
    tray.hIcon = Form1.Icon
    'system traya ekle
    Shell_NotifyIcon 0, tray
    Hide 'formu gizle
End Sub
```

Microsoft Visual Basic 6.0

- ❖ Eger bazı parametreleri degistirirseniz, örneğin lpData parametresindeki Icon'u degistirirseniz bu degisikligin etkili olmasi için Api'yi 1 parametresi ile tekrar çağırmanız gerekir.

```
Private Sub Command1_Click()  
    tray.szTip = "Bizim Uygulama Çalışıyor" + Chr(0)  
    'system trayi güncelle  
    Shell_NotifyIcon 1, tray  
End Sub
```

- ❖ Uygulamayı System Tray'dan kaldırmak istediğinizde ilk parametreye 2 degerini vererek çağırabilirsiniz.

```
Private Sub Form_Unload(Cancel As Integer)  
    'system traydan kaldır  
    Shell_NotifyIcon 2, tray  
End Sub
```

- ❖ System Tray'a yerleşmiş icon üzerinde kullanıcı tarafından bir işlem yapılmak istendiğinde MouseMove olayı meydana gelir. Bu olaydaki, normalde farenin tiklandığı koordinati gösteren X parametresi öze! bir amaçla kullanılır ve kullanıcının ne yaptığı programa bildirilir. MouseMove olayındaki X parametresi aşağıdaki degerlerden birini alabilir

&H201 : Sol tusa basıldı.
&H202 : Sol tus bırakıldı.
&H203 : Çift tiklandı.
&H204 : Sağ tusa basıldı.
&H205 : Sağ tus bırakıldı.
&H206 : Sağ tusla çift tiklandı.

MouseMove olayına yazılacak kodla X parametresinin yukarıdaki degerleri kontrol edilir ve buna uygun işlem yaptırılır. Ama öncelikle Formun ScaleMode özelliğine 3 degeri verilerek Pixel modu aktif hale getirilmektedir. Çünkü normalde X parametresi koordinat ifade eden bir parametredir ve VB ile Windows'un koordinat birimleri farklıdır. Pixel modu aktif hale getirilerek her iki birim eşitlenmelidir.

```
Private Sub Form_Load()  
    '.....  
    ScaleMode = 3 'Pixel  
End Sub
```

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As  
Single, Y As Single)  
    Select Case X  
        Case &H202  
            'Sol tusla tiklandığında yapılacak işlem  
        Case &H203  
            'Çift tiklandığında yapılacak işlem  
        Case &H205
```

```

        'Sag tusla tiklandiginda yapılacak islem
    End Select
End Sub

```

Örneğin kullanıcı System Tray'daki ikonu çift tıkladığında formun gösterilmesini ve sağ fare tusuna bastığında ise bir menünün gösterilmesini isteyebilirsiniz.

- ❖ Eğer ikonu sağ tusla tiklandığında bir menü açılmasını istiyorsanız önce Menü Editör aracılığı ile bir menü tasarlamamız ve MouseMove olayına yazacağınız kodla bu menüyü aktif hale getirmeniz gerekir.

Tools-Menu Editör aracılığı ile menünüzü tasarlayın.

Burada ana menünün ismini MnMenu verin. Normalde bu menünün menü çubuğunda gözükmemesi için Visible özelliğinin işaretini kaldırdık. Böylece menü formda görülmeyecek, ancak yazacağınız kodla aktif hale gelecektir.

```

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Select Case X
        Case &H203
            'Çift tıkladığında formu gösterir
            Form1.Show
        Case &H205
            'Sag tusla tiklandiginda menüyü gösterir
            PopupMenu MnMenu
    End Select
End Sub

```

Artık ikon çift tıkladığında formumuz görülecek, sağ tusla tiklandığında ise tasarladığımız menü açılacaktır.

Menülerin Click olaylarına gerekli kodu yazarak üzerine düşen işlemi yapmasını sağlayabilirsiniz.

- ❖ System Tray olarak tasarladığınız formun simge durumuna küçültülmesi durumunda görev çubuğuna yerleşmemesini isterseniz Formun resize olayına yazacağınız kodla formu gizleyebilirsiniz.

```

Private Sub Form_Resize()
    If WindowState = 1 Then 'minimize edildiyse gizle
        Hide
    End If
End Sub

```

Bu durumda formu gösterecek kodda da bir değişiklik yaparak ikon çift tıkladığında formun minimize durumunda çıkarılması gerekir. Aksi takdirde form hep gizli kalacaktır.

```

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Select Case X

```

Microsoft Visual Basic 6.0

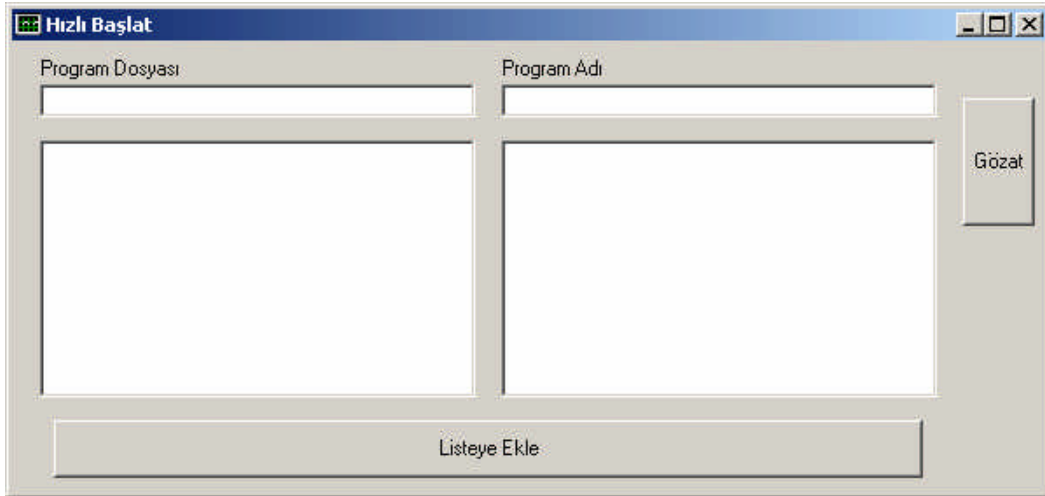
```
Case &H203 'Çift tiklandığında formu göster
Form1.Show
Form1.WindowState = 0 'Normal boyutlarına getir.
Case &H205 'Sağ tusla tiklandığında menüyü göster
PopupMenu MnMenu
End Select
End Sub
```

- ❖ System Tray olarak tasarladığınız bir uygulamanın Task Listesinde görölme mesini isteyebilirsiniz. Bu durumda App nesnesinin TaskVisible özelliğine False verebilirsiniz.

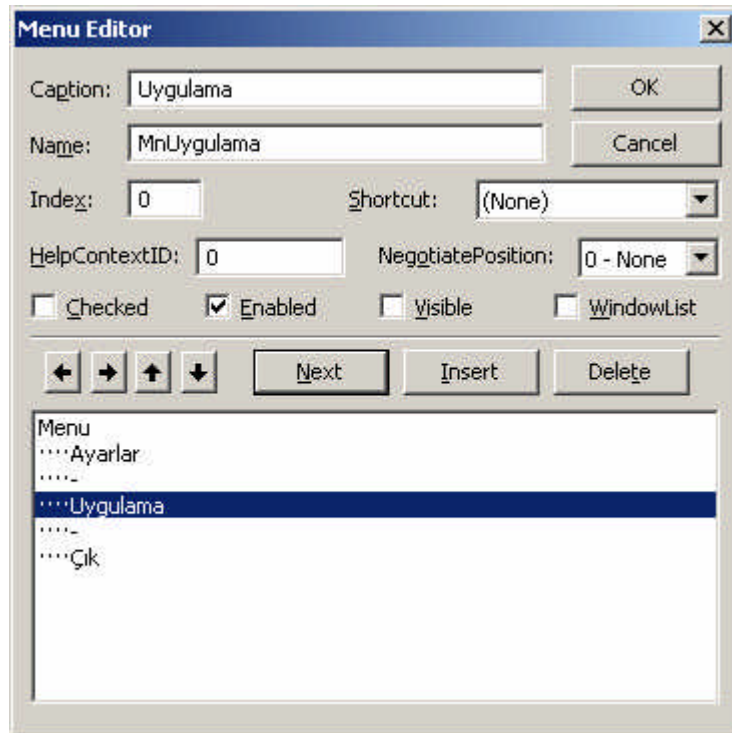
```
Private Sub Form_Load()
'.....
'Task listesinde gösterme
App.TaskVisible = False
End Sub
```

ÖRNEK: Örnek olarak bir System Tray uygulaması yazalım. Örneğimizde Kullanıcı en çok kullandığı programları tanıtılsın ve biz bunları bir menüye ekleyelim. Kullanıcı System Tray'a yerleştirdiğimiz simgeyi sağ fare tusu ile tıkladığında belirlendiği uygulamaların listesi gelsin ve listeden seçtiği bir uygulama çalıştırılsın.

Örneğimiz için aşağıdaki formu hazırlayın. Forma bir CommonDialog kontrolü, iki text kutusu, iki Liste ve iki tane komut düğmesi yerleştirin.



Tools-Menu Editör aracılığı ile açılan aşağıdaki pencereden menünüzü tasarlayın.



Ana menünün ismini MnMenu olarak belirleyin ve Visible özelliğini kaldırın. Böylece bu menü formda görünmeyecektir. Ayrıca Uygulama menüsünün ismini MnUygulama olarak verin, Visible özelliğini kaldırın ve Index özelliğine 0 verin. Böylece bu menüyü dizi olarak tanımlamış olacağız. Dizi olarak tanımladığımız bu menüden, program çalışırken yenilerini Load komutu ile yükleyip, belirlenen her bir uygulama için bir menü oluşturacağız.

ÖRNEK: System Tray Uygulaması

```
Option Explicit
Private Declare Function Shell_NotifyIcon Lib "shell32" Alias
"Shell_NotifyIconA" (ByVal dwMessage As Long, lpData As
NOTIFYICONDATA) As Long
    Private Type NOTIFYICONDATA
        cbSize As Long
        hwnd As Long
        uID As Long
        uFlags As Long
        uCallbackMessage As Long
        hicon As Long
        szTip As String * 64
    End Type
Dim tray As NOTIFYICONDATA

Private Sub Command1_Click()
    On Local Error GoTo iptal
```

Microsoft Visual Basic 6.0

```
CommonDialog1.DialogTitle = "Uygulamayi seçiniz"
CommonDialog1.CancelError = True
CommonDialog1.Filter = "Programlar|*.exe;*.com;*.bat;*.pif|Bütün
Dosyalar!*..*||"
CommonDialog1.FilterIndex = 0
CommonDialog1.ShowOpen
Text1 = CommonDialog1.FileName
Text2 = CommonDialog1.FileTitle
Command2.SetFocus
Exit Sub

iptal:
Exit Sub
End Sub

Private Sub Command2_Click()
List1.AddItem Text1
List2.AddItem Text2
'Yeni menü olustur
Dim i
i = List1.ListCount
Load MnUygulama(i)
MnUygulama(i).Visible = True
MnUygulama(i).Caption = Text2
End Sub

Private Sub Form_Load()
Caption = "Hizli Baslat"
tray.cbSize = Len(tray)
tray.hwnd = Form1.hwnd
tray.uID = vbNull
tray.szTip = "Hizli Baslat " + Chr(0)
tray.uCallbackMessage = &H200
tray.uFlags = 7
tray.hicon = Form1.Icon 'system traya ekle
Shell_NotifyIcon 0, tray
Hide 'formu gizle
ScaleMode = 3 'pixel
'Task listesinde gösterme
App.TaskVisible = False
If Dir("liste.dat") <> "" Then 'liste varsa aç
Dim m, i
i = 0
'Listeyi dosyaya kaydet
Open "liste.dat" For Input As #1
While Not EOF(1)
Input #1, m
List1.AddItem m
Input #1, m
List2.AddItem m
i = i + 1
Load MnUygulama(i)
MnUygulama(i).Visible = True
MnUygulama(i).Caption = m
```



```
        Wend
        Close #1
    End If
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, x As
Single, Y As Single)
    Select Case x
        Case &H203 'Çift tiklandığında formu göster
            Form1.Show
            Form1.WindowState = 0 'Normal boyutlarına getir.
        Case &H205 'Sağ tusla tiklandığında menüyü göster
            PopupMenu MnMenu
    End Select
End Sub

Private Sub Form_Resize()
    If WindowState = 1 Then
        'minimize edildiyse gizle
        Hide
    End If
End Sub

Private Sub Form_Unload(Cancel As Integer)
    'system traydan kaldır
    Shell_NotifyIcon 2, tray
    Dim m, i
    'Listeyi dosyaya kaydet
    Open "liste.dat" For Output As #1
    For i = 0 To List1.ListCount - 1
        m = List1.List(i)
        Write #1, m
        m = List2.List(i)
        Write #1, m
    Next
    Close #1
End Sub

Private Sub MnAyarlar_Click()
    WindowState = 0
    Show
End Sub

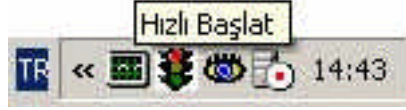
Private Sub cik_Click()
    Unload Me
End Sub

Private Sub MnUygulama_Click(Index As Integer)
    On Local Error GoTo program_hatasi
    Dim x
    x = Shell(List1.List(Index - 1), vbNormalFocus)
    Exit Sub
program_hatasi:
805
```

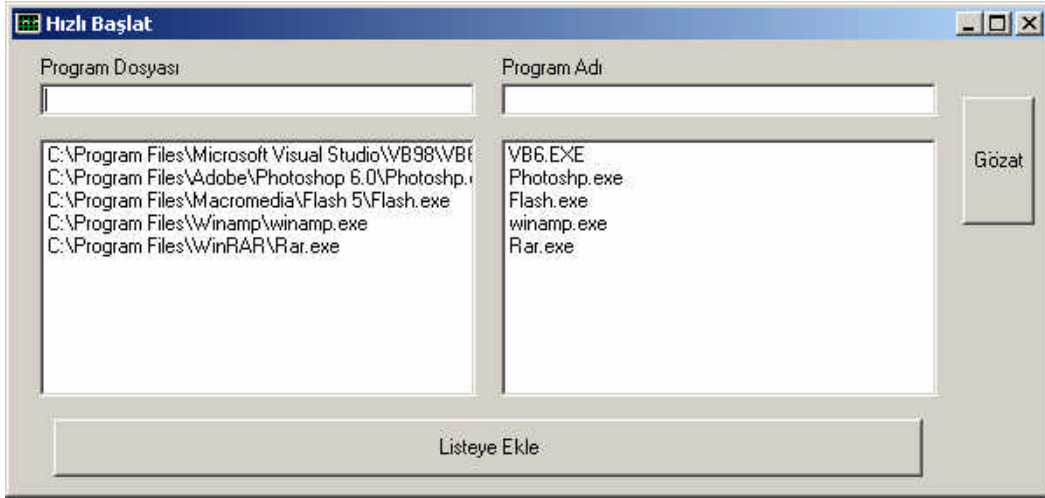
Microsoft Visual Basic 6.0

```
MsgBox ("Program alistirilamadi: " & Error)  
Exit Sub  
End Sub
```

Programi alistirdiginizda System Tray'a uygulamamiz yerlesecektir.

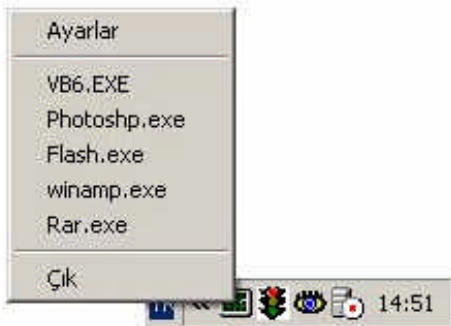


System Tray'a yerlesmis simgeyi ift tıklarsanız veya sag fare tusu ile tıklayıp açılan menüden Ayarlar komutunu seçerseniz formumuz ekranda görülecek ve menüye ekleyeceginiz uygulamalarinizi listeye ekleyebilmenizi



saglayacaktır.

Listeye eklediginiz uygulamalar menüye eklenecektir. System Tray'a yerlesmis simgeyi sag fare tusu ile tıklarsanız seçtiginiz uygulamaların listelendiği menü açılacaktır.



Listeden seçtiginiz bir uygulama alistirilacaktır.

17-NEDEN IIS?

IIS'ten bahsetmeden önce kısaca PWS(Personel Web Server) ile ilgili birkaç noktaya değinme gereği hissediyorum:

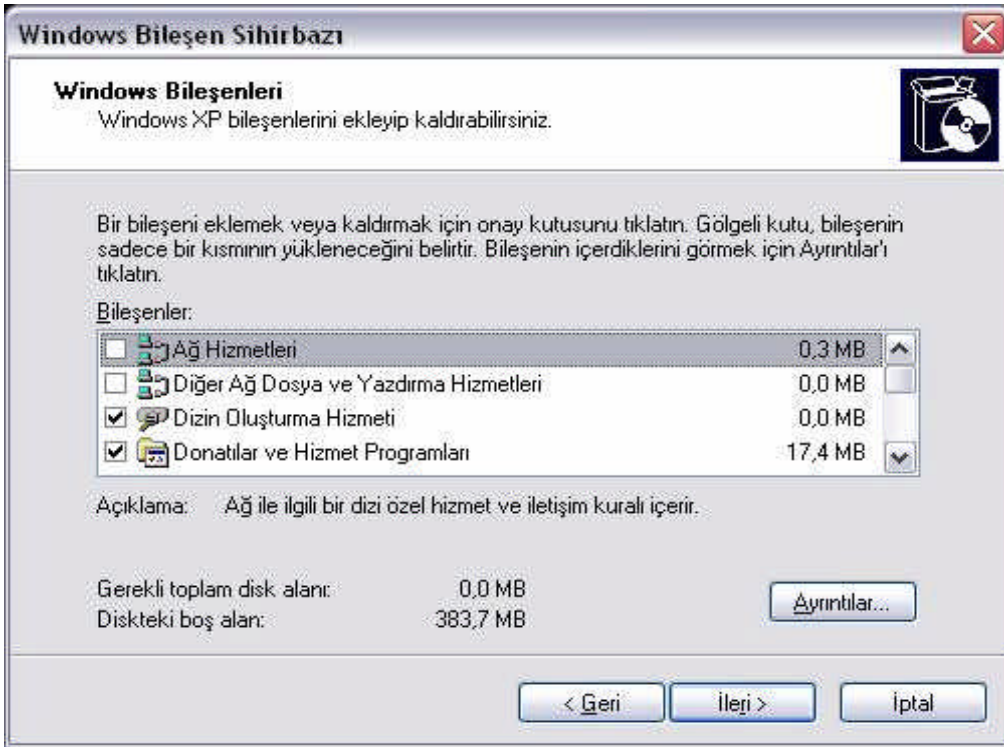
1. PWS, hepimizin bildiği gibi Windows 98 üzerinde çalışmaktadır. Windows ME ile beraber PWS gelmemektedir. Birkaç makina üzerinde yapılmış olan denemelerde Windows 98 CD'si içerisinde yer alan PWS'yi Windows ME üzerinde kurulduğunda hiçbir problemle karşılaşılmıyor. Ancak düzgün çalıştıramayan ve problem yaşayanların varlığını biliniyor.
2. Belki de çoğumuz ilk ASP kodlarını PWS üzerinde yazmıştır. Ancak Windows 98 ve PWS bize gerçek server ortamını sağlamadıkları için çoğu zaman yazdığımız kodları servera gönderdiğimiz zaman çalışmadığını, hata verdiğini görmüşüzdür.
3. Windows 98 bir server işletim sistemi olmadığı için, kullandığımız bileşenler (component) ile ilgili hatalar ve problemler oluşabilmektedir.
4. Eğer ASP ile sadece hobi olarak uğraşmıyorsanız, çalıştığınız ortamlarda Windows 2000 Server ile meşgul olacaksınız demektir. Bu nedenle Windows 2000 Server üzerinde çalışmak yararlı olacaktır.
5. Server üzerinde çalışacak olan ASP kodlarının tüm geliştirme aşamasında benzer bir ortamda çalışmak size oluşabilecek hataları daha kolay görme, anında çözümler üretebilme gibi avantajlar sağlayacaktır. Yazdığınız kodun IIS üzerinde nasıl bir tepki vereceğini programlamayı tam olarak bitirmeden görme şansına sahip olabileceksiniz.

Yukarıda yazılan sebeplerden dolayı tüm arkadaşlara kodlarını, eğer imkanları müsaitse Windows 2000 Server üzerinde veya en azından Windows 2000 Professional Edition üzerine IIS kurarak (kurulum sırasında default olarak yüklenmez) yazmalarını tavsiye ediyorum.

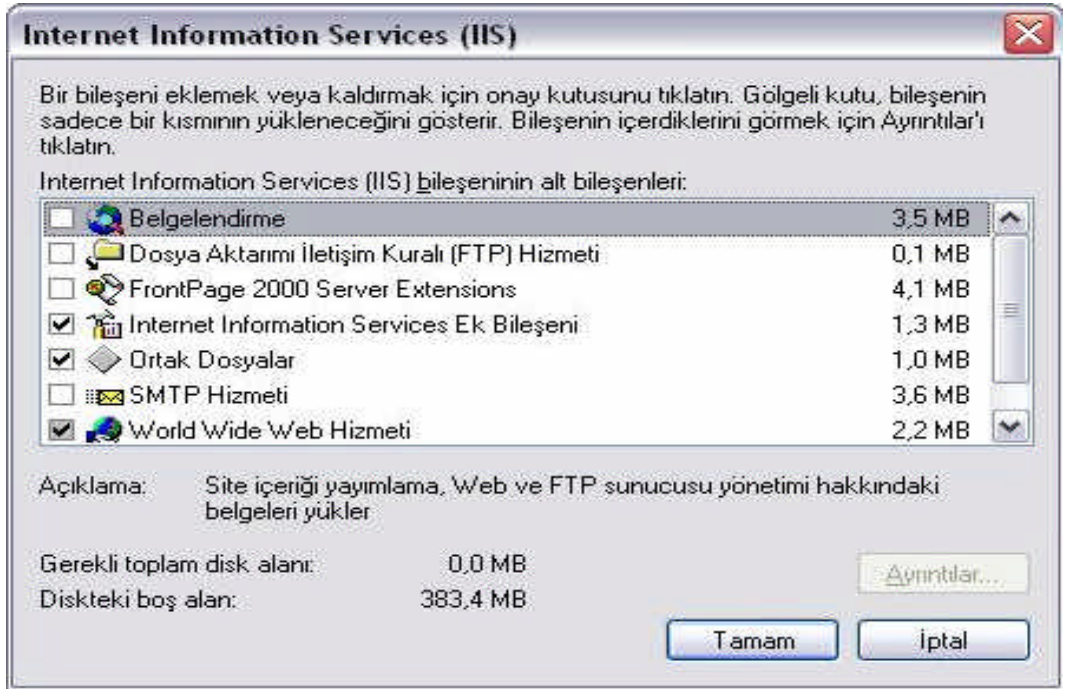
KURULUM

Windows 2000 Server üzerinde yüklenmemiş olma ihtimali ve Professional üzerine kurulması gerektiği için nasıl olduğunu bilmek amacıyla kısaca IIS'in kurulumundan bahsetmek istiyorum:

Start/Settings/Control Panel'e (Baslat/Ayarlar/Denetim Masası) tıklayarak **Control Panel'i** (Denetim Masası) açıyoruz. Control Panel'de **Add Remove Programs** (Program Ekle/Kaldır) linkine çift tıklayarak açılan pencerede "Add/Remove Windows Components"i (Windows Bileşenleri Ekle/Kaldır) tikliyoruz. Karşınıza gelecek olan yeni pencerede windows bileşenlerini görebilirsiniz :



IIS linki üzerine çift tıklayarak veya "Details" linkine basarak IIS içerisinde yer alan bileşenleri (server ve servisler) görebilir, ihtiyacınız olanları işaretleyerek kurulumlarını sağlayabilirsiniz. Sıra ile bunlardan bahsedelim:



Common Files : IIS'in çalışabilmesi için gerekli dosyalar. Bu dosyaların mutlaka kurulması gerekmektedir.

Documentation : IIS, Web ve FTP serverlar üzerinden publish (yayın) ile ilgili yardımları ve örnekleri içeren dokümanlar.

File Transfer Protocol (FTP) Server : Dosya upload ve download işlemleri için gerekli olan servis.

Frontpage 2000 Server Extensions : Frontpage ve Visual Interdev aracılığı ile web siteleriniz üzerinde işlem yapabilmeyi sağlayacak olan araç.

Internet Information Services Snap-In : IIS için gerekli olan yönetim arabirimi.

Internet Services Manager (HTML) : Browser aracılığı ile IIS'i ve web sitelerinizi yönetmeyi sağlar.

NNTP Service : Açılımı Network News Transfer Protocol olan NNTP servisi vasıtasıyla server üzerinden haber grupları yayınlayabilirsiniz.

SMTP Service : Açılımı Simple Mail Transfer Protocol olan SMTP servisi aracılığıyla server üzerinden mail gönderebilirsiniz. **Bu servis ile sadece mail gönderebilirsiniz, mail alabilmek için bir mail server'a ihtiyacınız vardır.**

Visual InterDev RAD Remote Deployment Support : Bu bileşen yardımıyla server'iniz üzerindeki dosyalara Visual InterDev ile uzaktan bağlanılarak direkt üzerinde çalışılabilmesi için gerekli desteği sağlayabilirsiniz.

World Wide Web Server : En önemli bileşen. Server'iniz üzerinden web sitelerinin tüm dünyaya yayınlanabilmesi için gerekli servis.

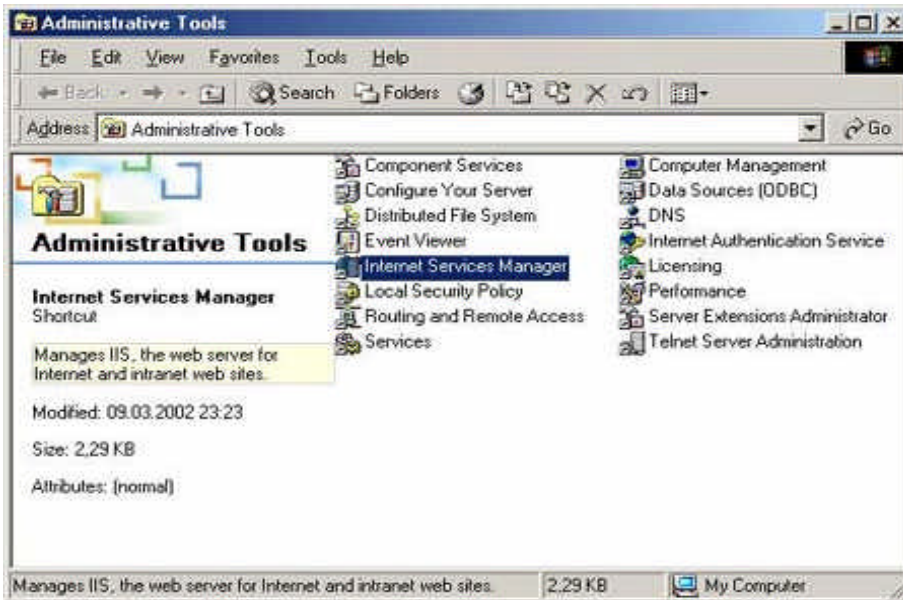
İhtiyacınıza göre istediğiniz bileşenleri seçtikten sonra sırasıyla OK ve Next tuslarına basıyoruz. Böylece IIS kurulumunu tamamlamış oluyoruz (Bu işlem için işletim sisteminin CD'sine ihtiyacınız olacaktır).

NOT : Windows 2000 Professional yukarıda saydığımız bileşenlerin tümünü içermez. FTP Server, Frontpage 2000 Server Extensions ve SMTP Service bileşenlerini içermektedir.

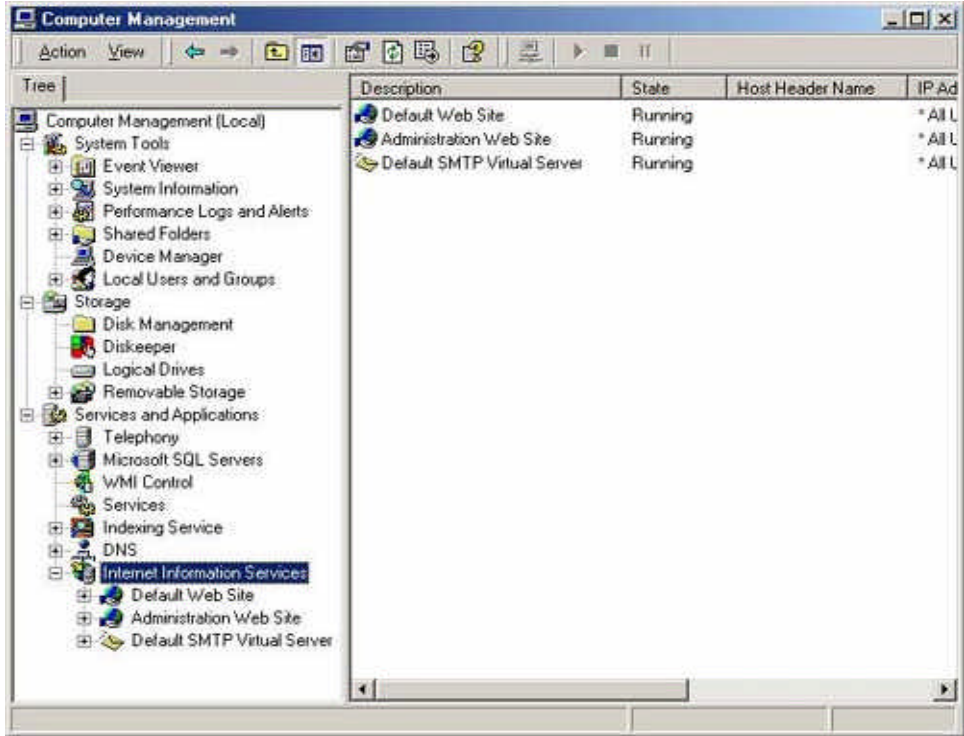
IIS'E ERISİM

Web-FTP sitesi tanımlamak ve ayarlarını yapmak için IIS'e birkaç farklı yoldan ulaşabilirsiniz :

1. Start/Settings/Control Panel/Administrative Tools/Internet Services Manager yolunu izleyerek,
2. Start/Programs/Administrative Tools/Internet Services Manager yolunu kullanarak,



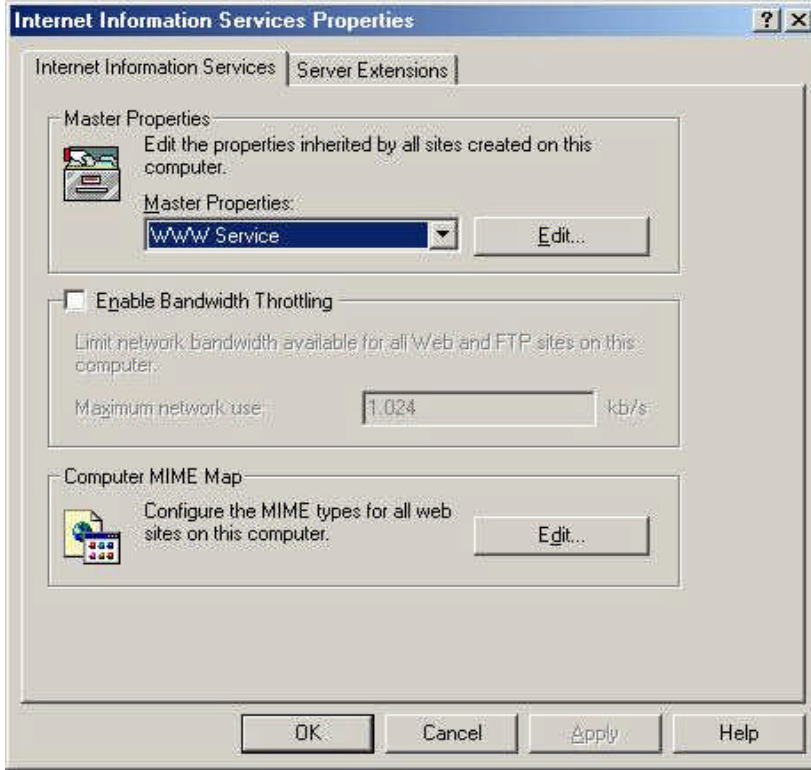
3. **"My Computer"**'e sag tiklayip, **"Manage"** seçeneğine basarak açılacak olan **"Computer Management"** penceresinde **"Services and Applications"** bölümünden IIS'e ulaşabilirsiniz.



Genelde üçüncü yolu kullanmayı tercih ediyorum, çünkü "Computer Management" penceresi vasıtasıyla IIS ile uğraşırken erismem gerebilecek hemen tüm araçlara ulaşabiliyorum (Event Viewer, Local Users and Groups, Services, DNS servisi gibi).

IIS'DE GENEL AYARLAR

IIS'e ulastiktan sonra hemen ayarlari yapmaya baslayalim :
"Internet Information Services" linkine sag tiklayip, "Properties"
seenegine basarak "Internet Information Services Properties"
penceresini aiyoruz.



Bu penceredeki seenekler yardimiyla yapılacak olan ayarlar, o makina
üzerinde yer alan tüm siteler için geçerli olacaktır.

Internet Information Services :

Internet Information Services bölümünde Web ve FTP siteleri için genel
ayarlar, bant genişligi sinirlaması ve dosya tipleri ile ilgili işlemler
gerçekleştirilebilir.

Master Properties :

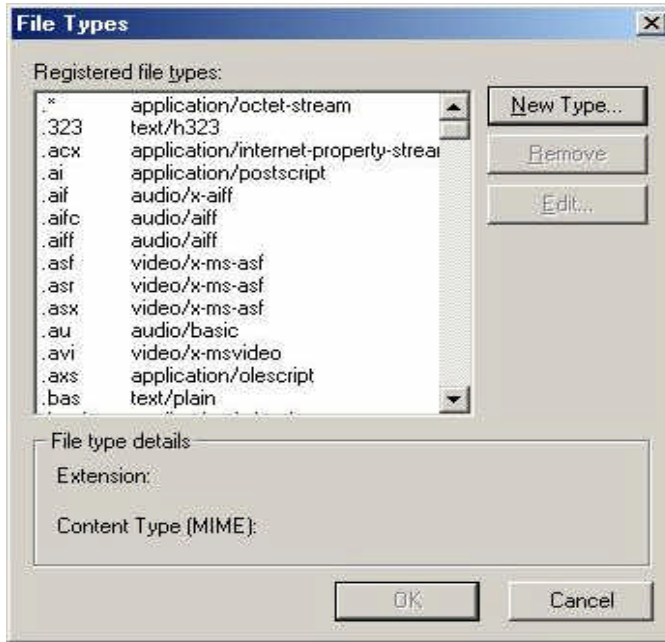
Bu bölümde **WWW Service** (Web Sitesi Yayinlama Servisi) ve **FTP Service** (FTP Sitesi Yayinlama Servisi) hizmetlerinin tüm sitelerinde geçerli olacak ayarlar yapılabilir. Her bir servis içi **'Edit'** tusuna basilarak açilacak olan pencere yardimiyla ayarlar gerçekleştirilebilir. Bu ayarlardan daha sonra, Web sitesi ve FTP sitesi olusturmayi anlatacagim bölümlerde bahsedecem.

Enable Banwidth Throttling :

Server üzerinde çalışacak olan tüm Web ve FTP siteleri için geçerli olacak bant genisligi sinirlamasi gerçekleştirilebilir. Bu tür genel bir kisitlama, bir veya birkaç adet sitenin yer aldigi kendinize ait bir server üzerinde uygulanmaz. Daha çok birçok sitenin barindirildigi hosting firmalari için uygundur.

Computer MIME Map :

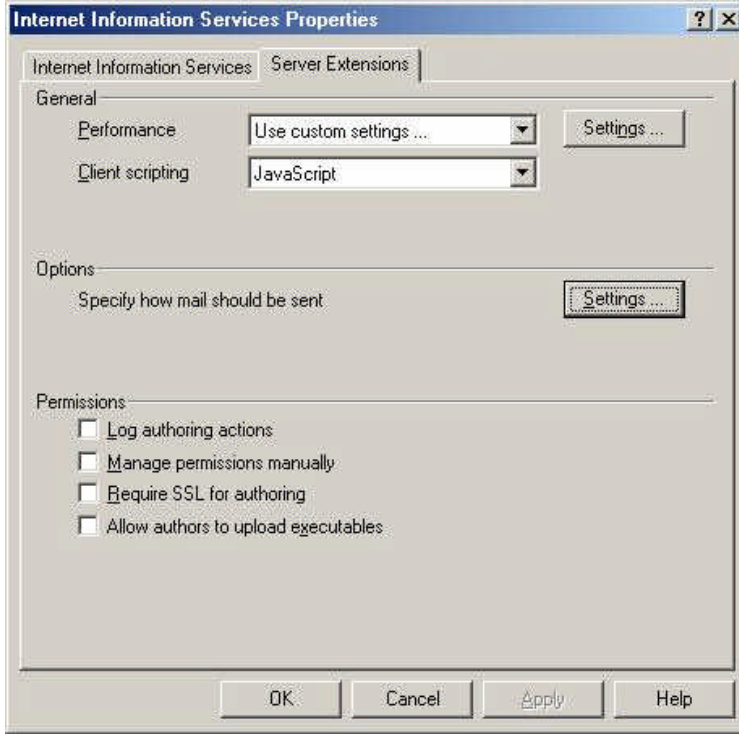
Bu bölümde makina üzerindeki tüm Web siteleri için geçerli olacak ve Web siteleri tarafından kullanıcıya ulasmasi istenen dosya tipleri belirlenebilir.



Web siteleri üzerinden ziyaretçiye iletilmesi istenmeyen dosya tipleri de yine bu bölümden kaldırılabilir.

Server Extensions :

Server Extensions bölümünde Frontpage Server Extensions ile hazırlanmış Web siteleri için ayarlar yapılabilir.



General :

Performance:

Web sitesinde yer alacak sayfa sayısını belirtilerek veya özel ayarlar yapılarak en iyi performans alınması sağlanabilir.

Client scripting : Frontpage Server Extensions tarafından otomatik oluşturulacak istemci taraflı scriptler için kullanılacak olan dil JavaScript veya VBScript olarak belirlenir.

Options :

Specify how mail should be sent :

Gerektigi durumda e-mail tabanlı web özelliklerini (e-mail form handler, ziyaretçiye mail göndermek) kullanırken gerekli olan mail ayarları "Settings" tusuna basılarak yapılabilir.

Permissions :

Log authoring actions :

Bu kutucuk işaretlenerek Web sitesi üzerinde yapılan işlemlerin (dosya ekleme, silme gibi) kaydı (log) tutulabilir. Bu loglar _vti_log klasörü altında yer almaktadır.

Manage Permissions Manually :

Bu kutu işaretlenerek güvenlik ayarlarının Frontpage Server Extensions yerine elle yapılması sağlanabilir.

Require SSL for authoring :

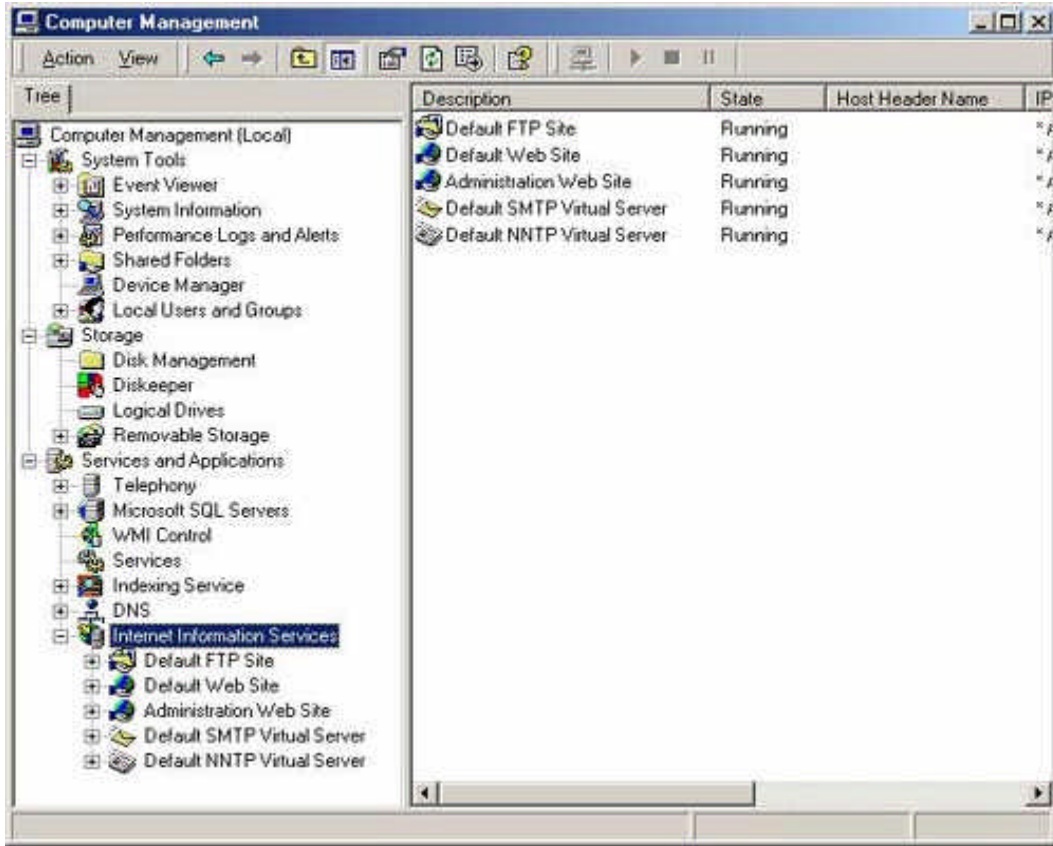
Web sitesine üzerinde dosya ekleme ve silme gibi işlemlerin gerçekleştirilmesi sırasında SSL ile güvenlik sağlanması için bu bölüm işaretlenmelidir.

Allow authors to upload executables :

Web sitesine CGI veya ASP gibi scriptlerin veya diğer çalıştırılabilir dosyaların atılıp atılmayacağına bu bölümden karar verilebilir.

IIS'TE YERALAN SERVİSLER

IIS'e bir önceki makalemizde bahsettiğimiz yollardan birisi ile ulaşıyoruz.



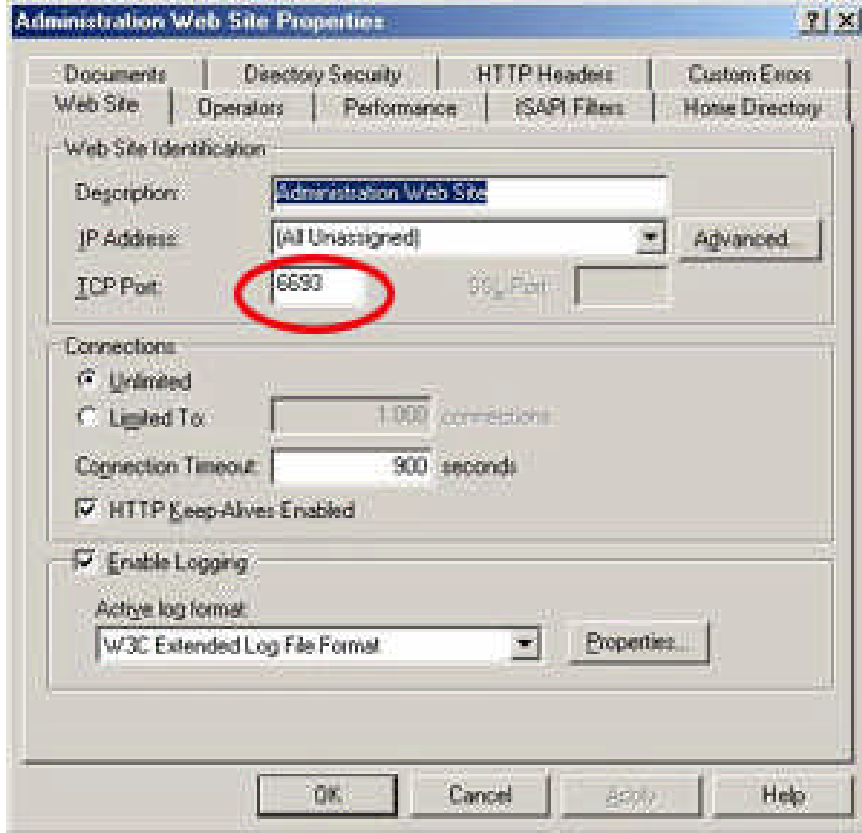
Internet Information Services altında Default Web Site, Administration Web Site ve yaptığınız kurulumu göre Default FTP Site, Default SMTP Virtual Server, Default NNTP Virtual Server servislerini bulabilirsiniz.

Default Web Site, IIS'in kurulumu esnasında oluşturulur. Ana dizin olarak "inetpub/wwwroot" klasörü seçilidir. Eğer bu siteyi hiçbir ayarını değiştirmeden korur ve tüm web sitelerini "inetpub/wwwroot" klasörü altına yerleştirirseniz <http://IP Adresi/klasör adi> şeklinde ulaşabilirsiniz. Default Web Site, otomatik oluşturulmasına rağmen Microsoft tarafından kullanılması pek tavsiye edilmemektedir.

Administration Web Site'da yine IIS kurulumu esnasında oluşturulur. IIS ve oluşturacağınız tüm web ve FTP sitelerinin ayarlarını makinenin başında olmaksızın internet üzerinden yapabilmeyi sağlar. <http://ip adresi:port numarası/> şeklinde ulaşabilirsiniz. Her server veya IIS kurulumunda Administration Web Site için port değişmektedir. Port numarasını sağ tıklayarak açılan menüden "Properties" e basıldığında

Microsoft Visual Basic 6.0

görüntülenen "Administration Web Site Properties" penceresinde yer alan "TCP Port" bölümünden öğrenebilir ve değistirebilirsiniz.

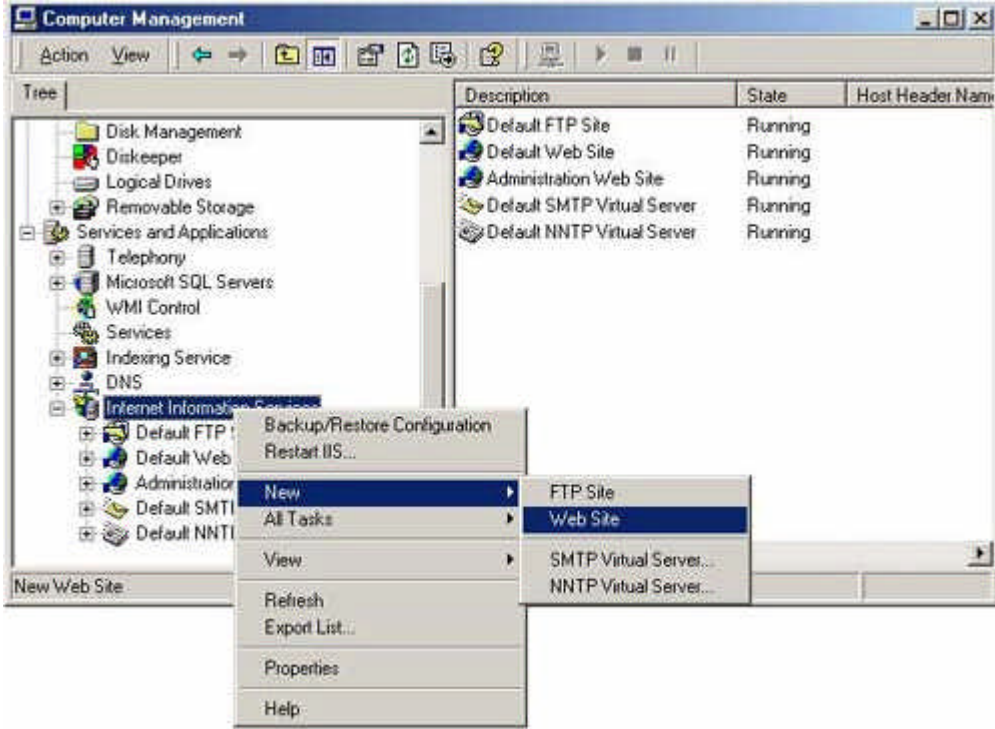


Default SMTP Virtual Server ile serverdan herhangi bir mail server kurmaya gerek kalmadan mail gönderebilirsiniz. Bu servis ile SMTP aracılığıyla mail gönderebilirsiniz, **mail alamazsınız**.

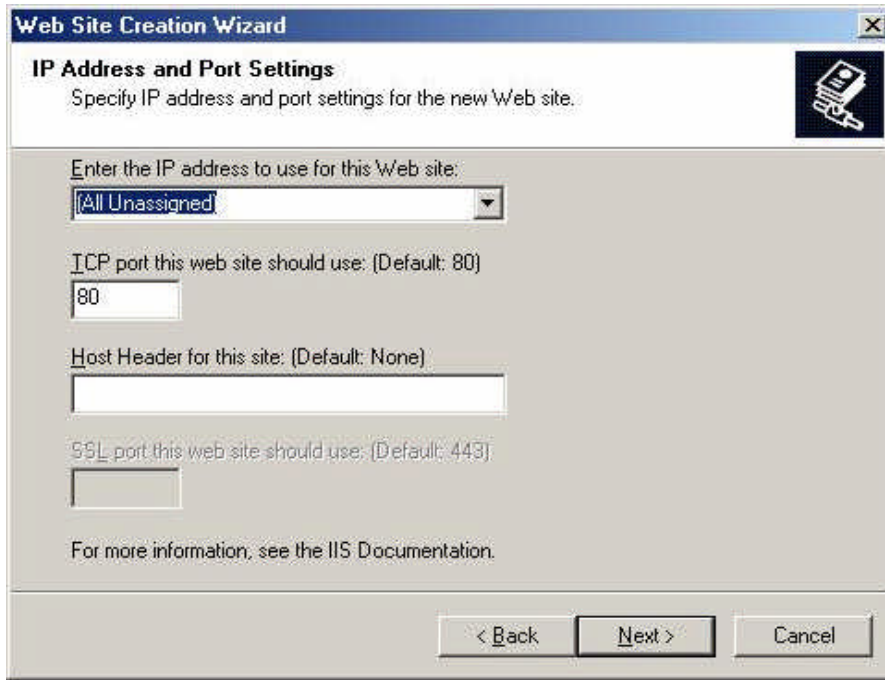
Default NNTP Virtual Server , server üzerinden newsgroup hizmeti vermenizi sağlar.

WEB SITESİ OLUSTURMA

Web sitesi oluşturmak için, Internet Information Services'a sağ tıklayıp açılan menüden New >> Web Site 'a basıyoruz.



Açılacak olan **'Welcome to the Web Site Creation Wizard'** isimli sayfada "Next" tusuna basıyoruz. **"Web Site Description"** sayfasında oluşturacağımız web sitesi için bir açıklama giriyoruz. Örnek : "Test Sitesi". Bu açıklama IIS'te görünecektir (Default Web Site gibi). Tekrar "Next" tusuna basıyoruz. **"IP Address and Port Settings"** sayfasında web sitemiz için IP, port ve header bilgilerini giriyoruz.



The screenshot shows a Windows-style dialog box titled "Web Site Creation Wizard". The subtitle is "IP Address and Port Settings". Below the subtitle, it says "Specify IP address and port settings for the new Web site." There is a small icon of a floppy disk in the top right corner. The main area contains four input fields: 1. "Enter the IP address to use for this Web site:" with a dropdown menu currently showing "All Unassigned". 2. "TCP port this web site should use: (Default: 80)" with a text box containing "80". 3. "Host Header for this site: (Default: None)" with an empty text box. 4. "SSL port this web site should use: (Default: 443)" with an empty text box. At the bottom, there is a line of text: "For more information, see the IIS Documentation." and three buttons: "< Back", "Next >", and "Cancel".

Enter the IP address to use for this Web site : Olusturdugumuz web sitesine ulasilabilecek olan IP adresini burada tanimliyoruz. Bu IP adresi, domainin DNS kayitlarinda yer alan IP adresi ile ayni olmalidir.

TCP port this web site should use : Olusturdugumuz siteye hangi porttan ulasilacagini burada belirliyoruz. Varsayilan olarak bu deger **80** 'dir. Normalde bu port degistirilmez.

Host Header for this site : Olusturdugumuz siteye ulasilmasi için gerekli olan bilgiyi buraya yaziyoruz. Bu genelde **www.siteadi.com** seklindedir. Eger www'den farkli bir header kullanmak istiyorsaniz öncelikle bunu DNS kayitlarina geçirmeniz gerekir. Daha sonra da bu kisma yazabilirsiniz. Ör : **test.siteadi.com**

Tekrar "Next" tusuna basiyoruz. "**Web Site Home Directory**" sayfasinda sitemizin yer aldigi klasörün harddiskteki tam adresini ister elimizle giriyoruz ister "Browse" tusu ile yerini buluyoruz. Adres kutusunun altinda yer alan "**Allow anonymous access to this Web site**" kutusu varsayilan olarak isaretlidir. Böylece internet üzerinden herkes web sitenize ulasabilir. Tekrar "Next" tusuna basiyoruz. "Web Site Access Permissions" sayfasinda web sitesi üzerinde nelerin çalisabilecegini ve ziyaretçilerin web sitesi üzerinde yapabileceklerini belirliyoruz.



Read : Varsayılan olarak işaretlidir. Web sitesinde yeralan tüm sayfaların görüntülenmesi için gereklidir.

Run scripts (such asp ASP) : Varsayılan olarak işaretlidir. Web sitesinde yer alan ASP sayfalarının çalışabilmesi için gereklidir.

Execute (such as ISAPI applications or CGI) : Olusturdugumuz web sitesinde CGI ve ISAPI uygulamalarının çalışması için bu kutuyu işaretliyoruz.

Write : Olusturdugumuz web sitesinde ziyaretçilere dosya yazma hakkı vermek için bu kutuyu işaretliyoruz. **Dikkat : Normalde böyle bir izin verilmez!.**

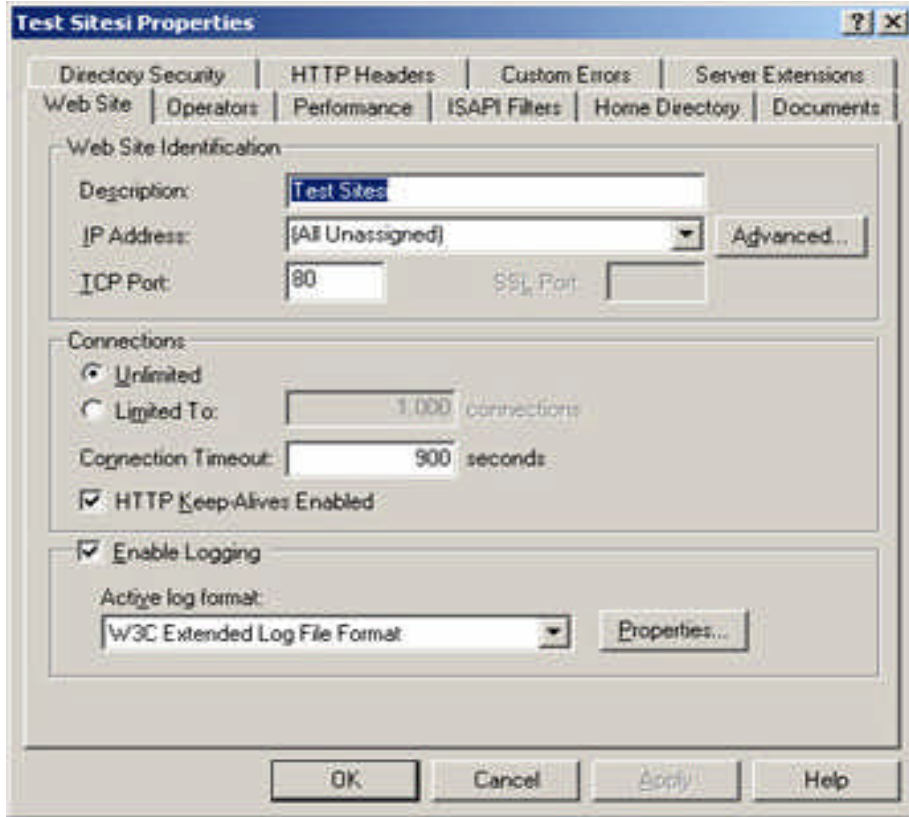
Browse : Olusturdugumuz sitenin klasörlerinde ziyaretçilerin dolasabilmesini sağlamak için bu kutuyu işaretliyoruz. Böylece ziyaretçiler sitede yer alan tüm sayfaları ve resim gibi nesneleri görebilirler. **Dikkat : Normalde böyle bir izin verilmez!.**

Tekrar "Next" tusuna basarak sitenin olusturulma asamasındaki ayarlari bitiriyoruz. Son sayfada "Finish" butonuna tiklayarak basariyla web sitemizi olusturmayi tamamliyoruz.

WEB SITESİNİN AYARLARI

Daha önce bahsettiğimiz yöntemlerden biri ile IIS'e ulaşıyoruz. IIS'de daha önce oluşturdığımız "**Test Sitesi**" isimli web sitesinin üzerine gelip sağ tusa tıklayarak "**Properties**"e basıyoruz.

Web Site



Web Site Identification

Description : Web sitesini oluştururken verdiğimiz ve IIS Manager'da görünen isim.

IP Address : Web sitesine ulaşılabilir IP adresi burada tanımlanır. Makina üzerinde birden fazla ve IP adreslerine sahip site tutulacak ise bunlara ait IP'ler tanımlandıktan sonra bu bölümden ilgili IP adresi seçilir.

TCP Port : Web sitesine ulaşılabilir port numarası. Varsayılan olarak 80'dir. Genelde bu port numarası değiştirilmez. Ancak özel durumlarda, özel

amaçlar için değiştirilir. Port değiştirildiğinde siteye http://www.siteadi.com:port_numarasi/ şeklinde ulaşılabilir.

"Advanced" tusuna basarak açılan "Advanced Multiple Web Site Configuration" isimli pencerede "Multiple identities for this Web Site" bölümünde web sitesine birden fazla isim ile ulaşılması sağlanabilir. Aynı şekilde eğer IIS'e SSL sertifikası eklenmiş ise "Multiple SSL identities for this Web Site" bölümünden web sitesine SSL ile güvenli bir şekilde ulaşılması sağlanabilir.

Connections

Bu bölümde "Unlimited" seçili durumda ise web sitesine bağlanan kişi sayısında bir sınırlama olmaz. "Limited To:" seçili durumda ise sadece belirtilen sayıda ziyaretçi kabul edilecektir.

Connection Timeout: Saniye cinsinden IIS'in aktif olmayan kullanıcılar ile bağlantıyı kesme zamanı. Ziyaretçi tarafında oluşan bir hata sonucu düzgün bir şekilde kapatılamayan bağlantılar belirtilen süre sonunda otomatik olarak kapatılır.

Enable Logging

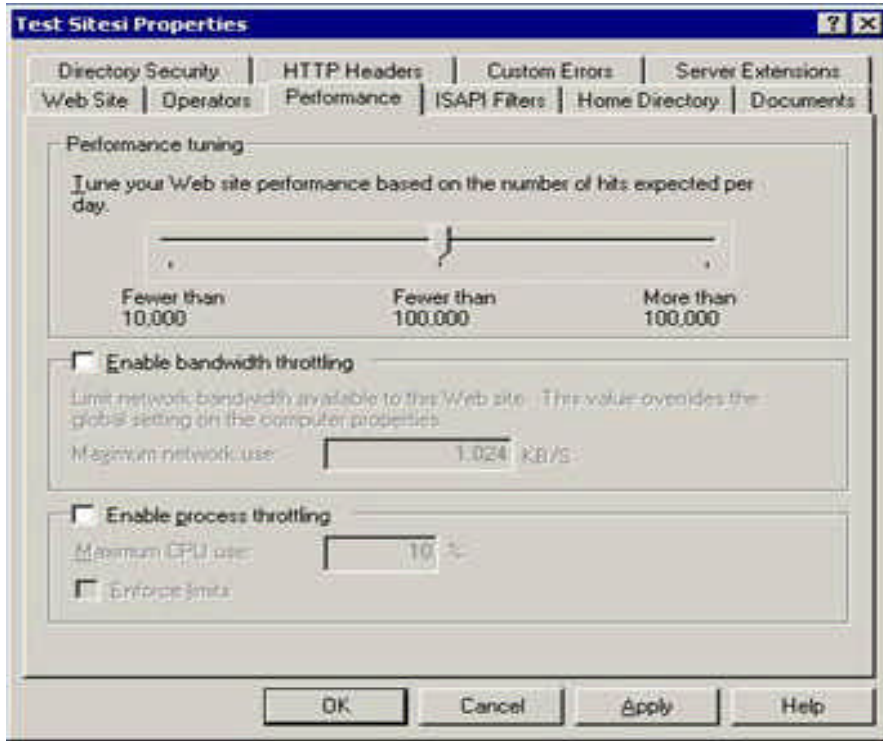
Bu bölümü işaretleyerek web sitesine yapılan tüm ziyaretlerin kaydını tutabilirsiniz. "Active log format" bölümünden Microsoft IIS Log File Format, NCSA Log File Format, ODBC Logging ve W3C Extended Log File Format seçeneklerinden istediğinize uyani seçerek kayıtları tutmaya başlayabilirsiniz. "Properties" tusuna basarak seçmiş olduğunuz log tutma yöntemi ile ilgili ayarları yapabilirsiniz. Böylece hangi sayfaların ne kadar ziyaret edildiğini, web sitesinin en çok hangi saatlerde ziyaret edildiği vb. gibi bilgiler öğrenilebilir.

Operators

Bu bölümde IIS Manager yardımıyla web sitesine ulaşabilecek ve ayarlarına müdahale edebilecek kullanıcılar tanımlanabilir. Varsayılan olarak "Administrators" grubuna dahil olan kullanıcılar IIS Manager'dan web sitesinin ayarlarına müdahale edebilirler. IIS üzerinde tanımlı birden fazla siteden sadece bir tanesi için yetki verilmek istenilen kullanıcı bu bölümden eklenebilir.

Performance

Bu bölümde adından da anlaşılabileceği gibi web sitesinin ne kadar performans ile çalışabileceği belirlenebilir.



Performance tuning

Günlük beklenen hit sayısına göre web sitesinin performansını ayarlamak için kullanılır. Beklenen hit sayısının biraz üzerinde ayarlanarak daha hızlı bir şekilde siteye ulaşılması sağlanabilir. Ancak günlük ziyaretçi sayısının çok üzerinde ayarlanması gereksiz bir şekilde server'in memory(hafıza)'sinin dolmasına ve serverin performansının düşmesine neden olmaktadır.

Enable bandwidth throttling

Web sitesinin kullanacağı bant genişliğini sınırlamak için kullanılır. Saniyede **Kilobyte** cinsinden ne kadar bant genişliği kullanabileceği belirlenir. Bu limit, eğer genel olarak tanımlanmış limitin üzerinde ise genel olan limit geçerli olacaktır.

Enable process throttling

Web sitesinin maksimum CPU kullanımına sınırlama getirmek için kullanılır. Birden çok web sitesinin tutulduğu serverlarda düzgün bir çalışmanın sağlanması için gerekli olabilir. Böylece hatalı çalışan bir web

sitesinin serveri isgörmez hale getirmesi önlenebilir.

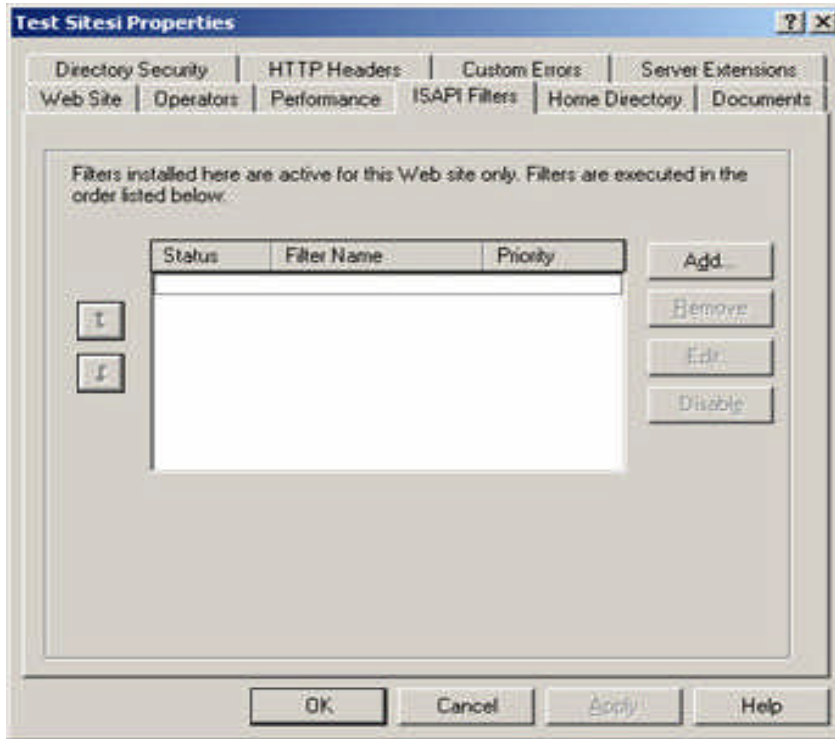
"Enforce limits" isaretlendiginde web sitesinin kısa süreler için limiti asması sağlanabilir.

Limitlerin kontrol süresi 24 saattir. Her 24 saat sonrasında limit asım kontrolleri sıfırlanır.

Örnek : Bir web sitesi %10 CPU kullanımı ile sinirlanmissa ve 24 saat içerisinde, bu sürenin %10'u olan 2.4 saatlik sürede limitleri asmissa Event Log'a bir kayıt eklenir. Web sitesi limitini **% 150** asmis ise ve bu süre 3.6 saat toplamına erismis ise Event Log'a bir kayıt yazilir ve limiti asmasini saglayan işlemler engellenir. Web sitesi limiti % 200 asmis ve bu süre toplam 4.8 saat veya üzeri olmus ise Event Log'a bir kayıt yazilir ve Web sitesinin **çalışması durdurulur**.

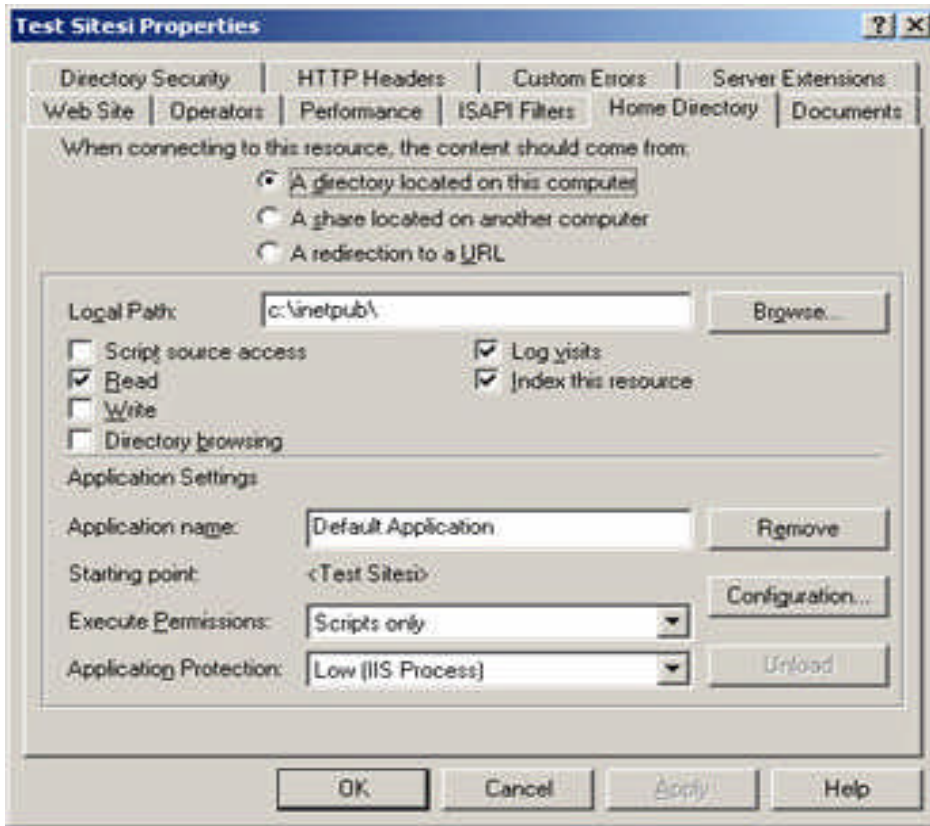
ISAPI Filters

Açılımı **Internet Server Application Programming Interface** olan ISAPI Filtreleri, HTTP isteklerine cevap verebilen programlar olarak tanımlanabilir. ISAPI Filtreleri, DLL dosyalarıdır. **Add** tusunu kullanarak ISAPI Filtresi eklenebilir, **Remove** tusu ile isaretlenen ISAPI Filtresi'ni silinebilir, **Edit** tusu ile ISAPI Filtresi ile ilgili düzenleme yapılabilir. Sol tarafta yer alan ok tuslarını kullanarak ISAPI Filtrelerinin çalışma öncelikleri değiştirilebilir.



Home Directory

Bu bölümde web sitesinine ait dosyaların yer aldığı klasör bilgileri ve uygulama ayarları ile ilgili bölümler yer almaktadır.



When connecting to this source, the connect should come from

Bu kisminda web sitesinin dosyalarinin nerede yer aldigi belirlenir. **"A directory located on this computer"** varsayilan olarak seçilidir ve web sitesi taniminin yapildigi makinada varolan dosyalarin çalistirilmesini saglar. **"A share located on another computer"** seçenegi ile ayni network üzerinde bulunan baska bir makinada yeralan dosyalarin tanimlanan web sitesinde çalistirilmesi saglanir. "A redirecton to a URL" seçenegi yardimiyla da web sitesi baska bir web sitesine yönlendirilebilir.

Local Path

Bu seçenek **"A directory located on this computer"** seçenegi isaretili oldugunda görüntülenir. Web sitesine ait olan dosyalarin yeraldigi klasör seçilerek web sitesinin çalışması saglanir.

Network Directory

Bu seçenek "**A share located on another computer**" seçeneği işaretli olduğunda görüntülenir. "**\\makina_adi\paylasim_adi**" şeklinde diğer makinada yer alan dosyalara ulaşılabilir.

Redirect to

"**A redirection to a URL**" seçeneği işaretli olduğunda görüntülenir. "**http://www.xxx.com**" şeklinde başka bir siteye yönlendirme sağlanır.

Script source Access

Bu seçenek işaretlendiğinde Read veya Write hakkı verilmiş olan kullanıcıların kaynak koduna ulaşması sağlanabilir. **Normal şartlarda kullanılması tavsiye edilmez.**

Read

Web sitesine bağlanacak kullanıcıların dosyaları okuması veya çalıştırabilmesi için verilmesi gerekir.

Write

Web sitesine bağlanan kullanıcıların dosyalar üzerinde değişiklik yapabilmesine, yeni dosya upload edebilmesine olanak sağlar. **Normal şartlarda kullanılması tavsiye edilmez.** Mutlaka write hakkı verilmesi gerekiyorsa ilgili klasörde bu hakkın verilmesi daha sağlıklıdır.

Directory browsing

Bu seçenek işaretlenerek web sitesine bağlanan kullanıcıların ilgili klasörde varolan dosyaların bir listesini görmesi sağlanır. Ancak bu klasörde default.asp, default.html, index.asp gibi klasör adresi verildiğinde çalışan dosyalar varsa klasör içeriği görüntülenemeyecektir. Gerekli olmadığı sürece bu seçeneğin aktif olması tavsiye edilmez.

Log visits

Bu seçenek varsayılan olarak işaretlidir ve ziyaretçilerin yaptıkları tüm işlemlerin kaydının tutulmasını sağlar.

Index this source

Bu seçenek yardımıyla Microsoft Indexing Service kullanılarak **full-text search** yapılabilmesi sağlanır.

"**A redirection to URL**" seçeneği işaretli olduğunda aşağıdaki seçenekler görüntülenecektir :

The exact URL entered above

Girilmiş olan tam adrese gidilmesini sağlar. Böylece web sitesinin yönlendirildiği sitede istenilen herhangi bir alt sayfadan başlanması sağlanabilir.

A directory below this one

Bu seçenek yardımıyla web sitesinin kendi içerisindeki herhangi bir alt klasöre yönlendirilmesi sağlanabilir. Ör : **/yenisite** yazıldığında www.test.com yazıldığında sayfa **www.test.com/yenisite** adresine yönlendirilecektir.

A permanent redirection for this source

Web sitesine bağlanan kullanıcıya yönlendirme ile ilgili bir mesaj gönderir.

Application

Settings

Application name

Web sitesinin bulunduğu uygulama bölümünün adı.

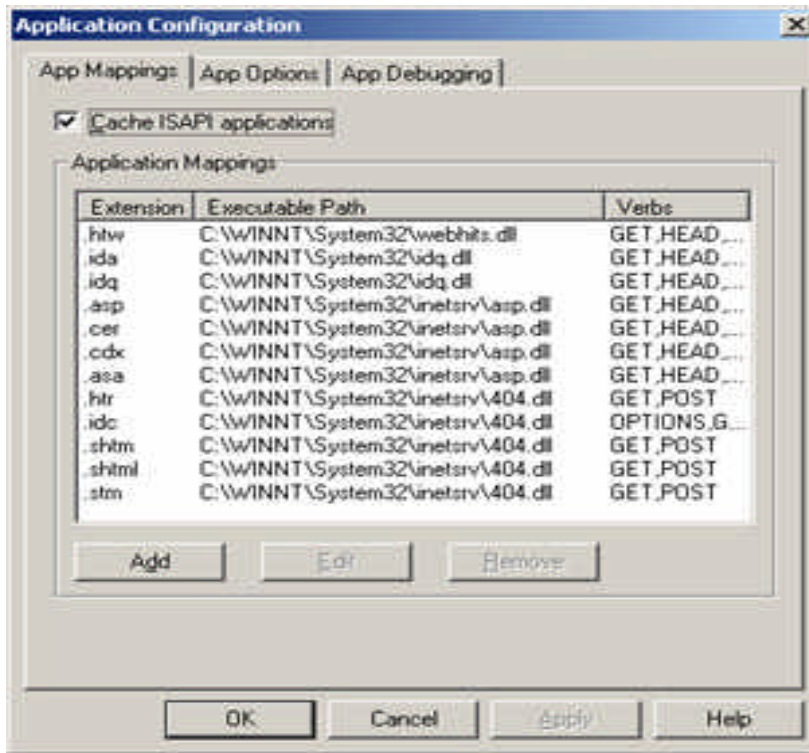
Execute Permissions

Bu bölümde çalıştırılacak dosya tipleri belirlenir. **None** işaretlendiğinde sadece statik HTML sayfaları ve resim dosyaları görüntülenebilir. **Scripts only** seçeneği işaretlendiğinde ASP gibi script dosyalarının çalıştırılması sağlanır. **Scripts and Executables** işaretli ise her türlü dosya çalıştırılabilir hale gelir.

Application Protection

Web sitelerinin yaptigi islemlerin nasil gerceklestirilecegi bu bölümde belirlenir. **Low(IIS process)** isaretlendiginde tüm islemler ayni süreç içerisinde çalistirilir. **Medium(Pooled)** seçenegi isaretlendiginde islemler **application** bazinda ayri olarak yürütölür. **High(Isolated)** seçenegi isaretlendiginde ilgili web sitesinin islemleri tamamen ayri olarak yürütölür.

Configuration tusuna basildiginda **Application Configuration** isimli pencere görölntölünir.



App Mappings

Cache ISAPI applications

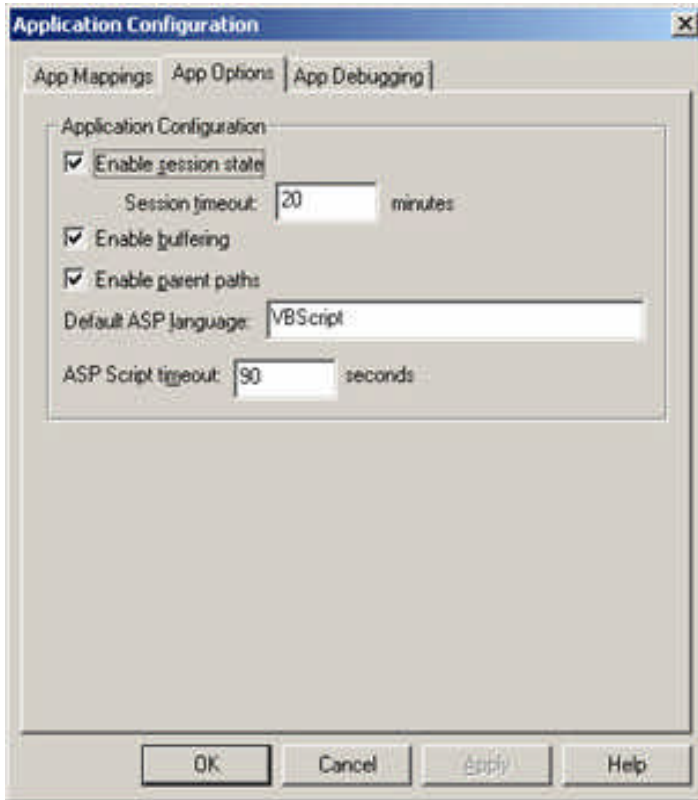
Bu seçenek isaretleterek çalistirilan ISAPI filtrelerinin cache'lenmesi ve daha sonraki kullanimlarda daha hizli islem yapilmasi saglanir.

Applicaton Mappings

Bu listede hangi uzantiya sahip olan dosyanin hangi DLL ile iliskili oldugu, hangi islemlerin gerceklestirilebilecegi görüntülenmektedir. Yapılmak istenilenlere göre bu listede ekleme, çıkarma veya düzenleme islemleri yapılabilir.

App Options

Bu bölümde ASP sayfalarinin nasıl çalışacağı ile ilgili ayarlar yapılabilir.



Enable session state

Varsayılan olarak isaretlidir. Web sitesinde session ile ilgili islemlerin yapılabilmesini sağlar.

Session timeout

Dakika olarak session ile ilgili bilgilerin tutulacağı zamanı gösterir. Varsayılan olarak 20 dakikadır. Amaca uygun olarak değiştirilebilir. Bu süre script içerisinde **Session.Timeout = xx** olarak değiştirilebilir.

Enable buffering

Varsayılan olarak isaretlidir. Bu seçenek ASP sayfaların işletilmesiyle çıkan sonuçların tamponlanmasını sağlar. Seçenek isaretlendiğinde ASP sayfaları browser'a gönderilmeden önce tamponlanır. Böylece aynı istek tekrarlandığında daha hızlı yanıt verilebilir. Ancak zaman zaman sayfaların her istenildiğinde tekrar üretilmesi gerekebilir. Bu durumda bu seçenekteki isaret kaldırılır.

Enable parent paths

Bu seçenek isaretleterek ASP dosyaları içerisinde göreceli yolların kullanılması sağlanabilir. Bulunulan klasörden üst klasördeki bir dosyaya ".." şeklinde ulaşılabilmesi sağlanır. Ör : şeklindeki bir yazım ile images klasöründe yer alan resim.gif dosyası gösterilebilir. Varsayılan olarak isaretlidir.

Default ASP language

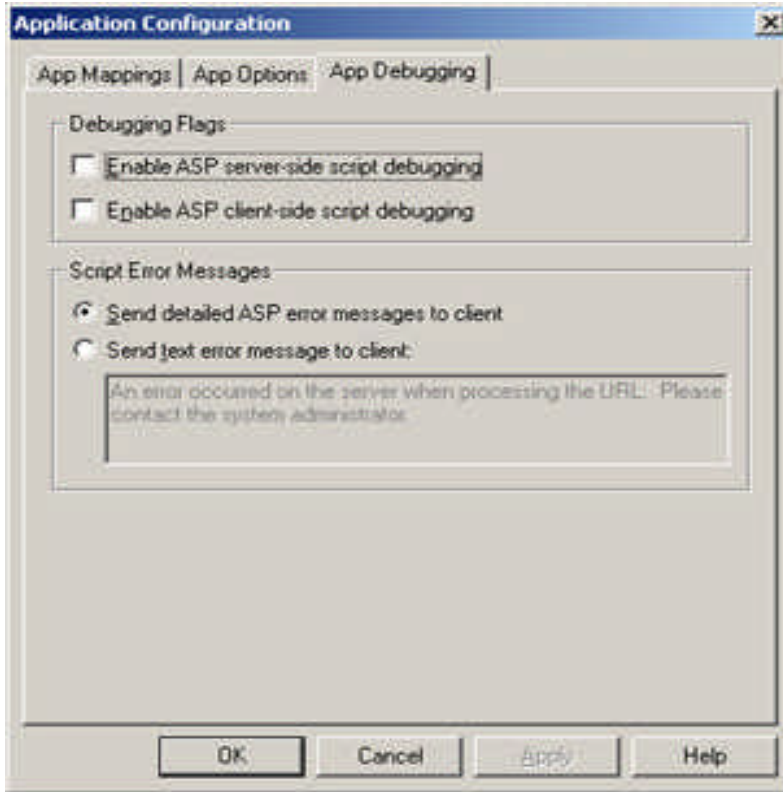
Web sitesinde ASP scriptlerinin yazımında kullanılacak olan script dilinin belirlenmesi için kullanılır. Varsayılan olarak VBScript tanımlıdır. Genel olarak dünya çapında kabul görmüştür. Farklı bir dil kullanılacaksa bu kısımdan değiştirilebilir. Eğer herhangi bir sayfada, sadece o sayfa için başka bir dil kullanılacaksa sayfanın ilk satırında yazılmalıdır.

ASP Script timeout

ASP kodlarının çalıştırılacağı zaman dilimini saniye cinsinden gösterir. Varsayılan olarak 90 saniyedir. Bu süre sonucunda ASP kodlarının çalışması sonlanmış olmalıdır. Eğer sonlanmamış ise kodların çalışması durdurulur, browser'da ilgili hata mesajı gösterilir ve Event Log'a kayıt eklenir. Sadece belirli bir sayfa için farklı bir timeout süresine ihtiyaç varsa bu sayfa içerisinde **Script.Timeout = xx** şeklinde belirtilebilir.

App Debugging

Bu bölümde ASP sayfalarının çalışması sırasında oluşan hataların ayıklanabilmesi ile ilgili işlemler gerçekleştirilebilir.



Debugging Flags

Enable ASP server-side script debugging

Bu seçenek işaretlendiğinde ASP sayfalarının çalışması sırasında Microsoft Script Debugger'da devreye girecektir. Microsoft Script Debugger kullanılarak kodlarda oluşan hatalar incelenebilir. Ancak bu durumda ASP kodları single-threaded (her bir işlemler sırayla yapılır) olarak çalışır ve performans kaybı yaşanır. Bu sebeple **bu işlem normal uygulamalarda tavsiye edilmez**. Uygulama geliştirme sırasında kullanılabilir.

Enable ASP client-side script debugging

Bu seçenek ASP 3.0 versiyonunda hiçbir etki yapmamaktadır ve daha sonraki sürümler için eklenmiştir.

Script Error Messages

Bu bölümde ASP kodlarının çalışması sırasında bir hata oluştuğunda ziyaretçiye gösterilecek mesaj ile ilgili ayarlamalar yapılabilir.

Send detailed ASP error messages to client

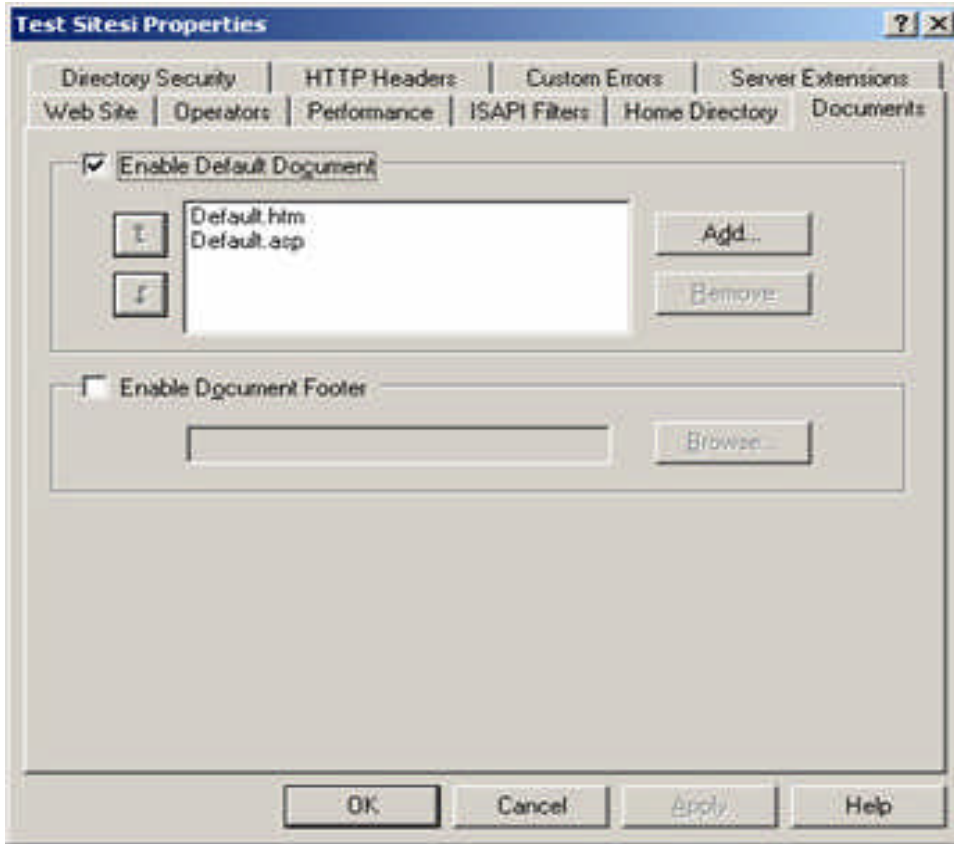
Varsayılan olarak isaretlidir. Hata oluştuğunda hata ile ilgili tüm açıklamaların ziyaretçinin browser'ında görünmesini sağlar. Bu açıklamalar hatanın olduğu satır, hatanın olduğu ASP sayfası, kod ile ilgili açıklama vs. içerir.

Send text error message to client

ASP kodlarının çalışması sırasında bir hata oluştuğunda istenilen mesajın ziyaretçiye ulaştırılmasını sağlar. Böylece hata ile ilgili ayrıntılar ziyaretçiden gizlenmiş olur.

Documents

Bu bölümde Web sitesinin çalışması sırasında varsayılan olarak çalıştırılacak sayfalar belirlenebilir.



Enable Default Document

Varsayılan olarak **Default.asp** ve **Default.htm** sayfaları yeralmaktadır. İsteğe göre farklı dosyalar eklenebilir.

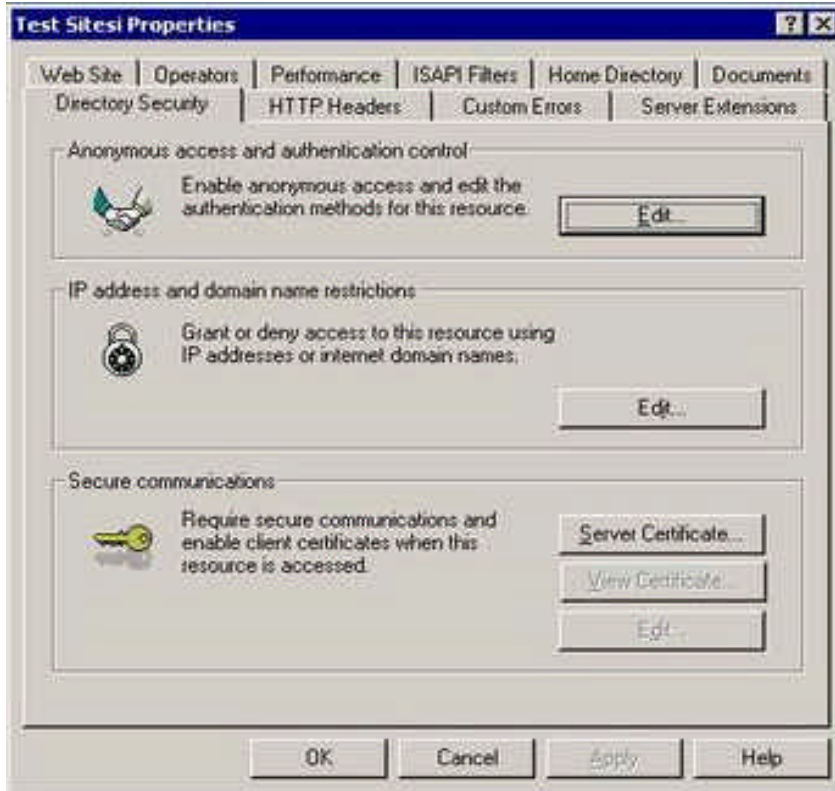
Enable Document Footer

Bu seçenek işaretlenerek Web sitesinde yer alan herhangi bir dosya çalıştırıldığında sayfanın altında HTML formatlı bir mesajın görüntülenmesi sağlanabilir.

WEB SITESİNİN AYARLARI -II

Daha önce bahsettiğimiz yöntemlerden biri ile IIS'e ulaşıyoruz. IIS'de daha önce oluşturdığımız "**Test Sitesi**" isimli web sitesinin üzerine gelip sağ tusa tıklayarak "**Properties**"e basıyoruz.

Directory Security

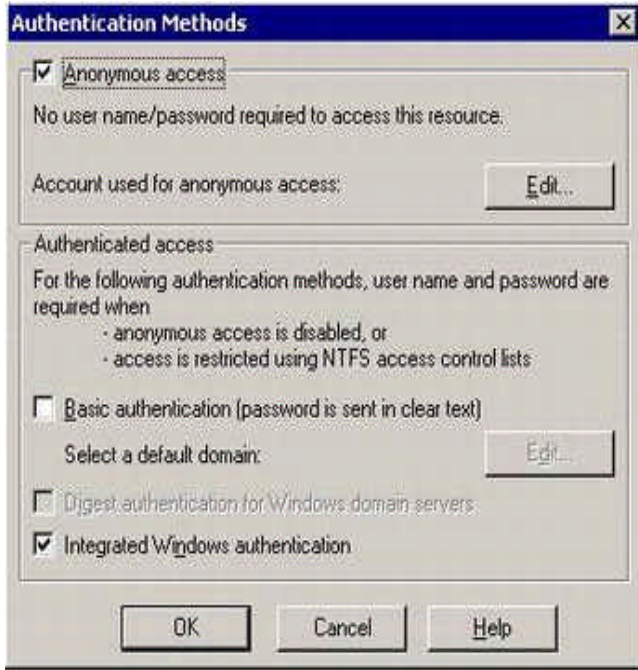


Bu bölümde web sitesi ve klasörlerine erişim ve güvenli erişim sertifikaları ile ilgili ayarlar belirlenebilir.

Anonymous access and authentication control

Bu kısımda erişimleri için yetkilendirme (authentication) ve anonim bağlantılar (anonymous access) özellikleri ayarlanabilir. Böylece özel bölümlere herkesin erismesi engellenebilir, erisebilecek kişiler tanımlanabilir. Herkesin web sitesine erisebilmesi isteniyorsa bir ayar yapma gereği bulunmamaktadır. Ancak belirli bölümlere tanımlanmış kişilerin erismesi

isteniyorsa o zaman ayarların yapılması gerekmektedir. Ayarları yapmak için "Edit" tusuna basıyoruz.



Bu pencerede de görüldüğü gibi web sitesine anonim ve yetkilendirilmiş olarak 2 farklı erişim vardır. Yetkilendirilmiş erişim genelde sadece belirli bölümlerde kullanılır.

Anonymous Access

Varsayılan olarak bu seçenek işaretlidir. Bu şekilde web üzerinden sitesine gelen ziyaretçilerin sayfalarını görüntülemesi sağlanır. Ziyaretçiler, site ziyaretleri sırasında **IUSR_makinadi** isimli kullanıcı ile bağlanmış olarak görünürler. Bu kullanıcı IIS ile beraber sistemde oluşturulmaktadır.

Edit tusuna basıldığında **Anonymous User Account** isimli pencere görüntülenir.



Bu ekranda resimde de görüldüğü gibi anonim bağlantılar için IUSR_makinaadi kullanicisi seçili durumdadır. **Allow IIS to control password** kutusu da varsayılan olarak isaretlidir. Bu ayarda bir degisiklik gerçekleştirilmedigi sürece anonim bağlantılar problemsiz olarak gerçekleseyecektir. Anonim bağlantıların gerçekleştirildiği kullanının adı degismis ise **Username** kısmına yazılarak veya **Browse** tusu yardimiyla kullanıcı bulunarak degisiklik aktif hale getirilebilir. **Aksi takdirde site düzgün çalışmayacaktır.**

Authenticated Access

Genellikle yetkiye sahip belirli kisilerin ulasmasi istenilen web siteleri veya web sitesi bölümleri için kullanılır. Burada bahsedilen yetkilendirme **NTFS dosya sistemi vasitasiyla saglanilan web sitesi, klasör veya dosya bazinda** yetkilendirmedir.

Yetkilendirme için kullanılan kullanıcılar **Windows** kullanıcılarıdır. Dolayısıyla yetkilendirilmiş erişim için öncelikle sistemde bu kullanıcıların tanımlanması ve ilgili klasörlerde gerekli haklarının verilmiş olması gerekmektedir.

Not : Kullanıcıların belirtilen alanlara web üzerinden yetkilendirilmiş erişimle ulaşabilmeleri için, ilgili makineye logon yetkilerinin olması ve kullanıcı hesabının aktif olması gerekmektedir.

Bu yetkilendirme türü aktif edildiğinde ziyaretçiler web sitesine erişmek istediklerinde kullanıcı adı ve şifre girilmesini isteyen bir pencere görüntülenir. İstenilen bilgiler doğru girildiğinde sayfa görüntülenir. Aksi takdirde hata mesajı içeren sayfa ekrana gelir.

Basic authentication : IIS tarafında yetkilendirme için en çok kullanılan yöntemdir. Girilen kullanıcı adı ve şifre bilgileri normal text halinde **sifrelenmemiş** olarak gönderilir. Bu yetkilendirme daha çok internet üzerinde kullanılır, intranet üzerinde "Integrated Windows authentication" ile kullanmak daha mantıklıdır.

Digest authentication for Windows domain servers : Bu yöntem IIS 5.0 ile beraber gelmektedir. Girilen kullanıcı adı ve şifre bilgileri özel bir algoritma ile **sifrelenmiş** olarak gönderilir.

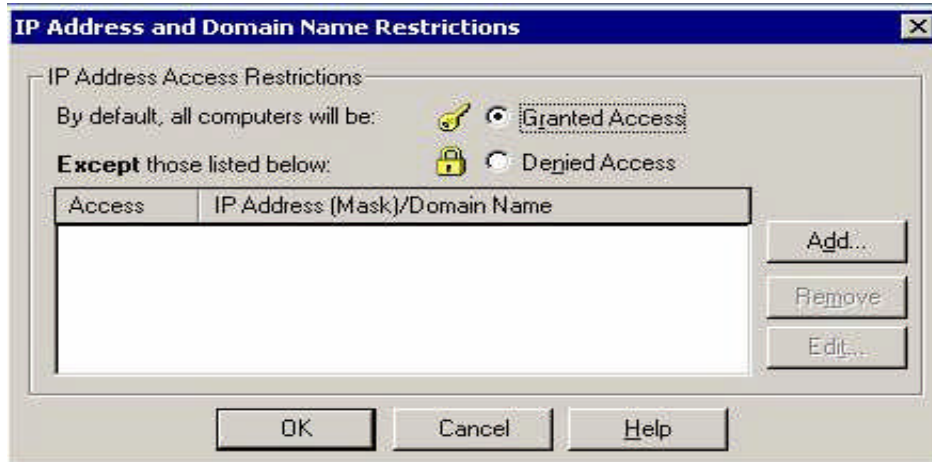
Integrated Windows authentication : Windows'un normal kullanıcı işlemleri sırasında kullandığı yetkilendirme metodunu kullanır. Genelde intranette ve domain yapısı içeren networklerde kullanılır. Internet üzerinden de kullanılabilir.

Bu yöntemin 2 kısıtlaması vardır :

1. Sadece Internet Explorer ile çalışır, diğer browser'lar ile çalışmaz. Internet Explorer versiyonu 2.0 ve üzeri olmalıdır.
2. Proxy sunucuları ve HTTP Proxy ile sağlanan bağlantılarla çalışmaz.

IP address and domain name restrictions

Bu bölüm yardımıyla web sitesine **erismesi** veya **erismemesi** istenilen IP adresleri, bilgisayarlar veya kullanıcılar tanımlanabilir. Bu özellik **sadece server** sistemlerde çalışmaktadır. Edit tusuna basıldığında "**IP Address and Domain Name Restrictions**" isimli pencere görüntülenir.

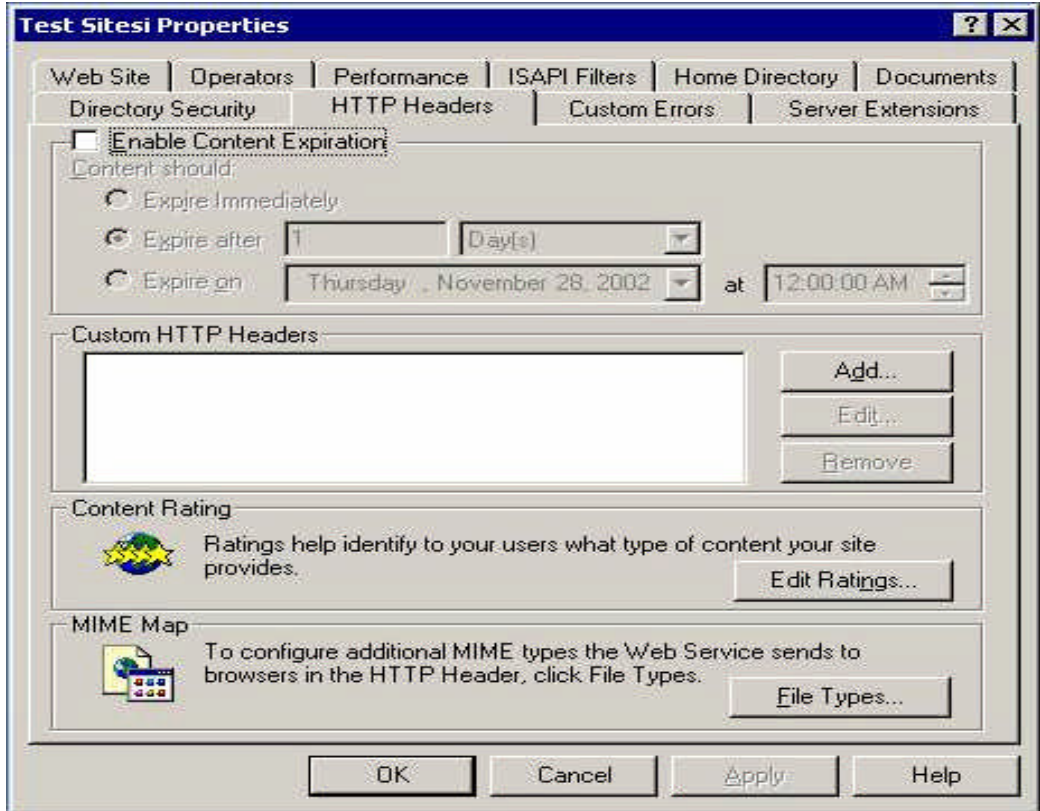


Pencerede görülecegi gibi varsayılan olarak web sitesi herkese açıktır. **Granted Access** isaretleli iken **Add** tusu yardımıyla IP adresi, IP adres grubu veya domain adi eklenirse; **eklenen bilgilere sahip** kullanıcılar web sitesine **erisemeyecektir**. Eger **Denied Access** isaretleli duruma getirilir ise o zaman da **sadece listeye eklenmiş** olan kullanıcılar web sitesine **erisebilecektir**.

Secure communications

Bu bölüm, güvenli bağlantıların gerektiği durumlarda (ör: internet bankacılığı, şirkete ait gizli bilgilere erişilmesi vb.) kullanılır. SSL sertifikaları bu bölüm yardımıyla eklenerek güvenli bağlantı kurulması sağlanır.

HTTP Headers



Enable Content Expiration

Bu bölümde Enable Content Expiration bölümünü işaretlenerek web sitesindeki tüm sayfaların belirli bir süre sonra expire (süresi dolması) olması sağlanabilir.

Bu özellik daha çok sürekli değişen sayfalar için kullanılabilir.

Custom HTTP Headers

Adından da anlaşılabileceği gibi web sitesine özel olarak bilgiler tanımlanabilir. Bu tanımlanan bilgiler server'dan client'a gönderilir.

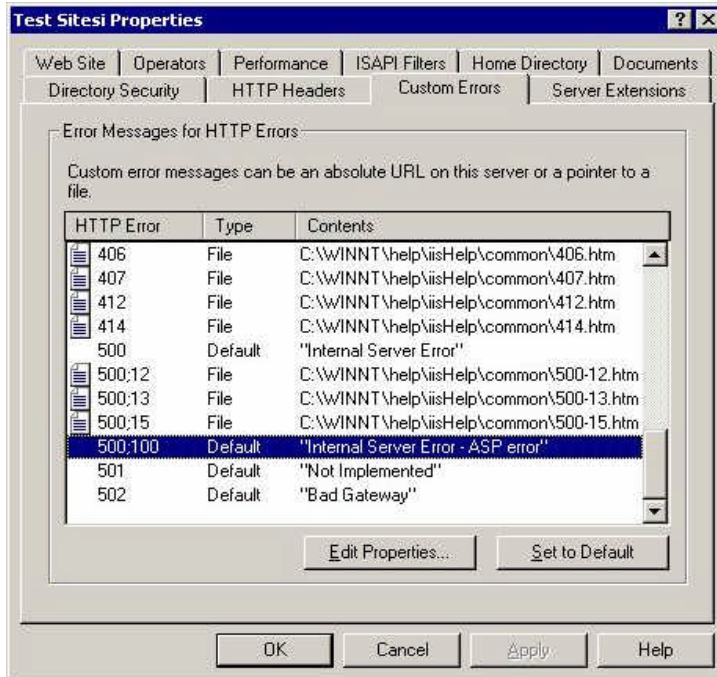
Content Rating

Web sitesinde barındırılan özel içerik (yetişkin vs.) ve seviyesi tanımlanarak bu bilgiler HTTP header bilgileriyle client'a gönderilir. Böylece kişilerin istenmeyen içeriğe erişmesi engellenebilir.

MIME Map

Web sitesinde kullanılmak istenilen farklı dosya tipleri bu bölümden eklenebilir.

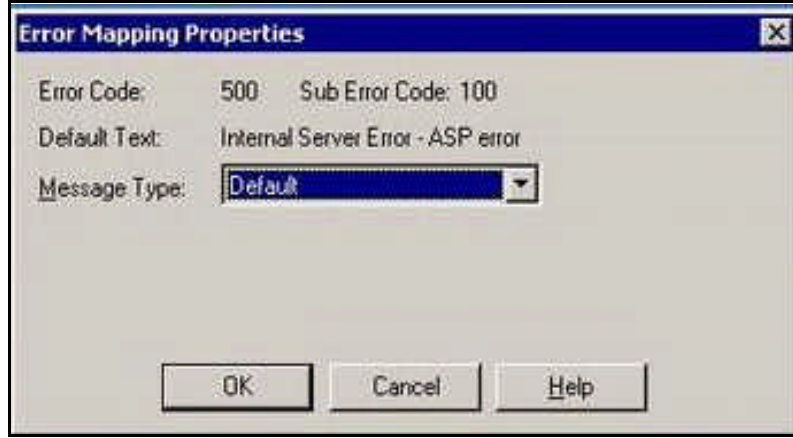
Custom Errors



Microsoft Visual Basic 6.0

Bu bölümde web sitesinde oluşan hatalar sonucunda kullanıcıya gösterilen sayfalar değiştirilebilir. Böylece kullanıcıya istenilen şekilde mesajlar verilebilir.

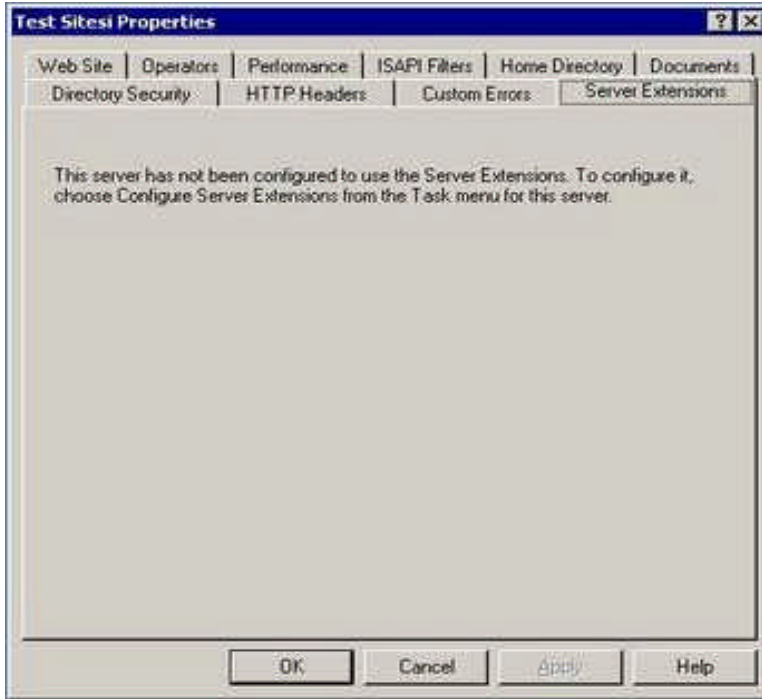
Değiştirilmek istenilen mesaja ait link seçili hale getirilerek **Edit Properties**'e basıldığında **Error Mapping Properties** isimli pencere görüntülenir.



Message Type kısmından isteye uygun seçim yapılarak istenilen mesajlar görüntülenebilir.

Default'da bırakıldığında varsayılan hata mesajı görüntülenir. **URL** seçildiğinde web sitesinin bulunduğu makinede **herhangi bir klasör altında** yeralan dosyayı tanımlayabilmek için öncelikli olarak dosyanın bulunduğu klasör **virtual directory** olarak eklenmelidir. (**Ör:** C:\test klasöründe yeralan my500.asp isimli dosyayı kullanmak için test isimli bir virtual directory oluşturulur ve adres /test/my500.asp olarak verilir.) File seçildiğinde makina üzerinden direkt olarak ilgili dosya bulunabilir veya adresi yazılabilir.

Server Extensions



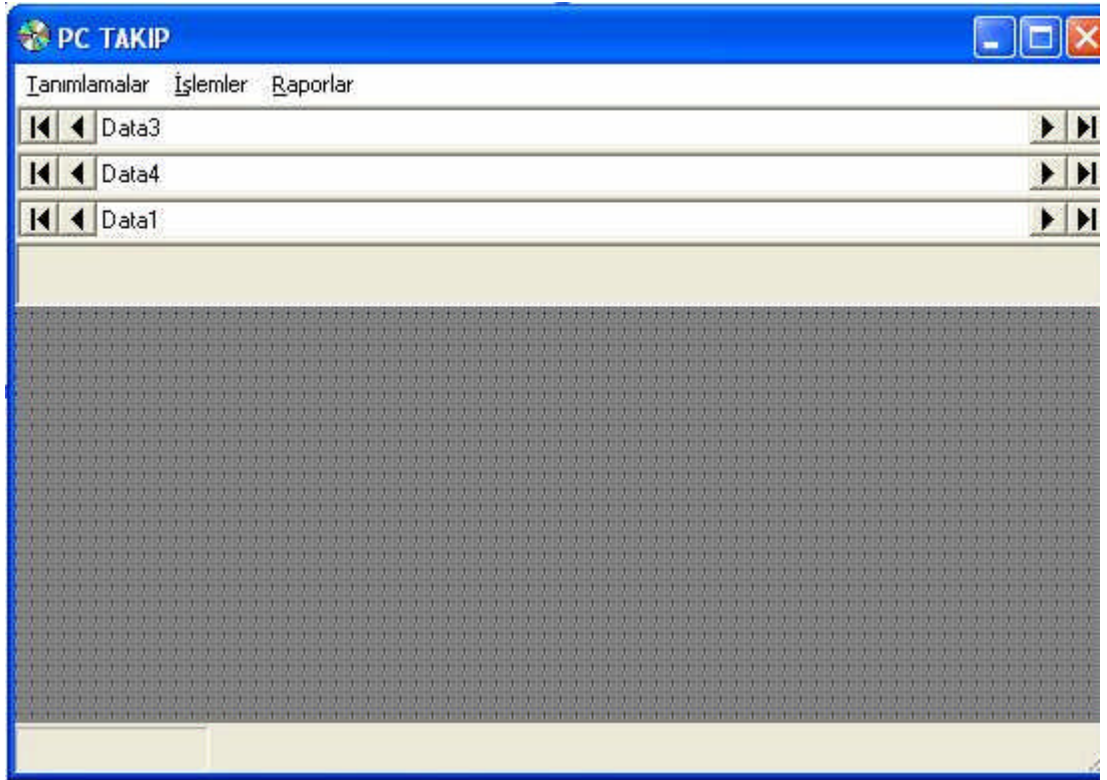
Bu bölüm Frontpage Server Extensions (sunucu uzantıları) kullanılabilmesini ve ayarlarının yapılmasını sağlar. Frontpage Server Extensions yardımıyla web sitesindeki dosyalar direkt olarak sunucuya erişilerek, programcının kendi makinasındaymış gibi düzenlenebilir. Dosyaların FTP veya diğer yöntemlerle sunucuya gönderilmesi gerekmez.

Bu özelliği kullanabilmek için Frontpage Server Extensions'in sunucu üzerine kurulması gerekir.

Bu özelliğin kullanılması direkt olarak dosyalara erişim olduğu için güvenlik açıkları meydana getirebilir. Çok gerekmedikçe kullanılması tavsiye edilmez.

PC TAKIP PROGRAMI

MDI FORM



```

Private Sub bilgisayartoplama_Click()
On Error GoTo hata

SQL = "select * from bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
Form4.Text3.Text = 1

hata:
Select Case Err
Case Is = 3021
MsgBox "Tanimli Bölüm olmadığı için giris yapılamaz.!"
End Select
End Sub

Private Sub bolum_Click()
Form1.Show
    
```

Microsoft Visual Basic 6.0

```
Form1.Caption = "Bölüm Tanımlama"  
Form1.Label1.Caption = "Olusturulmak istenen Bölüm adini  
giriniz..!"
```

```
End Sub
```

```
Private Sub Bolumad1_Click()
```

```
On Error GoTo hata  
SQL = "select * from bolum"  
Data1.RecordSource = SQL  
Data1.Refresh  
Data1.Recordset.MoveLast  
Form6.Show
```

```
hata:
```

```
Select Case Err  
Case Is = 3021  
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz..!"  
End Select
```

```
End Sub
```

```
Private Sub cikis_Click()
```

```
End
```

```
End Sub
```

```
Private Sub MDIForm_Load()
```

```
Dim i As Integer  
For i = 1 To 5  
    StatusBar1.Panels.Add  
Next i  
  
With StatusBar1.Panels  
    .Item(1).AutoSize = sbrContents  
    .Item(1).Text = "PC Takip Programi ZINNET KALAÇ"  
    .Item(1).MinWidth = 7000  
    .Item(2).Style = sbrCaps  
    .Item(2).MinWidth = 600  
    .Item(3).Style = sbrNum  
    .Item(3).MinWidth = 600  
    .Item(4).Style = sbrIns  
    .Item(4).MinWidth = 600  
    .Item(5).Style = sbrTime  
    .Item(5).MinWidth = 600  
    .Item(6).Style = sbrDate
```

```
End With
```

```
End Sub
```

```
Private Sub parca_Click()
```

```
On Error GoTo hata
```



```
SQL = "select * from bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
If Data1.Recordset.RecordCount <= 0 Then
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz.!"
Exit Sub
End If
Form2.Show
```

```
hata:
Select Case Err
Case Is = 3021
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz.!"
End Select
End Sub
```

```
Private Sub parcabilgisi0_Click()
On Error GoTo hata
```

```
SQL = "select * from bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
Form5.Text1.Text = 1
```

```
hata:
Select Case Err
Case Is = 3021
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz.!"
End Select
End Sub
```

```
Private Sub parcabilgisi1_Click()
On Error GoTo hata
```

```
SQL = "select * from bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
Form5.Text1.Text = 2
```

```
hata:
Select Case Err
Case Is = 3021
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz.!"
End Select
End Sub
```

Microsoft Visual Basic 6.0

```
Private Sub Parcafiyati_Click()
```

```
On Error GoTo hata
```

```
SQL = "select * from bolum"
```

```
Data1.RecordSource = SQL
```

```
Data1.Refresh
```

```
Data1.Recordset.MoveLast
```

```
Form4.Text3.Text = 0
```

```
hata:
```

```
Select Case Err
```

```
Case Is = 3021
```

```
MsgBox "Tanimli Bölüm olmadigi için giris yapilamaz.!"
```

```
End Select
```

```
End Sub
```

```
Private Sub pccc_Click()
```

```
On Error GoTo hata
```

```
tar = InputBox("Rapor alınmak istenen tarihi giriniz.!", "PC satis  
raporu", "##.##.####")
```

```
SQL = "select * from satis where tarih='" & tar & "'"
```

```
Data4.RecordSource = SQL
```

```
Data4.Refresh
```

```
Data4.Recordset.MoveLast
```

```
Form7.Text1.Text = tar
```

```
Form7.Text2.Text = 2
```

```
hata:
```

```
Select Case Err
```

```
Case Is = 3021
```

```
End Select
```

```
End Sub
```

```
Private Sub pparcca_Click()
```

```
On Error GoTo hata
```

```
tar = InputBox("Rapor alınmak istenen tarihi giriniz.!", "Parça  
satis raporu", "##.##.####")
```

```
SQL = "select * from parca where tarih='" & tar & "'"
```

```
Data3.RecordSource = SQL
```

```
Data3.Refresh
```

```
Data3.Recordset.MoveLast
```

```
Form7.Text1.Text = tar
```

```
Form7.Text2.Text = 1
```

```
hata:
```

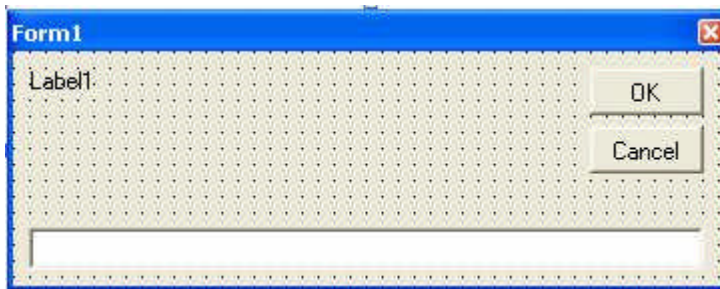
```
Select Case Err
```

```
Case Is = 3021
```

```
End Select
End Sub

Private Sub Satisrap_Click()
Form7.Show
End Sub
```

FORM 1



```
Private Sub bolum_kayit()
Data1.Recordset.AddNew
Data1.Recordset.bolum = Text1.Text
Data1.Recordset.Update
cevap = MsgBox("Olusturulan bölümle ilgili parça girisi yapmak  
istiyormusunuz.?", 4 + 32 + 256)
If cevap = 6 Then
Form2.Show
Form2.Combo1.Text = Text1.Text
Form2.Combo2.SetFocus
Unload Form1
Else Unload Form1 End If
End Sub

Private Sub CommandButton1_Click()
On Error GoTo hata
SQL = "select * from bolum where bolum='" & Text1.Text & "'"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
MsgBox "Bu Bölüm tanımlı. Lütfen dikkat ediniz.!"
Text1.Text = "" Text1.SetFocus
hata: Select Case Err
```

Microsoft Visual Basic 6.0

```
Case Is = 3021
bolum_kayit
End Select
End Sub
```

```
Private Sub CommandButton2_Click()
Unload Me
End Sub
```

```
Private Sub Form_Activate()
Form1.Left = 3390
Form1.Top = 2475
Text1.SetFocus
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Text1.Text <> "" Then CommandButton1.SetFocus
End Sub
FORM 2
```

```
Private Sub kaytdeg()
SQL = "select * from parca where bolum='" & Text6.Text & "' and
firma='" & Text7.Text & "' and model='" & Text8.Text & "'"
Data6.RecordSource = SQL
Data6.Refresh
Data6.Recordset.MoveFirst
Data6.Recordset.Edit
Data6.Recordset.bolum = Combo1.Text
Data6.Recordset.firma = Combo2.Text
Data6.Recordset.model = Text1.Text
Data6.Recordset.doviz1 = MaskedTextBox4.FormattedText
If Option1.Value = True Then Data6.Recordset.kur = 1
If Option2.Value = True Then Data6.Recordset.kur = 0
Data6.Recordset.Update
MsgBox "Degisiklik Tamamlandi..."
Unload Form2
End Sub
```

```
Private Sub kur_sapta()
```

```
SQL = "select * from kur where tarih='" & Date & "'"
Data5.RecordSource = SQL
Data5.Refresh
Data5.Recordset.MoveLast
MaskedTextBox7.Text = Data5.Recordset.dolar
MaskedTextBox8.Text = Data5.Recordset.mark
End Sub
```

```
Private Sub kayt()
Data4.Recordset.AddNew
Data4.Recordset.bolum = Combo1.Text
Data4.Recordset.firma = Combo2.Text
Data4.Recordset.model = Text1.Text
Data4.Recordset.doviz1 = MaskedTextBox4.FormattedText
If Option1.Value = True Then Data4.Recordset.kur = 1
If Option2.Value = True Then Data4.Recordset.kur = 0
Data4.Recordset.Update
MsgBox "Kayit Tamamlandi..!"
temizle
Combo1.SetFocus
End Sub
```

```
Private Sub temizle()
Combo1.Clear
Combo2.Clear
Text1.Text = ""
MaskedTextBox4.Text = ""
combol_doldur
Combo1_LostFocus
End Sub
```

```
Private Sub combol_doldur()
On Error GoTo hata
SQL = "select * from bolum order by bolum"
Datal.RecordSource = SQL
Datal.Refresh
Datal.Recordset.MoveLast
say = Datal.Recordset.RecordCount
Datal.Recordset.MoveFirst
For i = 1 To say
Combo1.AddItem Datal.Recordset.bolum
Datal.Recordset.MoveNext
Next i
hata:
Select Case Err
Case Is = 3021
End Select
End Sub
```

Microsoft Visual Basic 6.0

```
Private Sub Combol_KeyPress(KeyAscii As Integer)
```

```
If KeyAscii = 13 And Combol.Text <> "" Then
```

```
If Text5.Text = 0 Then Combo2.Clear
```

```
Combo2.SetFocus
```

```
End If
```

```
End Sub
```

```
Private Sub Combol_LostFocus()
```

```
On Error GoTo hata
```

```
If Combol.Text = "" Or Combol.Text = " " Then
```

```
Combol.SetFocus
```

```
Exit Sub
```

```
End If
```

```
SQL = "select distinct firma from parca where bolum='" &  
Combol.Text & "'"
```

```
Data2.RecordSource = SQL
```

```
Data2.Refresh
```

```
Data2.Recordset.MoveLast
```

```
a = Data2.Recordset.RecordCount
```

```
If Text5.Text = 0 Then Combo2.Clear
```

```
Data2.Recordset.MoveFirst
```

```
For i = 0 To a
```

```
Combo2.AddItem Data2.Recordset.firma
```

```
Data2.Recordset.MoveNext
```

```
Next i
```

```
hata:
```

```
Select Case Err
```

```
Case Is = 6148
```

```
End Select
```

```
End Sub
```

```
Private Sub Combo2_KeyPress(KeyAscii As Integer)
```

```
If KeyAscii = 13 And Combo2.Text <> "" Then
```

```
If Text5.Text = 0 Then Text1.Text = ""
```

```
Text1.SetFocus
```

```
End If
```

```
End Sub
```

```
Private Sub Form_Activate()
```

```
Option1.Value = True
```

```
kur_sapta
```

```
combol_doldur
```

```
If Form2.Text5.Text = 1 Then
```

```
Combol.Text = Text6.Text
```

```
Combo2.Text = Text7.Text
```

```
Text1.Text = Text8.Text
```

```
SQL = "select * from parca where bolum='" & Text6.Text & "' and  
firma='" & Text7.Text & "' and model='" & Text8.Text & "'"
```

```

Data6.RecordSource = SQL
Data6.Refresh
Data6.Recordset.MoveLast
If Data6.Recordset.kur = 1 Then
Option1.Value = True
Else
Option2.Value = True
End If
MaskedTextBox4.Text = Data6.Recordset.doviz1
End If
Combo1.SetFocus
End Sub

Private Sub MaskedTextBox4_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Text5.Text = 0 Then kayt
If KeyAscii = 13 And Text5.Text = 1 Then kaytdeg
End Sub

Private Sub Option1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then MaskedTextBox4.SetFocus
End Sub

Private Sub Option2_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then MaskedTextBox4.SetFocus
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Text1.Text <> "" Then Option1.SetFocus
End Sub

Private Sub Text1_LostFocus()
On Error GoTo hata
If Text5.Text = 0 Then
SQL = "select * from parca where bolum='" & Combo1.Text & "' and
firma='" & Combo2.Text & "' and model='" & Text1.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
MsgBox "Lütfen girilen bilgilere dikkat edini. Bu kayıt var.!"
Combo1.SetFocus
Exit Sub
End If
hata:
Select Case Err
Case Is = 3021
Option1.SetFocus
End Select
End Sub

```

FORM 3



```
Private Sub tarih_kontrol()
On Error GoTo hata
SQL = "select * from kur where tarih='" & Date & "'"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
Text1.Text = 1
MaskedTextBox1.Text = Data1.Recordset.dolar
MaskedTextBox2.Text = Data1.Recordset.mark
hata:

Select Case Err
Case Is = 3021
Text1.Text = 0
End Select
End Sub

Private Sub CommandButton1_Click()
If MaskedTextBox1.Text = "" Or MaskedTextBox2.Text = "" Then
Unload Form3

Else
    If Text1.Text = 0 Then
        Data1.Recordset.AddNew
        Data1.Recordset.dolar = MaskedTextBox1.Text
        Data1.Recordset.mark = MaskedTextBox2.Text
        Data1.Recordset.tarih = Date
        Data1.Recordset.Update
        MDIForm1.Show
    Else
        SQL = "select * from kur where tarih='" & Date & "'"
        Data1.RecordSource = SQL
        Data1.Refresh
        Data1.Recordset.MoveLast
        Data1.Recordset.Edit
        Data1.Recordset.dolar = MaskedTextBox1.Text
        Data1.Recordset.mark = MaskedTextBox2.Text
```



```
Datal.Recordset.Update
MDIForm1.Show
End If
Unload Form3
End If
End Sub
```

Private Sub Form_Activate()

```
If Date = "28.02.2000" Or Date = "29.02.2000" Or Date =
"01.03.2000" Or Date = "02.03.2000" Then

MsgBox "Programin kullanim süresi dolmustur.Geçmiş olsun.", 32 +
256, "Bitti"
End
End If
tarih_kontrol
MaskedTextBox1.SetFocus
End Sub
```

Private Sub MaskedTextBox1_KeyPress(KeyAscii As Integer)

```
If KeyAscii = 13 Then MaskedTextBox2.SetFocus
End Sub
```

Private Sub MaskedTextBox2_KeyPress(KeyAscii As Integer)

```
If KeyAscii = 13 Then CommandButton1.SetFocus
End Sub
```

Bilgisayar Toplama

Parça Türü Üretici Firma Model

Fiyatı

Dolar karşılığı TL Ekle

✓ SAT

Toplam

	Parça Türü	Üretici Firma	Model	Dolar Karşılığı	TL Karşılığı
*					

```
Private Sub yazdir()  
On Error GoTo hata  
SQL = "select * from aktarma"  
Data4.RecordSource = SQL  
Data4.Refresh  
Data4.Recordset.MoveLast  
qq = Data4.Recordset.RecordCount  
DoEvents  
Printer.PaperSize = 9  
Printer.Font = "Courier New Tur"  
Printer.FontSize = 8  
GoTo K  
m:  
Printer.NewPage  
K:  
Printer.Print "  
"; Date  
Printer.FontSize = 10  
Printer.FontBold = True  
If Text3.Text = 0 Then Printer.Print "  
PARÇA SATIS BILGILERI"  
If Text3.Text = 1 Then Printer.Print "  
PC SATIS BILGILERI"  
Printer.Print
```

```

Printer.Print "                Alicinin Adi Soyadi veya Unvani : ";
Text1.Text
Printer.Print
Printer.FontSize = 8
Printer.Print "                NO    BÖLÜM                FIRMA
MODEL                FIYATI"
Printer.Print "                -----
-----"

j = 2.5
Data4.Recordset.MoveFirst
For i = 1 To qq
h = h + 1
Printer.CurrentX = 2.6 * 567
Printer.CurrentY = j * 567
Printer.Print h
Printer.CurrentX = 3.4 * 567
Printer.CurrentY = j * 567
Printer.Print Data4.Recordset("tur")
Printer.CurrentX = 7.6 * 567
Printer.CurrentY = j * 567
Printer.Print Data4.Recordset("firma")
Printer.CurrentX = 11.5 * 567
Printer.CurrentY = j * 567
Printer.Print Data4.Recordset("model")
Printer.CurrentX = 15.1 * 567
Printer.CurrentY = j * 567
Printer.Print Data4.Recordset("tl")
Data4.Recordset.MoveNext
j = j + 0.4
If j >= 27 Then
GoTo m
End If
Next i
Printer.FontBold = True
Printer.Print ""
Printer.Print "
Toplam Tutar.....: "; MaskedTextBox5.FormattedText
Printer.FontBold = False
Printer.EndDoc

hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub aktarma_bosalt()
On Error GoTo hata

```

Microsoft Visual Basic 6.0

```
SQL = "select * from aktarma"
Data4.RecordSource = SQL
Data4.Refresh
Data4.Recordset.MoveLast
sayi = Data4.Recordset.RecordCount
Data4.Recordset.MoveFirst
For i = 1 To sayi
Data4.Recordset.Delete
Data4.Recordset.MoveNext
Next i
Comb1.SetFocus

hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub kur_sapta()
SQL = "select * from kur where tarih='" & Date & "'"
Data3.RecordSource = SQL
Data3.Refresh
Data3.Recordset.MoveLast
MaskedTextBox3.Text = Data3.Recordset.dolar
MaskedTextBox4.Text = Data3.Recordset.mark
End Sub

Private Sub comb1_doldur()
On Error GoTo hata
SQL = "select * from bolum order by bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
say = Data1.Recordset.RecordCount
Data1.Recordset.MoveFirst
For i = 1 To say
Comb1.AddItem Data1.Recordset.bolum
Data1.Recordset.MoveNext
Next i

hata:
Select Case Err
Case Is = 3021
End Select
End Sub
Private Sub Combo21_Change()
End Sub

Private Sub Comb1_KeyPress(KeyAscii As Integer)
```

```
If KeyAscii = 13 And Combo1.Text <> "" Then
Combo2.Clear
Combo2.SetFocus
End If
End Sub
```

```
Private Sub Combo1_LostFocus()
On Error GoTo hata
SQL = "select distinct firma from parca where bolum='" &
Combo1.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
a = Data2.Recordset.RecordCount
Combo2.Clear
Data2.Recordset.MoveFirst
For i = 0 To a
Combo2.AddItem Data2.Recordset.firma
Data2.Recordset.MoveNext
Next i

hata:
Select Case Err
Case Is = 3021
End Select
End Sub
```

```
Private Sub Combo2_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Combo2.Text <> "" Then
Combo3.Clear
Combo3.SetFocus
End If
End Sub
```

```
Private Sub Combo2_LostFocus()
On Error GoTo hata
SQL = "select model from parca where bolum='" & Combo1.Text & "'"
and firma='" & Combo2.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
a = Data2.Recordset.RecordCount
Combo3.Clear
Data2.Recordset.MoveFirst
For i = 0 To a
Combo3.AddItem Data2.Recordset.model
Data2.Recordset.MoveNext
Next i
```

Microsoft Visual Basic 6.0

```
hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub Combo3_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Combo3.Text <> "" Then
CommandButton1.Enabled = True
CommandButton1.SetFocus
End If
End Sub

Private Sub Combo3_LostFocus()
On Error GoTo hata

SQL = "select kur,doviz1 from parca where bolum='" & Combo1.Text &
"' and firma='" & Combo2.Text & "'" and model='" & Combo3.Text &
""
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
Text2.Text = Data2.Recordset.kur
MaskedTextBox1.Text = Data2.Recordset.doviz1
If Text2.Text = 1 Then
MaskedTextBox2.Text = MaskedTextBox1.Text * MaskedTextBox3.Text
Label1.Caption = "Dolar karsiligi"
Else
MaskedTextBox2.Text = MaskedTextBox1.Text * MaskedTextBox4.Text
Label1.Caption = "Euro karsiligi"
End If

hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub CommandButton1_Click()
Data4.Recordset.AddNew
Data4.Recordset.tur = Combo1.Text
Data4.Recordset.firma = Combo2.Text
Data4.Recordset.model = Combo3.Text
Data4.Recordset.doviz = MaskedTextBox1.FormattedText
Data4.Recordset.t1 = MaskedTextBox2.FormattedText
Data4.Recordset.Update
Data4.Refresh
Combo2.Clear
Combo3.Clear
```

```

MaskedTextBox1.Text = ""
MaskedTextBox5.Text = Val(MaskedTextBox5.Text) + Val(MaskedTextBox2.Text)
MaskedTextBox2.Text = ""
CommandButton2.Enabled = True
Combo1.SetFocus
End Sub

Private Sub CommandButton2_Click()
On Error GoTo hata
SQL = "select * from aktarma"
Data4.RecordSource = SQL
Data4.Refresh
Data4.Recordset.MoveLast
If Data4.Recordset.RecordCount > 15 Then
MsgBox "Fazla sayıda parça satıldı...! Lütfen düzeltiniz..."
Exit Sub
End If

sor = MsgBox("Satis yapmak istediginize emin misiniz?", 4 + 32 +
256, "Onay")
If sor = 6 Then
ad = InputBox("Alicini Adi Soyadi veya Ünvanı", "Alici")
Text1.Text = ad
yazdir
    If Text3.Text = 1 Then
        Data5.Recordset.AddNew
        Data5.Recordset.tarih = Date
        Data5.Recordset.alici = ad
        Data5.Recordset.tl_toplam = MaskedTextBox5.FormattedText
        K = 2
        For i = 0 To 14
            DBGrid1.Row = i
            For j = 0 To 4
                K = K + 1
                DBGrid1.Col = j
                Data5.Recordset(K) = DBGrid1()
            Next j
        Next i
        Data5.Recordset.Update
        aktarma_bosalt
    Else
        Data6.Recordset.AddNew
        Data6.Recordset.tarih = Date
        Data6.Recordset.alici = ad
        Data6.Recordset.tl_toplam = MaskedTextBox5.FormattedText
        K = 2
        For i = 0 To 14
            DBGrid1.Row = i
            For j = 0 To 4

```

Microsoft Visual Basic 6.0

```
        K = K + 1
        DBGrid1.Col = j
        Data6.Recordset(K) = DBGrid1()
    Next j
Next i
Data6.Recordset.Update
aktarma_bosalt
End If
End If
```

```
hata:
Select Case Err
Case Is = 6148
If Text3.Text = 1 Then
Data5.Recordset.Update
aktarma_bosalt
Else
Data6.Recordset.Update
aktarma_bosalt
End If
MaskedTextBox5.Text = ""
End Select
End Sub
```

```
Private Sub DBGrid1_DblClick()
xads = Data4.Recordset("tur")
tlfiyat = Data4.Recordset("tl")
cevap = MsgBox("Bu kaydi silmek istediginize eminmisiniz?", 4 +
32, "Dikkat")
If cevap = 6 Then
Data4.Recordset.Delete
MaskedTextBox5.Text = MaskedTextBox5.Text - tlfiyat
Exit Sub
End If
End Sub
```

```
Private Sub Form_Activate()
kur_sapta
combol_doldur
Combol.SetFocus
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
aktarma_bosalt
End Sub
```

FORM 5



```

Private Sub combol_doldur()
On Error GoTo hata
SQL = "select * from bolum order by bolum"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
say = Data1.Recordset.RecordCount
Data1.Recordset.MoveFirst
For i = 1 To say
Combo1.AddItem Data1.Recordset.bolum
Data1.Recordset.MoveNext
Next i
hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub Combo1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Combo1.Text <> "" Then
Combo2.Clear
Combo2.SetFocus
End If
End Sub

Private Sub Combo1_LostFocus()
On Error GoTo hata
SQL = "select distinct firma from parca where bolum='" &
Combo1.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
a = Data2.Recordset.RecordCount
Combo2.Clear
Data2.Recordset.MoveFirst
For i = 0 To a
Combo2.AddItem Data2.Recordset.firma
Data2.Recordset.MoveNext

```

Microsoft Visual Basic 6.0

```
Next i
hata:
Select Case Err
Case Is = 3021
End Select
End Sub
Private Sub Combo2_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Combo2.Text <> "" Then
Combo3.Clear
Combo3.SetFocus
End If
End Sub

Private Sub Combo2_LostFocus()
On Error GoTo hata

SQL = "select model from parca where bolum='" & Combo1.Text & "'
and firma='" & Combo2.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
a = Data2.Recordset.RecordCount
Combo3.Clear
Data2.Recordset.MoveFirst
For i = 0 To a
Combo3.AddItem Data2.Recordset.model
Data2.Recordset.MoveNext
Next i

hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub Combo3_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And Combo3.Text <> "" Then
If Text1.Text = 1 Then
CommandButton1.Enabled = True
CommandButton1.SetFocus
End If

If Text1.Text = 2 Then
CommandButton2.Enabled = True
CommandButton2.SetFocus
End If
End If
End Sub
```

```
Private Sub CommandButton1_Click()
Form2.Text6.Text = Form5.Combo1.Text
Form2.Text7.Text = Form5.Combo2.Text
Form2.Text8.Text = Form5.Combo3.Text
Form2.Text5.Text = 1
Unload Form5
End Sub
```

```
Private Sub CommandButton2_Click()
On Error GoTo hata
SQL = "select * from parca where bolum='" & Combo1.Text & "'" and
firma='" & Combo2.Text & "'" and model='" & Combo3.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
Data2.Recordset.Delete
MsgBox (Combo1.Text & " " & Combo2.Text & " " & Combo3.Text & "
silindi.!")
Unload Me
hata:
Select Case Err
Case Is = 3021
End Select
End Sub
```

```
Private Sub Form_Activate()
If Text1.Text = 1 Then
Form5.Caption = "Degisiklik"
CommandButton1.Visible = True
CommandButton2.Visible = False
End If
If Text1.Text = 2 Then
Form5.Caption = "Silme"
CommandButton1.Visible = False
CommandButton2.Visible = True
End If
combol_doldur
Combo1.SetFocus
End Sub
```

FORM 6

Microsoft Visual Basic 6.0



```
Private Sub combol_doldur()  
On Error GoTo hata  
SQL = "select * from bolum order by bolum"  
Data1.RecordSource = SQL  
Data1.Refresh  
Data1.Recordset.MoveLast  
say = Data1.Recordset.RecordCount  
Data1.Recordset.MoveFirst  
For i = 1 To say  
Combol.AddItem Data1.Recordset.bolum  
Data1.Recordset.MoveNext  
Next i  
hata:  
Select Case Err  
Case Is = 3021  
End Select  
End Sub  
  
Private Sub CommandButton1_Click()  
If Combol.Text = "" Then  
Combol.SetFocus  
Exit Sub  
End If  
SQL = "select * from bolum where bolum='" & Combol.Text & "'">
Data1.RecordSource = SQL  
Data1.Refresh  
Data1.Recordset.MoveLast  
Data1.Recordset.Delete  
MsgBox "Bölüm adı silindi.!"  
Unload Form6  
End Sub  
  
Private Sub CommandButton2_Click()  
Unload Me  
End Sub  
  
Private Sub Form_Activate()  
combol_doldur  
Combol.SetFocus  
864
```

End Sub

FORM 7

Satış Raporu

Rapor Tarihi: #### ##

Data1 Data2 Data3

PC SATIŞ RAPORU		PARÇA SATIŞ RAPORU	
Alicının Adı Soyadı veya Ünvanı	Toplam Tutar	Alicının Adı Soyadı veya Ünvanı	Toplam Tutar
*		*	

Alicının Adı Soyadı veya Ünvanı: Toplam Tutar:

Parça Türü	Üretici Firma	Model	Dolar Karşılığı	TL Karşılığı
*				

YAZDIR

SİL

```
Private Sub dbgasaal()
SQL = "select * from aktarma"
Data3.RecordSource = SQL
Data3.Refresh
Data3.Recordset.MoveFirst
End Sub
```

```
Private Sub aktarma_bosalt()
On Error GoTo hata
SQL = "select * from aktarma"
Data3.RecordSource = SQL
```

Microsoft Visual Basic 6.0

```
Data3.Refresh
Data3.Recordset.MoveLast
sayi = Data3.Recordset.RecordCount
Data3.Recordset.MoveFirst
For i = 1 To sayi
Data3.Recordset.Delete
Data3.Recordset.MoveNext
Next i
Text2.Text = ""
MaskedTextBox2.Text = ""
Data3.Refresh
Data3.Recordset.MoveFirst
hata:
Select Case Err
Case Is = 3021
End Select
End Sub
```

Private Sub yazdir()

```
On Error GoTo hata

SQL = "select * from aktarma"
Data3.RecordSource = SQL
Data3.Refresh
Data3.Recordset.MoveLast
qq = Data3.Recordset.RecordCount
DoEvents
Printer.PaperSize = 9
Printer.Font = "Courier New Tur"
Printer.FontSize = 8
GoTo K
m:
Printer.NewPage
K:
Printer.Print "
"; Date
Printer.FontSize = 10
Printer.FontBold = True
If Text3.Text = 0 Then Printer.Print "
PARÇA SATIS BILGILERI"
If Text3.Text = 1 Then Printer.Print "
PC SATIS BILGILERI"
Printer.Print
Printer.Print "                Alicinin Adi Soyadi veya Unvani : ";
Text2.Text
Printer.Print
Printer.FontSize = 8
Printer.Print "                NO    BÖLÜM                FIRMA
MODEL                FIYATI"
```

```

Printer.Print " -----
-----"
j = 2.5
Data3.Recordset.MoveFirst
For i = 1 To qq
h = h + 1
Printer.CurrentX = 2.6 * 567
Printer.CurrentY = j * 567
Printer.Print h
Printer.CurrentX = 3.4 * 567
Printer.CurrentY = j * 567
Printer.Print Data3.Recordset("tur")
Printer.CurrentX = 7.6 * 567
Printer.CurrentY = j * 567
Printer.Print Data3.Recordset("firma")
Printer.CurrentX = 11.5 * 567
Printer.CurrentY = j * 567
Printer.Print Data3.Recordset("model")
Printer.CurrentX = 15.1 * 567
Printer.CurrentY = j * 567
Printer.Print Data3.Recordset("tl")
Data3.Recordset.MoveNext
j = j + 0.4
If j >= 27 Then
GoTo m
End If
Next i
Printer.FontBold = True
Printer.Print ""
Printer.Print "
Toplam Tutar.....: "; MaskedTextBox2.FormattedText
Printer.FontBold = False
Printer.EndDoc
hata:
Select Case Err
Case Is = 3021
End Select
End Sub

Private Sub aktarma_doldurl()
On Error GoTo hata
SQL = "select * from parca where alici='" & Text2.Text & "' and
tl_toplam='" & MaskedTextBox2.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveLast
For i = 3 To 73 Step 5
Data3.Recordset.AddNew
Data3.Recordset.tur = Data2.Recordset(i)

```

Microsoft Visual Basic 6.0

```
Data3.Recordset.firma = Data2.Recordset(i + 1)
Data3.Recordset.doviz = Data2.Recordset(i + 3)
Data3.Recordset.model = Data2.Recordset(i + 2)
Data3.Recordset.tl = Data2.Recordset(i + 4)
Data3.Recordset.Update
Data3.Refresh
Next i
hata:
Select Case Err
Case Is = 3021
Data3.Refresh
End Select
End Sub
```

Private Sub aktarma_doldur()

```
On Error GoTo hata
```

```
SQL = "select * from satis where alici='" & Text2.Text & "' and  
tl_toplam='" & MaskedTextBox2.Text & "'"
Data1.RecordSource = SQL
Data1.Refresh
Data1.Recordset.MoveLast
For i = 3 To 78 Step 5
Data3.Recordset.AddNew
Data3.Recordset.tur = Data1.Recordset(i)
Data3.Recordset.firma = Data1.Recordset(i + 1)
Data3.Recordset.doviz = Data1.Recordset(i + 3)
Data3.Recordset.model = Data1.Recordset(i + 2)
Data3.Recordset.tl = Data1.Recordset(i + 4)
Data3.Recordset.Update
Data3.Refresh
Next i
hata:
Select Case Err
Case Is = 3021
End Select
End Sub
```

Private Sub doldur1()

```
On Error GoTo hata
```

```
SQL = "select * from parca where tarih='" & MaskedTextBox1.Text & "'"
Data2.RecordSource = SQL
Data2.Refresh
Data2.Recordset.MoveFirst
hata:
Select Case Err
Case Is = 3021
Text1.Text = Text1.Text + 1
End Select
```


End Sub

Private Sub doldur()

On Error GoTo hata

SQL = "select * from satis where tarih='" & MaskedTextBox1.Text & "'"

Data1.RecordSource = SQL

Data1.Refresh

Data1.Recordset.MoveFirst

hata:

Select Case Err

Case Is = 3021

Text1.Text = 1

End Select

End Sub

Private Sub CommandButton1_Click()

yazdir

aktarma_bosalt

Form7.Height = 3285

Form7.Width = 10500

End Sub

Private Sub CommandButton2_Click()

sor = MsgBox("Bu kaydi silmek istediginize emin misiniz?", 4 + 32 + 256, "Silme")

If sor = 6 Then

If Text3.Text = 1 Then

SQL = "select * from satis where alici='" & Text2.Text & "'" and
tl_toplam='" & MaskedTextBox2.Text & "'" and tarih='" & MaskedTextBox1.Text
& "'"

Data1.RecordSource = SQL

Data1.Refresh

Data1.Recordset.MoveLast

Data1.Recordset.Delete

doldur

MsgBox "Kayit Silindi.!"

Else

SQL = "select * from parca where alici='" & Text2.Text & "'" and
tl_toplam='" & MaskedTextBox2.Text & "'" and tarih='" & MaskedTextBox1.Text
& "'"

Data2.RecordSource = SQL

Data2.Refresh

Data2.Recordset.MoveLast

Data2.Recordset.Delete

doldur1

MsgBox "Kayit Silindi.!"

End If

aktarma_bosalt

Microsoft Visual Basic 6.0

```
Form7.Height = 3285
Form7.Width = 10500
End If
End Sub
```

```
Private Sub CommandButton4_Click()
SQL = "select * from aktarma"
Data3.RecordSource = SQL
Data3.Refresh
Data3.Recordset.MoveLast
qq = Data3.Recordset.RecordCount
Data3.Recordset.MoveFirst
For i = 1 To qq
Data3.Recordset.Delete
Data3.Recordset.MoveNext
Next i
```

End Sub

```
Private Sub DBGrid1_DblClick()
aktarma_bosalt
Text2.Text = Data1.Recordset("alici")
MaskedTextBox2.Text = Data1.Recordset("tl_toplam")
Text3.Text = 1
aktarma_doldur
doldur
dbgasaal
Form7.Height = 7350
Form7.Width = 10500
End Sub
```

```
Private Sub DBGrid2_DblClick()
aktarma_bosalt
Text2.Text = Data2.Recordset("alici")
MaskedTextBox2.Text = Data2.Recordset("tl_toplam")
Text3.Text = 0
aktarma_doldur1
doldur1
dbgasaal
Form7.Height = 7350
Form7.Width = 10500
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
aktarma_bosalt
End Sub
```

```
Private Sub MaskedTextBox1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 And MaskedTextBox1.Text <> "" Then
```

```
doldur
doldur1
If Text1.Text = 0 Or Text1.Text = 1 Then
Form7.Height = 3285
Form7.Width = 10500
MaskedTextBox1.SetFocus
Else
Form7.Height = 900
Form7.Width = 3180
Text1.Text = 0
End If
End If
End Sub
```

ADRES VE TELEFON DEFTERI PROGRAMI

FORM 1

```
Private Declare Function ShellExecute Lib "shell32.dll" Alias  
"ShellExecuteA" (ByVal hwnd As Long, ByVal lpOperation As String,  
ByVal lpFile As String, ByVal lpParameters As String, ByVal  
lpDirectory As String, ByVal nShowCmd As Long) As Long
```

```
Private Sub Command1_Click()  
RcSet.CancelUpdate  
YeniKayit = False  
Unload Me  
Set Form1 = Nothing  
End Sub  
Private Sub Command2_Click()  
verikaydet  
End Sub
```

```
Private Sub Command3_Click()  
If Not Text3(1) = "" Then a = ShellExecute(Form1.hwnd, "Open",  
"mailto:" & Text3(1), "", "", 0)  
End Sub
```

```

Private Sub Form_Activate()
Text4.SetFocus
End Sub
Private Sub Form_Load()
Image2.Picture = Image1(0).Picture
If YeniKayit Then
Label14.Caption = "Kisi Hakkindaki Kisisel Bilgiyi Bu Kisma Girin"
Else
Label14.Caption = "'" & Form1.Caption & "'" & " Hakkindaki Kisisel
Bilgiyi Bu Kisma Girin"
End If
End Sub
Private Sub Form_Unload(Cancel As Integer)
YeniKayit = False
End Sub
Sub verikaydet()
Dim X As ListItem, C As String
On Error GoTo son
Select Case Combol.ListIndex
Case 0: C = "n"
Case 1: C = "Bayan"
Case 2: C = "Erkek"
End Select

RcSet![adisoyadi] = Form1.Text4
RcSet![fax] = Form1.Text2(2)
RcSet![cins] = C
RcSet![dogumtarihi] = Form1.Text3(0)
RcSet![evtelefonu2] = Form1.Text1(2)
RcSet![emailadresi] = Form1.Text3(1)
RcSet![evtelefonul] = Form1.Text1(1)
RcSet![evadresi] = Form1.Text1(0)
RcSet![postakodu] = Form1.Text1(3)
RcSet![semt] = Form1.Text1(4)
RcSet![il] = Form1.Text1(5)
RcSet![ilce] = Form1.Text1(6)
RcSet![issemt] = Form1.Text2(4)
RcSet![Istelefonu2] = Form1.Text2(1)
RcSet![IsAdresi] = Form1.Text2(3)
RcSet![isil] = Form1.Text2(5)
RcSet![isilce] = Form1.Text2(6)
RcSet![Istelefonul] = Form1.Text2(0)
RcSet![Not] = Form1.Text3(2)
RcSet.Update

If YeniKayit = True Then
Select Case Combol.ListIndex

```

Microsoft Visual Basic 6.0

```
Case 0: Set X = Form4.ListView1.ListItems.Add(, ,
Form1.Text4.Text, "no", "no")
Case 1: Set X = Form4.ListView1.ListItems.Add(, ,
Form1.Text4.Text, "bayan", "bayan")
Case 2: Set X = Form4.ListView1.ListItems.Add(, ,
Form1.Text4.Text, "erkek", "erkek")
End Select
If Not IsNull(Text1(1)) Then X.SubItems(1) = Text1(1)
If Not IsNull(Text1(2)) Then X.SubItems(2) = Text1(2)
If Not IsNull(Text3(1)) Then X.SubItems(3) = Text3(1)
Else 'yeni kayıt diilse

Select Case Combol.ListIndex
Case 2
Form4.ListView1.ListItems(no).Icon = "erkek"
Form4.ListView1.ListItems(no).SmallIcon = "erkek"
Case 1
Form4.ListView1.ListItems(no).Icon = "bayan"
Form4.ListView1.ListItems(no).SmallIcon = "bayan"
Case 0
Form4.ListView1.ListItems(no).Icon = "no"
Form4.ListView1.ListItems(no).SmallIcon = "no"
End Select
Form4.ListView1.ListItems(no).Text = Text4
Form4.ListView1.ListItems(no).SubItems(1) = Text1(1)
Form4.ListView1.ListItems(no).SubItems(2) = Text1(2)
Form4.ListView1.ListItems(no).SubItems(3) = Text3(1)
End If
YeniKayit = False
Unload Me
Set Form1 = Nothing
Exit Sub
son:
If Err = 3315 Then
MsgBox (" 'Adi Soyadi' Alani Bos Birakilamaz"), 16, "Hata"
Text4.SetFocus
End If
If Err = 3022 Then
MsgBox ("Ayni Ada ve Soyada Sahip Baska bir Kayit Daha Var lütfen
Bu kisimdaki bilgiyi Tekrar girin"), 16, "Uyari !"
Text4.SelStart = 0
Text4.SelLength = Len(Text4)
Text4.SetFocus
Else

Clipboard.SetText Str(Err)
MsgBox Err.Description, vbCritical, "Hata"
RcSet.CancelUpdate
End If
```

End Sub

Private Sub TabStrip1_Click()

Select Case TabStrip1.SelectedItem.Index

Case 1

Frame1.Visible = True

Frame2.Visible = False

Frame3.Visible = False

Frame4.Visible = False

Frame1.Left = 180

If Caption <> "Yeni Kayit" Then

Label14.Caption = "'" & Caption & "'" & " Hakkindaki Kisisel

Bilgileri Bu Kisma Girin"

Else

Label14.Caption = "Kisi Hakkindaki Kisisel Bilgileri Bu Kisma

Girin"

End If

Image2.Picture = Image1(0).Picture

Case 2

Frame1.Visible = False

Frame2.Visible = True

Frame3.Visible = False

Frame4.Visible = False

Frame2.Left = 180

Image2.Picture = Image1(1).Picture

If Caption <> "Yeni Kayit" Then

Label14.Caption = "'" & Caption & "'" & " Hakkindaki evle ilgili

Bilgileri Bu Kisma Girin"

Else

Label14.Caption = "Kisi Hakkindaki evle ilgili Bilgileri Bu Kisma

Girin"

End If

For i = 0 To 3

Image1(0).Visible = False

Next

Image1(1).Visible = True

Case 3

Frame1.Visible = False

Frame2.Visible = False

Frame3.Visible = True

Frame4.Visible = False

Frame3.Left = 180

Image2.Picture = Image1(2).Picture

If Caption <> "Yeni Kayit" Then

Label14.Caption = "'" & Caption & "'" & " Hakkindaki isle ilgili

Bilgileri Bu Kisma Girin"

Else

Label14.Caption = "Kisi Hakkindaki isle ilgili Bilgileri Bu Kisma

Girin"

Microsoft Visual Basic 6.0

```
End If
Case 4
Frame1.Visible = False
Frame2.Visible = False
Frame3.Visible = False
Frame4.Visible = True
Frame4.Left = 180
Image2.Picture = Image1(3).Picture
If Caption <> "Yeni Kayit" Then
Label14.Caption = "'" & Caption & "'" & " Hakkındaki diger
Bilgileri Bu Kisma Girin"
Else
Label14.Caption = "Kisi Hakkındaki diger Bilgileri Bu Kisma Girin"
End If
End Select
```

End Sub

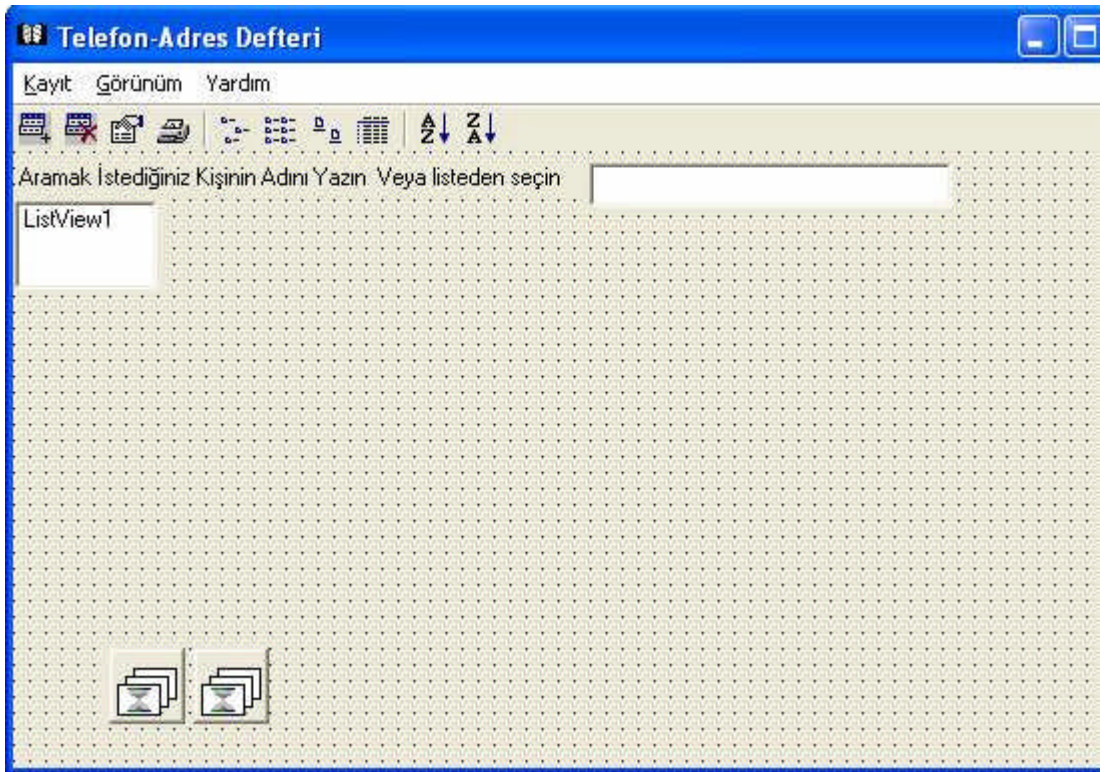
```
Private Sub Text3_GotFocus(Index As Integer)
If Index = 2 Then Command2.Default = False
End Sub
Private Sub Text3_LostFocus(Index As Integer)
If Index = 2 Then Command2.Default = True
End Sub
```

FORM 2



```
Private Sub Command1_Click()
Unload Me
Set Form2 = Nothing
End Sub
```

FORM 3



```

Sub isimyükle(isim As String, RecSet As ADODB.Recordset)

    Dim X As ListItem
    On Error GoTo Hata

    While Not RecSet.EOF
        If RecSet![cins] = "Bayan" Then Set X =
Form4.ListView1.ListItems.Add(, , RecSet![adisoyadi], "bayan",
"bayan")
        If RecSet![cins] = "Erkek" Then Set X =
Form4.ListView1.ListItems.Add(, , RecSet![adisoyadi], "erkek",
"erkek")
        If RecSet![cins] = "n" Then Set X =
Form4.ListView1.ListItems.Add(, , RecSet![adisoyadi], "no", "no")
        If Not IsNull(RecSet![evtelefonu1]) Then X.SubItems(1)
= RecSet![evtelefonu1]
        If Not IsNull(RecSet![evtelefonu2]) Then X.SubItems(2)
= RecSet![evtelefonu2]
        If Not IsNull(RcSet![emailadresini]) Then X.SubItems(3)
= RecSet![emailadresini]
        RecSet.MoveNext
    Wend

```

Microsoft Visual Basic 6.0

```
Set X = Nothing

Exit Sub

Hata:
MsgBox "Hata :" & Err.Description
End Sub

Sub kayityazdir()
Dim kri As String
Dim X As Printer

On Error GoTo son
    kri = Form4.ListView1.SelectedItem.Text

    RcSet.MoveFirst

    While Not RcSet.EOF
        If RcSet![adisoyadi] = kri Then GoTo Bitir
        RcSet.MoveNext
    Wend

Bitir:

'yazici ayarlari
Printer.ScaleMode = 6
Printer.FontName = "Times New Roman Tur"
Printer.FontSize = 14
Y = 10: x1 = 5: X2 = x1 + 50: artim = 7

'Baslik
Printer.CurrentX = 1
Printer.CurrentY = 2
Printer.Print "Adres-Telefon Belgesi" & " " &
Format(Date, "Long Date")
Printer.Line (0, 9)-(Printer.ScaleWidth, 9)

'Yazmaya Basla

'////////////////////////////////////
'//      For Each alan In RcSet.Fields          //
'//      If Not IsNull(alan.Value) Then        //
'//          Y = Y + artim                      //
'//          Printer.CurrentX = x1              //
```

```

'//          Printer.CurrentY = Y          //
'//          Printer.Print alan.Name       //
'//          Printer.CurrentX = X2        //
'//          Printer.CurrentY = Y         //
'//          Printer.Print alan.Value     //
'//          End If                       //
'//          Next                         //
'//////////
'Not!! : Kutunun içindeki kod parçası, aşağıdaki tüm satırların
yaptığın
'isin aynısını yapabilir

```

```

'adi soyadi yaz
  If Not IsNull(RcSet![adisoyadi]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Adi Soyadi:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![adisoyadi]
  End If

' ikinci adi yaz
  If Not IsNull(RcSet![fax]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "ikinci Adi:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![fax]
  End If

'dogum tarihi
  If Not IsNull(RcSet![dogumtarihi]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Dogum Tarihi:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![dogumtarihi]
  End If

'e-mail adresi yaz

```

Microsoft Visual Basic 6.0

```
If Not IsNull(RcSet![emailadres]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "E-mail Adresi:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![emailadres]
End If
```

```
'Cep tel yaz
If Not IsNull(RcSet![evtelefonu2]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Cep Telefonu:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![evtelefonu2]
End If
```

```
' Ev Telefonu Yaz
If Not IsNull(RcSet![evtelefonu1]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Ev Telefonu:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![evtelefonu1]
End If
```

```
'Ev adresi Yaz
If Not IsNull(RcSet![evadres]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Ev Adresi:"
    Printer.CurrentX = X2
    Printer.CurrentY = Y
    Printer.Print RcSet![evadres]
End If
```

```
'ev posta kodu yaz
If Not IsNull(RcSet![postakodu]) Then
    Y = Y + artim
    Printer.CurrentX = x1
    Printer.CurrentY = Y
    Printer.Print "Posta Kodu(Ev):"
```

```

        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![postakodu]
    End If

    ' ev semtini yaz
    If Not IsNull(RcSet![semt]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Semt(Ev):"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![semt]
    End If

    'ev ilini yaz
    If Not IsNull(RcSet![Il]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "il(Ev):"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![Il]
    End If

    'ev ilçe
    If Not IsNull(RcSet![Ilce]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Ilçe(Ev):"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![Ilce]
    End If

    'is yeri yaz
    If Not IsNull(RcSet![issem]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Is Yeri:"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![issem]
    End If

    ' Bölümü
    If Not IsNull(RcSet![Istelefonu2]) Then

```

Microsoft Visual Basic 6.0

```
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Bölümü:"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![Istelefonu2]
    End If

' is telefonu
    If Not IsNull(RcSet![Istelefonu1]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Is Telefonu:"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![Istelefonu1]
    End If

'isadresi yaz
    If Not IsNull(RcSet![IsAdresi]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Is Adresi:"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![IsAdresi]
    End If

'is ili
    If Not IsNull(RcSet![isil]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Il(is):"
        Printer.CurrentX = X2
        Printer.CurrentY = Y
        Printer.Print RcSet![isil]
    End If

'isilçe
    If Not IsNull(RcSet![isilce]) Then
        Y = Y + artim
        Printer.CurrentX = x1
        Printer.CurrentY = Y
        Printer.Print "Ilçe(is)"
        Printer.CurrentX = X2
```

```

        Printer.CurrentY = Y
        Printer.Print RcSet![isilce]
    End If

Printer.Line (0, 140)-(Printer.ScaleWidth, 140)
Printer.EndDoc
Exit Sub
son:
MsgBox "Hata :" & Err.Description, 16
End Sub
Sub veriyükle() ' Listedten seçilen kaydın diğer bilgilerini
gösteren alt program
    Dim kri As String
    On Error GoTo son

    kri = ListView1.SelectedItem.Text 'Kayıt Arama Kriteri: Adı
    Soyadı

    RcSet.MoveFirst

    While Not RcSet.EOF
        If RcSet![adisoyadı] = kri Then GoTo Bitir
        RcSet.MoveNext
    Wend

Bitir:

    Form1.Text4 = RcSet![adisoyadı] 'Ayrıntılar formunda bulunan
    kaydın adı soyadını görüntüle
    Form1.Caption = RcSet![adisoyadı] 'ayrıntılar formunun
    başlığını bulunan kaydın adı soyadı yap
    'ayrıntılar formundaki etiketin başlığını belirle
    Form1.Label14.Caption = "" & Form1.Caption & "" & "
    Hakkındaki Kisisel Bilgileri Bu Kısma Girin"
    Form1.Combol.AddItem ""
    Form1.Combol.AddItem "Bayan"
    Form1.Combol.AddItem "Erkek"

    'bulunan kaydın diğer ayrıntılarını form1(ayrıntı formu) 'de
    bulunan text'lerde görüntüle

    If Not IsNull(RcSet![cins]) Then
        Select Case RcSet![cins]
            Case "Erkek": Form1.Combol.ListIndex = 2
            Case "Bayan": Form1.Combol.ListIndex = 1
            Case "n": Form1.Combol.ListIndex = 0
        End Select
    End If
End Sub

```

Microsoft Visual Basic 6.0

```
End Select
End If

If Not IsNull(RcSet![fax]) Then Form1.Text2(2) = RcSet![fax]
If Not IsNull(RcSet![dogumtarihi]) Then Form1.Text3(0) =
RcSet![dogumtarihi]
If Not IsNull(RcSet![evtelefonu2]) Then Form1.Text1(2) =
RcSet![evtelefonu2]
If Not IsNull(RcSet![emailadres1]) Then Form1.Text3(1) =
RcSet![emailadres1]
If Not IsNull(RcSet![evtelefonu1]) Then Form1.Text1(1) =
RcSet![evtelefonu1]
If Not IsNull(RcSet![evadres1]) Then Form1.Text1(0) =
RcSet![evadres1]
If Not IsNull(RcSet![postakodu]) Then Form1.Text1(3) =
RcSet![postakodu]
If Not IsNull(RcSet![semt]) Then Form1.Text1(4) = RcSet![semt]
If Not IsNull(RcSet![il]) Then Form1.Text1(5) = RcSet![il]
If Not IsNull(RcSet![ilce]) Then Form1.Text1(6) = RcSet![ilce]
If Not IsNull(RcSet![issem]) Then Form1.Text2(4) =
RcSet![issem]
If Not IsNull(RcSet![Istelefonu2]) Then Form1.Text2(1) =
RcSet![Istelefonu2] 'bölümü
If Not IsNull(RcSet![IsAdresi]) Then Form1.Text2(3) =
RcSet![IsAdresi]
If Not IsNull(RcSet![isil]) Then Form1.Text2(5) = RcSet![isil]
If Not IsNull(RcSet![isilce]) Then Form1.Text2(6) =
RcSet![isilce]
If Not IsNull(RcSet![Istelefonu1]) Then Form1.Text2(0) =
RcSet![Istelefonu1]
If Not IsNull(RcSet![Not]) Then Form1.Text3(2) = RcSet![Not]

Form1.Show 1 'Ayrıntılar formunu göster

Exit Sub
son: 'hata isyeyici
MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub Form_Load()
Dim BagStr As String

On Error GoTo son 'hata olustugunda hata isleyicye git
If App.PrevInstance Then End ' Programin Ikinci Defa
çalışmasını önle
With Baglanti
.CursorLocation = adUseClient
```



```

        .ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=" & App.Path & "\Adresler.mdb"
        .Open
        End With
        RcSet.CursorType = adOpenDynamic
        RcSet.CursorLocation = adUseClient
        RcSet.LockType = adLockOptimistic
        RcSet.Open "genel", Baglanti
        RegYükle ' Registry verilerini al
        ÖzellikYükle 'Registry verilerine göre nesnelerin
        özelliklerini belirle
        isimYükle "Genel", RcSet 'Adres defterindeki kayitlari Listeye
        doldur
        Exit Sub
    ' Hata isleyici
son:
    MsgBox Err.Description, vbCritical, "Hata"
Resume Next
End Sub

Private Sub Form_Resize() ' formun boyutlari degistiginde
    On Error GoTo son
    'liste ve text kutusunun uzunlugu ve genisligini formun yeni
    boyutlarına göre ayarla
    ListView1.Width = Form4.Width - ListView1.Left - 100
    ListView1.Height = Form4.Height - (ListView1.Top + 680)

    Exit Sub
son:
End Sub
Private Sub Form_Unload(Cancel As Integer) ' form geri
yüklendiğinde(Programdan çıkıldığında)
    On Error GoTo Hata
    RegKaydet ' Registry'e Nesnelerin özelliklerini son haliyle
    kaydet
    RcSet.Close 'Tabloyu kapat
    Baglanti.Close 'Baglantiyi kapat

    'Degiskenleri bellekten sil
    Set RcSet = Nothing
    Set Baglanti = Nothing

    End 'programi sonlandir
Hata:
    Resume Next
End Sub
Private Sub ListView1_DblClick()

    On Error GoTo son 'hata olustugunda hata isleyiciye git

```

Microsoft Visual Basic 6.0

```
'listede seçili bir isim yoksa alt programdan çık
    If ListView1.SelectedItem.Selected = False Then Exit Sub
'seçili ismin numarasini al
    no = ListView1.SelectedItem.Index
'seçili isme göre kisinin diger ayrıntilarini form1(ayrintilar
formunda göster)
    veriyükle

    Exit Sub
son:
End Sub

Private Sub ListView1_KeyDown(KeyCode As Integer, Shift As
Integer)
    On Error GoTo son

    If ListView1.SelectedItem.Selected Then
        If KeyCode = 46 Then sil ListView1.SelectedItem.Text 'eger
'Del' Tusuna basilirsaya kaydi sil
        If KeyCode = 13 Then ListView1_DblClick 'Enter tusuna
basilirsaya kisinin ayrıntilarini göster
        End If

    Exit Sub
son:
    MsgBox ("Hata:" & Err.Description), 16, "Hata"

End Sub
Private Sub ListView1_MouseUp(Button As Integer, Shift As Integer,
X As Single, Y As Single)
    On Error GoTo son
    If Button = 2 Then 'mouse'in sag tusuna basilirsaya

        kontrol 'menü öğelerinin Durumlarini kontrol et
        PopupMenu menudosya, 2, , , menuac 'menugörünüm menüsünü
PopupMenu(patlayan menu) olarak yarat

    End If

    Exit Sub
son:
    MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menuac_Click()
    ListView1_DblClick 'listenin çift tiklamasiyla ayni görevi
yerine getiriyor
End Sub
Private Sub menuaraççubugu_Click() 'araç çubuguna tiklanirsaya
```

```

        On Error GoTo son 'hata olustugunda hata isleyiciye git
        Aracubugu = Not Aracubugu
        Toolbar1.Visible = Aracubugu
' diger nesnelerin genisliklerini araç çubugunun görünürlüğüne
göre ayarla
        If Aracubugu Then
            Text1.Top = Toolbar1.Height + 30: Label1.Top =
Toolbar1.Height + 30
            ListView1.Top = Toolbar1.Height + Text1.Height + 10
            ListView1.Height = ListView1.Height - Toolbar1.Height
        Else
            Text1.Top = 30: Label1.Top = 30
            ListView1.Top = Text1.Height + 10
            ListView1.Height = ListView1.Height + Toolbar1.Height
        End If
        Exit Sub
son:
        MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menuartan_Click()
        On Error GoTo son
        'listede siralama A da Z ye
        Siralama = lvwAscending
        ListView1.SortOrder = Siralama
        'Araç çubugundaki siralama butonlarini degistir
        Toolbar1.Buttons(11).Value = tbrPressed
        Toolbar1.Buttons(12).Value = tbrUnpressed
        Exit Sub
son:
        MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menuayrinti_Click()
        On Error GoTo son
        Görünüm = lvwReport
        ListView1.View = Görünüm
        Toolbar1.Buttons(6).Value = tbrUnpressed
        Toolbar1.Buttons(7).Value = tbrUnpressed
        Toolbar1.Buttons(8).Value = tbrUnpressed
        Toolbar1.Buttons(9).Value = tbrPressed
        Exit Sub
son:
        MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menuazalan_Click()
        On Error GoTo son
        Siralama = lvwDescending
        ListView1.SortOrder = Siralama

```

Microsoft Visual Basic 6.0

```
    Toolbar1.Buttons(11).Value = tbrUnpressed
    Toolbar1.Buttons(12).Value = tbrPressed
    Exit Sub
son:
    MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menubüyük_simgesi_Click()
On Error GoTo son
Görünüm = lvwIcon
ListView1.View = Görünüm
Toolbar1.Buttons(6).Value = tbrUnpressed
Toolbar1.Buttons(7).Value = tbrUnpressed
Toolbar1.Buttons(8).Value = tbrPressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Exit Sub
son:
MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menudosya_Click()
On Error GoTo son
menuac.Enabled = ListView1.ListItems.Count > 0
menusil.Enabled = ListView1.ListItems.Count > 0
menuyaz.Enabled = ListView1.ListItems.Count > 0
menuac.Enabled = ListView1.SelectedItem.Selected
menusil.Enabled = ListView1.SelectedItem.Selected
menuyaz.Enabled = ListView1.SelectedItem.Selected
son:
End Sub

Private Sub menuend_Click()
    On Error GoTo Hata
    RegKaydet ' Registry'e Nesnelerin özelliklerini son haliyle
kaydet
    RcSet.Close 'Tabloyu kapat
    Baglanti.Close 'Baglantiyi kapat
    Set RcSet = Nothing
    Set Baglanti = Nothing
    End
Hata:
    Resume Next
End Sub

Private Sub menugörünüm_Click()
kontrol
End Sub

Private Sub menuhakkında_Click()
```

```

Form2.Show 1
End Sub
Private Sub menuküçüksimge_Click()
On Error GoTo son
Görünüm = lvwSmallIcon
ListView1.View = Görünüm
Toolbar1.Buttons(6).Value = tbrPressed
Toolbar1.Buttons(7).Value = tbrUnpressed
Toolbar1.Buttons(8).Value = tbrUnpressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Exit Sub
son:
MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub

Private Sub menuliste_Click()
On Error GoTo son
Görünüm = lvwList
ListView1.View = Görünüm
Toolbar1.Buttons(6).Value = tbrUnpressed
Toolbar1.Buttons(7).Value = tbrPressed
Toolbar1.Buttons(8).Value = tbrUnpressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Exit Sub
son:
MsgBox ("Hata:" & Err.Description), 16, "Hata"
End Sub
Private Sub menusal_Click()
On Error GoTo son
If Not ListView1.SelectedItem.Selected Then Exit Sub
sil ListView1.SelectedItem.Text
son:
End Sub

Private Sub menyuz_Click()
Dim soru As Integer, isim As String
isim = Me.ListView1.SelectedItem.Text
soru = MsgBox("Adres Defterindeki '" & isim & "' adli kisinin tüm bilgileri yazdirilicak.Bunu yapmak istediginizden emin misiniz", vbYesNo)
If soru = vbYes Then kayityazdir
End Sub

Private Sub menyuyeni_Click()

YeniKayit = True
RcSet.AddNew
Form1.Caption = "Yeni Kayit"

```

Microsoft Visual Basic 6.0

```
Form1.Label14.Caption = "Kisi Hakkındaki Kisisel Bilgileri Bu  
Kisma Girin"  
Form1.Combol.AddItem ""  
Form1.Combol.AddItem "Bayan"  
Form1.Combol.AddItem "Erkek"  
Form1.Combol.ListIndex = 0  
Form1.Show 1
```

End Sub

Sub sil(X As String)

```
Dim kri  
On Error GoTo son  
kri = ListView1.SelectedItem.Text  
a = MsgBox(X & " Kaydi Kalici Olarak Silinecek Bunu Yapmak  
istediginizden eminmisiniz ?", vbYesNo, "Sil")  
If a = vbNo Then Exit Sub  
Baglanti.Execute "DELETE FROM genel WHERE [Adisoyadi]=' " & kri &  
" '"
```

'Not:satir silme islemi "Rcset.Delete" seklindedede yapılabilir"

```
ListView1.ListItems.Remove (ListView1.SelectedItem.Index)
```

son:

End Sub

Sub RegKaydet()

```
SaveSetting "AdresTelefon", "Ozellik", "Gorunum", Görünüm  
SaveSetting "AdresTelefon", "Ozellik", "Siralama", Siralama  
SaveSetting "AdresTelefon", "Ozellik", "Toolbar", Aracubugu  
SaveSetting "AdresTelefon", "Ozellik", "Status", durumcubugu
```

End Sub

Sub RegYükle()

On Error GoTo son

```
Görünüm = Val(GetSetting("AdresTelefon", "Ozellik", "Gorunum"))  
Siralama = Val(GetSetting("AdresTelefon", "Ozellik", "Siralama"))  
Aracubugu = Val(GetSetting("AdresTelefon", "Ozellik", "Toolbar"))  
durumcubugu = Val(GetSetting("AdresTelefon", "Ozellik", "Status"))
```

Exit Sub

son:

```
MsgBox "Hata :" & Err.Description
```

End Sub

Sub özellikeykle()

On Error GoTo son

```
Toolbar1.Visible = Aracubugu
```

```
ListView1.View = Görünüm
```

```
ListView1.SortOrder = Siralama
```

If Not Aracubugu Then

```
Text1.Top = 30: Label1.Top = 30
```

```
ListView1.Top = Text1.Height + 10
```

End If

```

Select Case Görünüm
Case lvwList
Toolbar1.Buttons(6).Value = tbrUnpressed
Toolbar1.Buttons(7).Value = tbrPressed
Toolbar1.Buttons(8).Value = tbrUnpressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Case lvwSmallIcon
Toolbar1.Buttons(6).Value = tbrPressed
Toolbar1.Buttons(7).Value = tbrUnpressed
Toolbar1.Buttons(8).Value = tbrUnpressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Case lvwIcon
Toolbar1.Buttons(6).Value = tbrUnpressed
Toolbar1.Buttons(7).Value = tbrUnpressed
Toolbar1.Buttons(8).Value = tbrPressed
Toolbar1.Buttons(9).Value = tbrUnpressed
Case lvwReport
Toolbar1.Buttons(6).Value = tbrUnpressed
Toolbar1.Buttons(7).Value = tbrUnpressed
Toolbar1.Buttons(8).Value = tbrUnpressed
Toolbar1.Buttons(9).Value = tbrPressed
End Select
Select Case Siralama
Case lvwAscending
Form4.Toolbar1.Buttons(11).Value = tbrPressed
Form4.Toolbar1.Buttons(12).Value = tbrUnpressed
Case lvwDescending
Form4.Toolbar1.Buttons(11).Value = tbrUnpressed
Form4.Toolbar1.Buttons(12).Value = tbrPressed
End Select
Toolbar1.Buttons(6).Style = tbrButtonGroup
Toolbar1.Buttons(7).Style = tbrButtonGroup
Toolbar1.Buttons(8).Style = tbrButtonGroup
Toolbar1.Buttons(9).Style = tbrButtonGroup
Toolbar1.Buttons(11).Style = tbrButtonGroup
Toolbar1.Buttons(12).Style = tbrButtonGroup
Exit Sub
son:
MsgBox "Hata :" & Err.Description
End Sub

Private Sub Text1_Change()
Dim x1 As ListItem
Dim kri As String
On Error GoTo son
kri = Text1
If Len(kri) = 0 Then Exit Sub
Set x1 = ListView1.FindItem(kri, lvwText, , lvwPartial)

```

Microsoft Visual Basic 6.0

```
If x1 Is Nothing Then Exit Sub
x1.Selected = True
x1.EnsureVisible
son:
End Sub
```

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
```

```
If KeyAscii = 13 Then
ListView1.SetFocus
KeyAscii = 0
End If
End Sub
Sub kontrol()
menudurum.Checked = durumcubugu
menuaraççubugu.Checked = Aracubugu
Select Case Görünüm
```

```
Case lvwSmallIcon
menüküçüksimge.Checked = True
menuliste.Checked = False
menubüyüksimge.Checked = False
menuayrinti.Checked = False
```

```
Case lvwList
menüküçüksimge.Checked = False
menuliste.Checked = True
menubüyüksimge.Checked = False
menuayrinti.Checked = False
```

```
Case lvwIcon
menüküçüksimge.Checked = False
menuliste.Checked = False
menubüyüksimge.Checked = True
menuayrinti.Checked = False
```

```
Case lvwReport
menüküçüksimge.Checked = False
menuliste.Checked = False
menubüyüksimge.Checked = False
menuayrinti.Checked = True
End Select
```

```
Select Case Siralama
Case lvwAscending
menuartan.Checked = True
menuazalan.Checked = False
Case lvwDescending
menuazalan.Checked = True
menuartan.Checked = False
```



```

End Select
End Sub

Private Sub Toolbar1_ButtonClick(ByVal Button As
MSComctlLib.Button)
    Dim kri As String
    On Error GoTo son
    Select Case Button.Index
        Case 1
            YeniKayit = True
            RcSet.AddNew
            Form1.Caption = "Yeni Kayit"
            Form1.Label14.Caption = "Kisi Hakkindaki Kisisel
Bilgileri Bu Kisma Girin"
            Form1.Combol.AddItem ""
            Form1.Combol.AddItem "Bayan"
            Form1.Combol.AddItem "Erkek"
            Form1.Combol.ListIndex = 0
            Form1.Show 1
        Case 2
            If ListView1.SelectedItem.Selected = False Then Exit
Sub
            sil ListView1.SelectedItem.Text
        Case 3: ListView1_DblClick
        Case 4
            If ListView1.SelectedItem.Selected = False Then Exit
Sub
            kayityazdir
        Case 6
            ListView1.View = lvwSmallIcon
            Görünüm = lvwSmallIcon
        Case 7
            ListView1.View = lvwList
            Görünüm = lvwList
        Case 8
            ListView1.View = lvwIcon
            Görünüm = lvwIcon
        Case 9
            ListView1.View = lvwReport
            Görünüm = lvwReport
        Case 11
            ListView1.SortOrder = lvwAscending
            Siralama = lvwAscending
        Case 12
            ListView1.SortOrder = lvwDescending
            Siralama = lvwDescending
    End Select
    Exit Sub
son:

```

Microsoft Visual Basic 6.0

```
MsgBox ("Hata:" & Err.Description), 16, "Hata"  
End Sub
```

MODULE 1

```
Public Görünüm As Integer  
Public Siralama As Integer  
Public Aracubugu As Integer  
Public durumcubugu As Integer  
Public YeniKayit As Boolean  
Public KeyNo As Integer  
Public no As Integer  
Public Baglanti As New ADODB.Connection  
Public RcSet As New ADODB.Recordset
```