T.C.
PAMUKKALE ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
MAKİNA MÜHENDİSLİĞİ BÖLÜMÜ

VISUAL BASIC 6.0 UYGULAMALARI

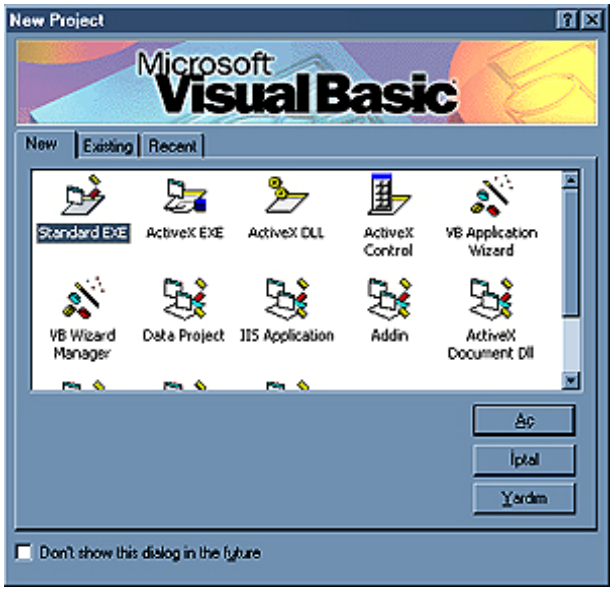
Kamil Gökhan ÜNLÜER

Yrd. Doç. Dr. Şahin ÇONKUR

DENİZLİ
2002

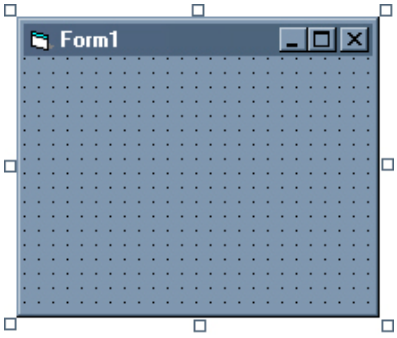
UYGULAMA 1

Klavyenin tuşlarının her birinin programlama ortamında birer kodları vardır. Bu kodlar vasıtasıyla her tuşa görevler verilebilir , işlemler yaptırılabilir. Hazırlayacağımız program hangi tuşun hangi koda sahip olduğunu belirlemek amacındadır.

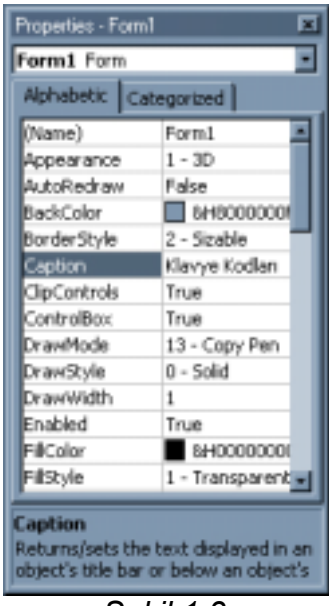


Şekil 1.1

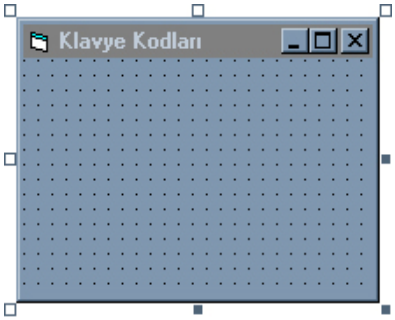
1.Visual Basic açılırken *Şekil 1.1* deki "New Project" menüsü ekrana gelir. "Standart EXE" simgesini seçip "Aç"ı tıklayın.



Şekil 1.2



Şekil 1.3

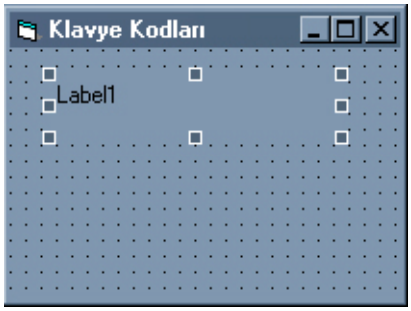


Şekil 1.4

2. Açılan yeni proje içerisinde bir form mevcuttur. *Şekil 1.2* deki formun biçimlendirilmesi, forma gerekli nesnelerin eklenmesi ve bunların yerleştirilmesi ile programın görünür yüzü olan form oluşturulmuş olacaktır. Formun kenarlarındaki noktalar yardımıyla formun boyutları ayarlanabilir.

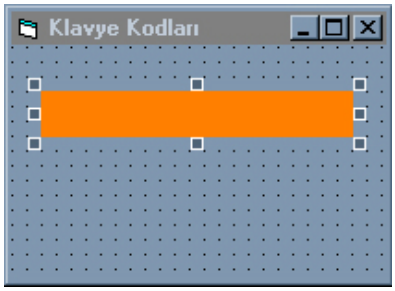
3. Formun başlığı ilk durumda “Form1” dir. Bunun değiştirilmesi için “Properties” menüsünün kullanımı gerekmektedir. Visual Basic açıldığında Properties menüsü ekranda mevcuttur. Ancak herhangi bir nedenle menünün kapatılmış olması durumunda, menüyü ekrana getirmek için  simgesine tıklanmalıdır. Form seçiliyken *Şekil 1.3* de görülen Properties penceresinden “Caption” özelliğini kullanarak “Form1” başlığını “Klavye Kodları” olarak değiştirin. Böylece form başlığının değiştiğini göreceksiniz. (*Şekil 1.4*)

Şekil 1.5



Şekil 1.6

Şekil 1.7



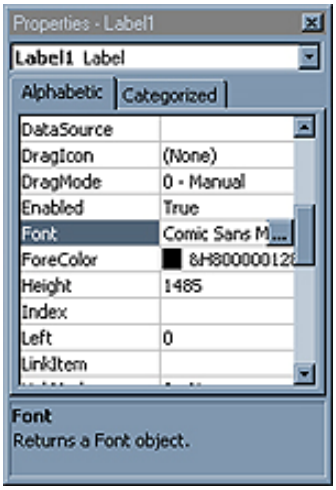
Şekil 1.8

4. Toolbar penceresinden *Şekil 1.5* üzerinde görülen “Label” seçeneğine tıklanarak form üzerine “Label1” isimli etiket nesnesi oluşturulur.(*Şekil 1.6*) Etiket yardımıyla program kullanılırken elde edilen tuş kodları kullanıcıya bildirilecektir.

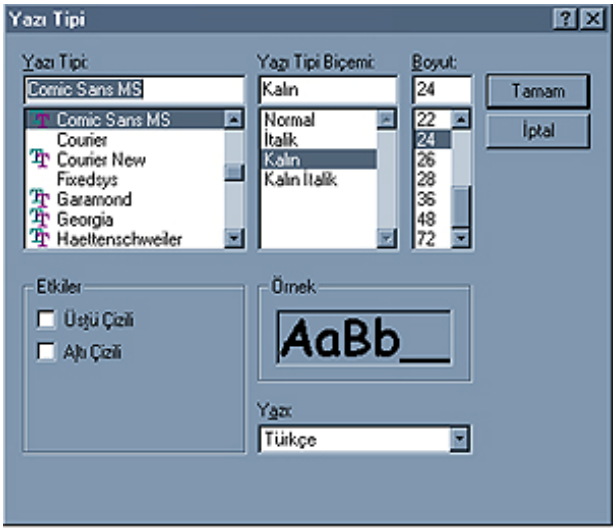
5. Etiketin zemin renginin değiştirilmesi properties menüsünden “BackColor” özelliğini kullanarak gerçekleştirilir. Bu özelliğin seçilmesi ile programcı ister Windows un system renklerini isterse de paletten herhangi bir rengi etiketin zemin rengi olarak kullanabilir.(Şekil 1.7) Renk seçimi ile etiket *şekil 1.8* deki görünümü kazanacaktır.

Şekil 1.9

6. Forma boyutlandırma noktalarını kullanılarak yatay bir görünüm kazandırın. Etiketin boyutlandırma noktaları yardımıyla, etiketi formun içine tam olarak yerleştirin. Bu işlemin amacı sadece form görünümüne estetik katmaktır.

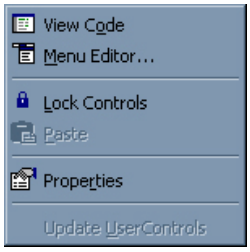


Şekil 1.10



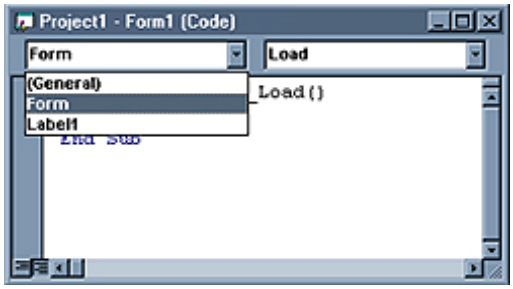
Şekil 1.11

7. Etiketin yazı tipinin,yazı tipi boyutu ve rengi properties menüsünün kullanımı ile seçilebilir. *Şekil 1.10* da görülen font özelliği seçilerek "Yazı Tipi" menüsü çalıştırılır. Bu menü yardımıyla yazı tipi özellikleri belirlenir. (*Şekil 1.11*) Yazının rengi ise "ForeColor" özelliği ile seçilir. Uygulamamızda kullandığımız yazı tipinin boyutu 24, tipi "Comic Sans MS" ve siyah renktir.

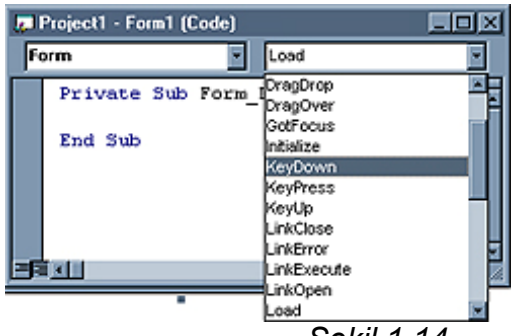


Şekil 1.12

8. Uygulamamızın form hazırlama işlemleri tamamlanmasıyla artık kod yazımına başlanmaktadır. Bunun için kod penceresi ekrana getirilmelidir. Kod penceresinin ekrana getirilmesi için bir çok yöntemden biri, projenin herhangi bir yerine sağ tıklanmasıyla *şekil 1.12* de görülen pencereden "View Code" özelliğinin seçimidir.

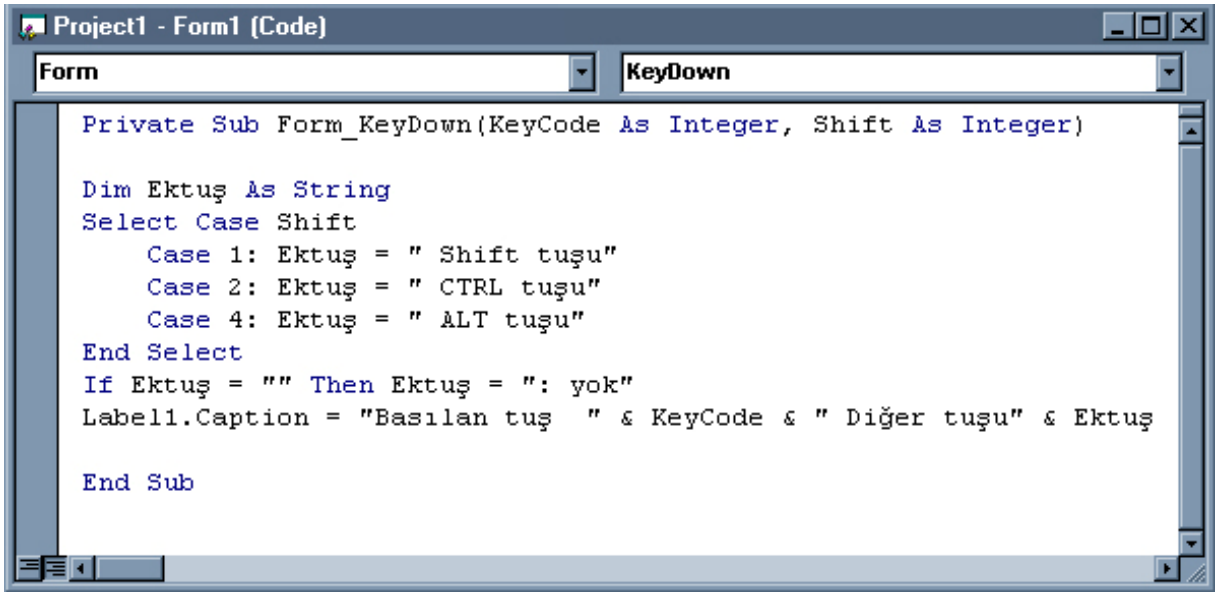


Şekil1.13



Şekil 1.14

9.Kod penceresinin sol kısmında yer alan ve nesnelerin yer aldığı listeden "Form" seçeneğini seçin.(*Şekil 1.13*) "Form" un seçilmesiyle sağdaki seçenekler penceresinden "Keydown" olayını seçin.



Şekil 1.15

10. Kod penceresinden "Keydown" özelliğinin seçilmesi ile "Private Sub Form_ ...." ve "End Sub " satırlarının otomatik olarak geldiği görülür. Bu iki satır arasına *Şekil 1.15* de görülen kodlar yazılır. Şimdi sırasıyla her satırın işlevine değinelim:

Private Sub Form_Keydown(Keycode as Integer, Shift as Integer)

Kendiliğinden gelen bu satırla "Keycode" ve "Shift" değişkenleri atanmaktadır. "Keycode" değişkeni klavyenin ektuşları haricinde herhangi bir tuşa basıldığında, basılan tuşun kod numarasını kazanır. "Shift" değişkeni ise basılan ektuş eğer CTRL ise 2, ALT ise 4, Shift ise 1 değerini kazanacaktır. Eğer herhangi bir ektuş kullanımı yok ise bu değişken herhangi bir değer kazanmayacaktır.

Dim Ektuş As String

Klavyede bulunan ve ikinci karekterlerin kullanımını sağlayan Ctrl, Alt, Shift tuşlarının programımız tarafaından belirlenmesi için kullanılacak olan "Ektuş" değişkeni "Dim" komutu ile tanımlanmaktadır.

Select Case Shift
Case 1: Ektuş = " Shift tuşu"
Case 2: Ektuş = " CTRL tuşu"
Case 4: Ektuş = " ALT tuşu"
End Select

Select Case komutu kullanımı ile "Ektuş" değişkenine atama yapılacaktır. Shift değişkeni eğer 1 ise ektuş değişkenine " Shift tuşu" kelimeleri, Shift değişkeni eğer 2 ise ektuş değişkenine " CTRL tuşu" kelimeleri, Shift değişkeni eğer 4 ise ektuş değişkenine " ALT tuşu" kelimesi atanacaktır.

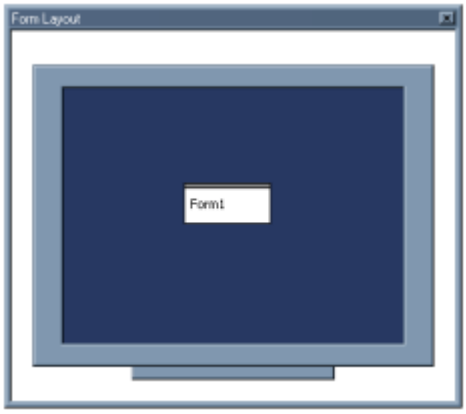
If Ektuş = "" Then Ektuş = " :yok"

Kullanıcı eğer herhangi bir ektuşu kullanmaz ise yukarıdaki select case satırlarında herhangi bir sözcüğe eşleştirilmemiş durumda olacaktır. Dolayısıyla eğer ektuş boş ise (boş durumda "" şeklindedir) bu değişkene " :yok" sözcüğü atanacaktır.

Label1.Caption = "Basılan tuş " & KeyCode & " Diğer tuşu" & Ektuş

Ektuş değişkenine sözcük ataması işleminin sona ermesiyle artık elde edilen verilerin ekrana yazdırılması gerekmektedir. Formumuzda bulunan Label1 isimli etiketimiz yardımıyla bu işlemi gerçekleştireceğiz. Etiket üzerine herhangi bir şey yazdırmak için "Caption" özelliği kullanılır. Ayrıca "&" ibaresinin kullanımının amacı sayısal işlemlerde kullanılan toplama işlemiyle aynıdır. Ancak burada sayısal sonuç yerine tek bir cümle elde edilir.

Bizim uygulamamızda etiket üzerine önce "Basılan tuş" yazılacak daha sonra bunun yanına "Keycode" değişkenin değeri yazılacaktır. Bunların yanına ise "Diğer tuşu" yazılacak ve yanına da "Ektuş" değerinin karşılığı eklenecektir.



Şekil 1.16

11. Program çalıştırıldığında formun ekrandaki başlangıç konumunu "Form Layout" penceresini kullanarak ayarlayabilirsiniz. Bunun için öncelikle  simgesine tıklayarak pencereyi ekrana getirin. "Form layout" penceresinde formun görünür olması için projede formun açık olması gerekmektedir. Fareyi pencere içerisindeki formun üzerine getirip sol tuşuna basılı tutarak, formu ekranın istediğiniz konumuna getirebilirsiniz.(*Şeki1.16*)

12. Kodların yazılımının sona ermesi ile "F5" tuşu kullanılarak program çalıştırılır.

Şekil 1.17

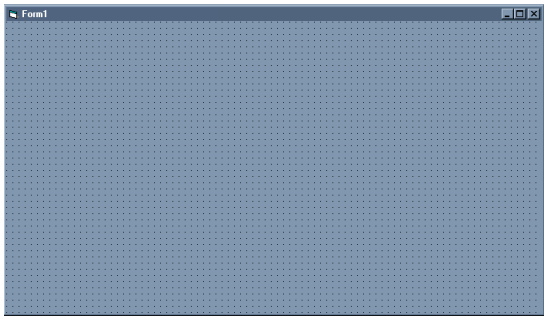
Şekil 1.18

13. Program çalıştırıldığında eğer "e" tuşuna basıldığında *Şekil 1.17* da görüldüğü gibi "e" tuşunun kodu olan "68" değeri yazılacak, kullanılan herhangi bir ektuş olmadığından "yok" yazmaktadır. Eğer "CTRL" ve "c" tuşlarına beraber basıldığında "c" tuşunun kodu olan "67" sayısı ve ektuş "CTRL" görülecektir. (*Şekil 1.18*)

UYGULAMA 2

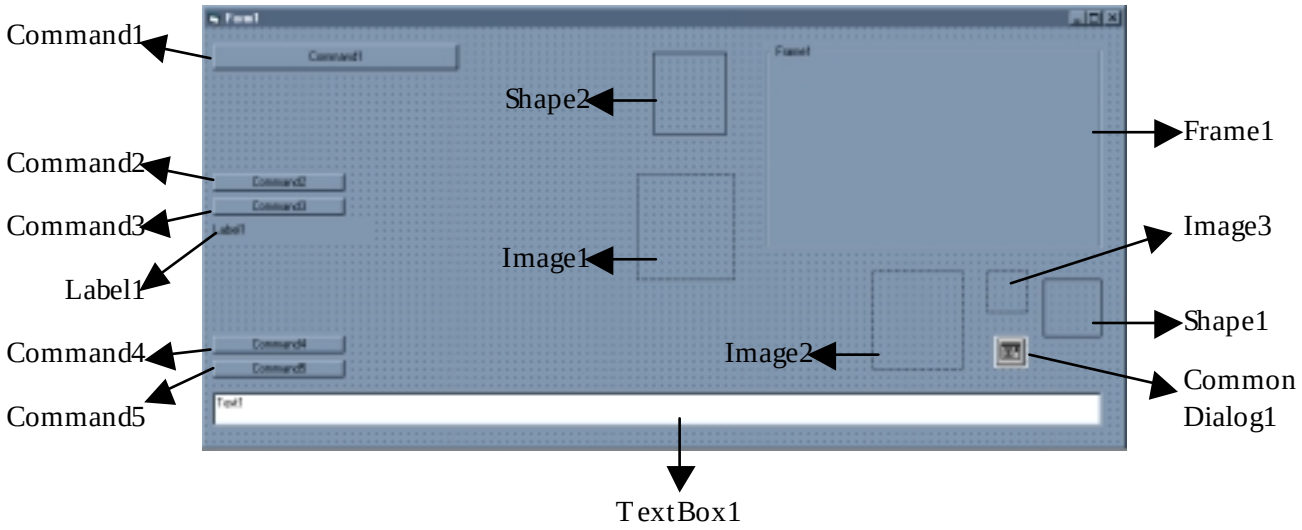
Windows ortamında kullandığınız programlarda sık sık kullandığınız iletişim kutularının ve yine windows işletim sisteminin kullanıcıya büyük kolaylıklar sağlayan

“Drag-Drop” yani “Sürükle-Bırak” özelliğinin kullanımı Visual Basic ortamında oldukça kolaydır. Şimdi gerçekleştireceğimiz uygulama, windowsun bu iki özelliğinin kullanımında giriş yapma amacındadır.



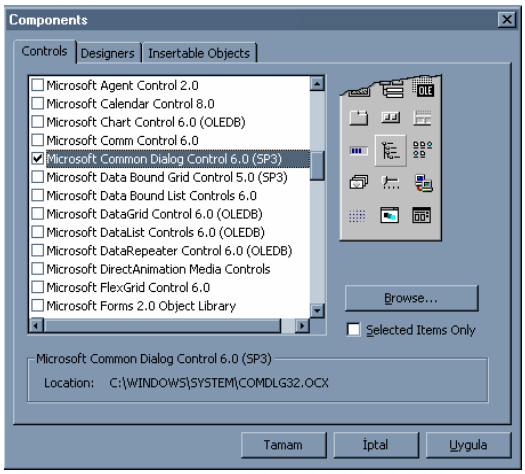
Şekil 2.1

1. Visual Basic ‘de yeni bir proje çalışmasına başlandığında , şekil 2.1 de görülen ve programımızın görünen yüzü olan “Form 1” projeye otomatik olarak eklenir. Şekil 2.1 de görülen form bir standart Microsoft formudur.



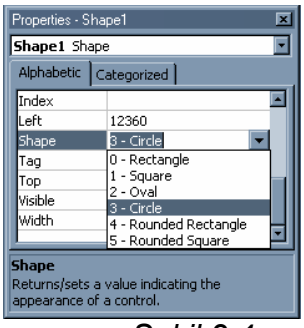
Şekil 2.2

2. Toolbar penceresini kullanarak formumuza 5 tane “Command Button”, 1 tane “Label”, 1 tane “Textbox”, 1 tane “Frame”, 3 tane “Image” ve 2 tane “Shape” nesnesi ekleyin.

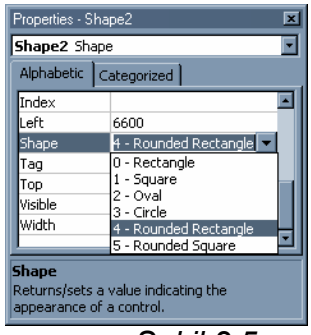


Şekil 2.3

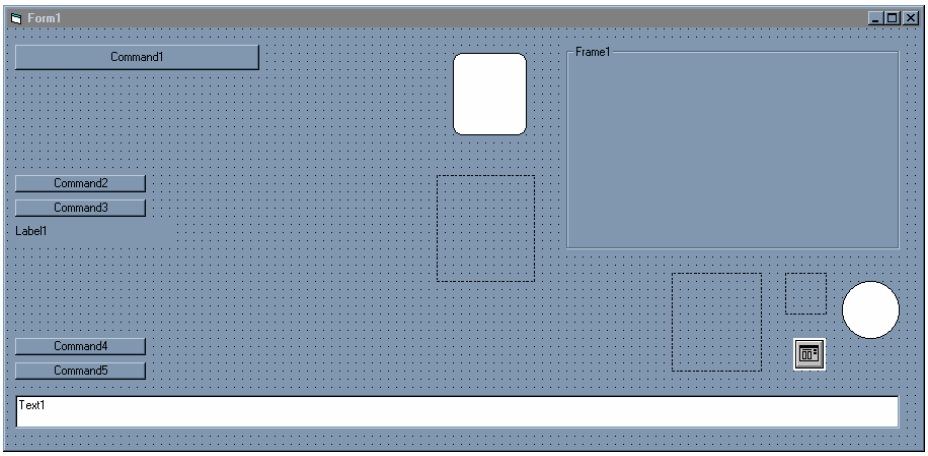
3. Ayrıca formumuza 1 tane “CommonDialog” kontrolü eklememiz gerekiyor. Ancak bu kontrol toolbar pencerenizde bulunmayabilir. “CommonDialog” kontrolünü toolbar penceresine eklemek için, öncelikle toolbar penceresine sağ kliklenmeli ve açılan pencereden “Components” işlevi seçilmelidir. Bu seçim ile şekil 2.3 de görülen pencere ekrana gelecektir. Components penceresinde bulunan ve kontrollerin isimlerinin yer aldığı listeden, ” Microsoft Common Dialog Control” seçeneğinin yanında bulunan kutu işaretlenmesi ve daha sonra “Uygula” butonuna basılması ile “CommonDialog” kontrolünün simgesi toolbar üzerinde yer alır.



Şekil 2.4



Şekil 2.5



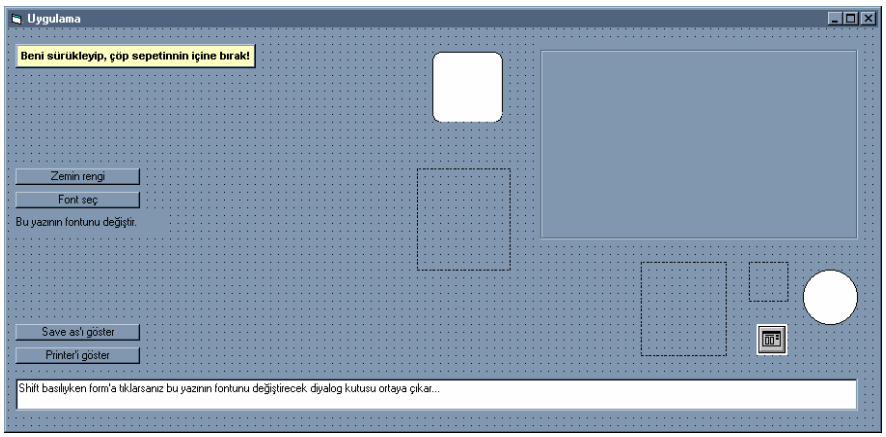
Şekil 2.6

4. Formumuza eklediğimiz Shape1 ve Shape2 nesnelerinin geometrik şekillerini “Properties” penceresini kullanarak değiştireceğiz. Shape1 nesnesinin geometrik formunun çember olması için , shape1 seçili konumda iken properties penceresindeki shape1 nesnesinin “Shape” özelliğini "3-Circle” olarak değiştirin. (Şekil 2.4)

Shape2 nesnesi ise şekil 2.6 da görüldüğü gibi yuvarlak köşeli dikdörtgen formundadır. Bunun için ise yine shape2 nesnesi seçili iken properties penceresinden, nesnenin “Shape” özelliği “4-Rounded Rectangle” olarak değiştirilir.

5. Shape1 ve shape2 nesnelerinin iç kısımlarını herhangi bir renk ile doldurabilmek için öncelikle her ikisi içinde properties penceresinden “BackStyle” özelliklerinin “Opaque” olarak değiştirilmesi gerekir. Daha sonra yine properties penceresinden “BackColor” özelliği kullanılarak istenilen renk seçimi yapılır.

Uygulamamızda paletten beyaz renk seçilerek nesnelerin şekil 2.6 daki görünümleri kazanması sağlanmıştır. Ayrıca nesnelerin bu zemin renkleri üzerine “FillStyle” ve “FillColor” özellikleri kullanılarak düz, desenli dolgular yapılabilir.

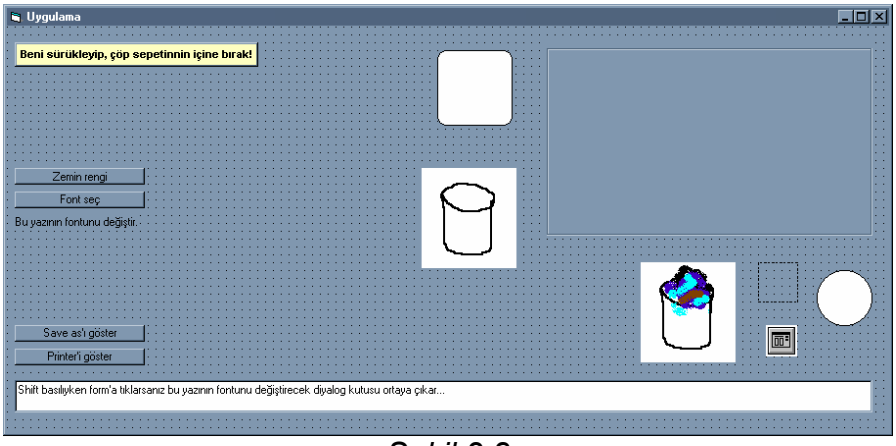


Şekil 2.7

6. Form üzerindeki formun, command butonlarının, label nesnesinin ve text kutusunun başlıklarını şekil 2.7 de görüldüğü gibi değiştireceğiz. Bu işlem için daha önceki uygulamamızda olduğu gibi yine properties penceresinin kullanacağız. Command butonlarının,formun ve labellerin başlık özelliğini properties pencersinde "Caption" özelliği kontrol eder.

Textbox kontrolü ise yazı girdi kontrolü olarak kullanıldığı için, textbox içerisinde bulunan yazı, properties penceresinde yer alan "Text" özelliği ifade eder. Şekil 2.7 de görülen başlıklar bu nesnelerin properties penceresindeki ilgili özellikleri karşısındaki kutulara girilir.

7. Windows ortamında çok sık kullanılan command butonların zemin renginin değiştirilmesi, başlığının yazı karekterinin değiştirilmesi gibi görsel işlemler Visual Basic 'de olanaklıdır. Örneğin bu uygulamamızda command1 nesnesinin properties penceresinden "Style" özelliğini "1-Graphical" yapıp daha sonra "BackColor" özelliği ile herhangi bir renk seçerek Şekil 2.7 de görüldüğü gibi değiştirelim. Ayrıca başlığın yazı karekterini değiştirmek için "Font" özelliğini kullanılır.



Şekil 2.8

8. Command1 nesnesinin "Drag-Drop" özelliğinden yararlanmak için bu nesnenin "DragMode" özelliğini properties penceresinden "1-Automatic" olarak değiştirin.

9. Formumuzun oluşturulmasında son işlem basamağı olarak image1 ve image2 nesnelerine resim ataması yapacağız. Image3 nesnesine ise bu basamakta herhangi bir resim ataması yapmayacağız. Bunun nedeni ise aşağıda kod yazımında anlaşılacaktır. İlk olarak image1 nesnesine atama yapalım. Bunun için properties penceresinde image1 nesnesinin özellikleri arasında yer alan "Picture" özelliğine

tıklamalı ve daha sonra açılan diyalog penceresinden atanacak resim dosyası seçilmelidir. Aynı işlemi image2 nesnesi içinde yapmalısınız. Uygulamada image1 nesnesine boş bir çöp kutusu, image2 nesnesine ise dolu çöp kutusu resmi atanmıştır.(Şekil 2.8)

10. Artık programımızın kod yazım basamaklarını gerçekleştireceğiz. Bu işleme command2, command3, command4, command5 nesnelerinin tıklama olaylarını yazarak başlayacağız. Öncelikle önceki uygulamamızda olduğu gibi kod yazım penceresini ekrana getirin.

Şekil 2.9

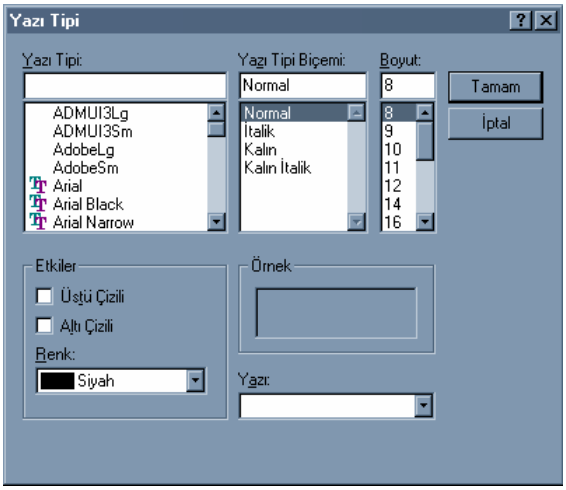
11. Kod penceresinin nesneler kısmından "Command2" 'yi seçin. Kod penceresinin olaylar kısmından ise "Click" olayını seçin. Bu seçim ile otomatik olarak "Private Sub Command2_Click()" ve "End Sub " satırları görülecektir. Command2 nesnesinin klik olayı için kodlarımızı aşağıda görüldüğü gibi bu iki satır arasına yazacağız.

```
Private Sub Command2_Click()
CommonDialog1.ShowColor
Command2.BackColor = CommonDialog1.Color
Form1.BackColor = CommonDialog1.Color
Text1.BackColor = CommonDialog1.Color
End Sub
```

Programımızın kullanıcısı command2 butonuna tıkladığında yukarıda görülen satırlardaki işlemler, aşağıdaki gibi sırasıyla yerine getirilecektir.

"CommonDialog1.ShowColor" satırı, commondialog1 kontrolümüz bünyesinde bulunan renk seçimi iletişim kutusu ekrana gelmesini sağlayacak. (Şekil 2.9)

Diğer 3 satır ise sırasıyla command2,form1 ve text1 nesnelerinin zemin rengini, renk seçimi iletişim kutusundan seçilen renk yapacak.



Şekil 2.10

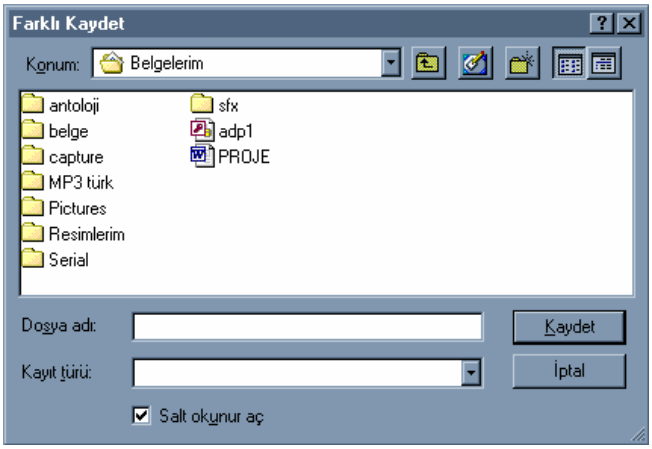
12. Command3 nesnesinin "Click" olayı için aşağıdaki kodlar yazılmalıdır.

```
Private Sub Command3_Click()
CommonDialog1.Flags = cdlCFBoth Or cdlCFEffects
CommonDialog1.ShowFont
Label1.Font.Name = CommonDialog1.FontName
Label1.Font.Size = CommonDialog1.FontSize
Label1.Font.Bold = CommonDialog1.FontBold
Label1.Font.Italic = CommonDialog1.FontItalic
Label1.ForeColor = CommonDialog1.Color
Label1.FontStrikethru = CommonDialog1.FontStrikethru
Label1.FontUnderline = CommonDialog1.FontUnderline
End Sub
```

"CommonDialog1.Flags = cdlCFBoth Or cdlCFEffects" satırı ile "Standart İletişim Kutusu Bayrakları" terimi karşımıza çıkmaktadır. Bu bayraklar yardımıyla iletişim kutularının çeşitli ayarlarını yapabiliriz. Örneğin bu satırla, yazı tipi iletişim kutusunda, ekran ve yazıcı için kullanılabilecek yazı tiplerinin yer alması için "cdlCFBoth" bayrağı kullanılmıştır. Eğer bu bayrağı kullanmamış olsaydık iletişim penceresinde herhangi bir yazı tipi olmazdı. "cdlCFEffects" bayrağını ise yazı tipi iletişim kutusunda çizgi, altçizgi ve renk efektlerinin yer alması için kullandık.

"CommonDialog1.ShowFont" komut satırı, yazı tipi iletişim kutusunun ekranda görülmesini sağlayacak. (Şekil 2.10)

Diğer satırlar ise label1 nesnesinin sırasıyla yazıtipi, yazı tipi boyutu, kalın ve eğik yazı stili, rengi, üstü çizili ve alt çizgi özelliklerini, iletişim kutusundan seçilen özelliklere göre değiştirirler.



Şekil 2.11

13. Command4 nesnesinin "Click" olayı için aşağıdaki kodlar yazılmalıdır.

```
Private Sub Command4_Click()
CommonDialog1.ShowSave
End Sub
```

Yazdığımız bu tek komut satırı ile şekil 2.11'de görülen kayıt iletişim kutusunun ekranda görülmesi sağlanır. Bu iletişim kutusu için ekrana getirilmesi dışında herhangi bir kod yazılmadığı için iletişim kutusunda yapılan seçimlerin program üzerine herhangi bir etkisi olmayacaktır.

14. Command4 nesnesinin "Click" olayı için aşağıdaki kodlar yazılmalıdır.

```
Private Sub Command5_Click()
CommonDialog1.ShowPrinter
End Sub
```

Bu satırda ise yazıcı iletişim kutusunun ekranda görülmesi amaçlanmıştır. Ancak bilgisayarınıza yüklü bir yazıcı yoksa, program hata mesajı verecektir. Yukarıda olduğu gibi bu iletişim kutusu için de ayrıca herhangi bir kod yazılmadığı için iletişim kutusunda yapılan seçimlerin program üzerine herhangi bir etkisi olmayacaktır.

15. Fare, form üzerinde gezdirilirken herhangi bir ektuşla yada ektuş olmadan, farenin butonlarının birisine basıldığında devreye girecek olan kodlar, aşağıdaki gibi form nesnesinin "Mouse Down" olayına yazılmalıdır.

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Shift = vbShiftMask Then
            CommonDialog1.Flags = cdlCFBoth Or cdlCFEffects
            CommonDialog1.ShowFont
             Text1.Font.Name = CommonDialog1.FontName
             Text1.Font.Size = CommonDialog1.FontSize
             Text1.Font.Bold = CommonDialog1.FontBold
             Text1.Font.Italic = CommonDialog1.FontItalic
             Text1.ForeColor = CommonDialog1.Color
             Text1.FontStrikethru = CommonDialog1.FontStrikethru
             Text1.FontUnderline = CommonDialog1.FontUnderline
```

```
End If

If Shift = vbCtrlMask And Button = vbLeftButton Then
        CommonDialog1.ShowPrinter
End If

If Shift = vbAltMask And Button = vbRightButton Then
        CommonDialog1.ShowColor
        Command2.BackColor = CommonDialog1.Color
        Form1.BackColor = CommonDialog1.Color
        Text1.BackColor = CommonDialog1.Color
End If
End Sub
```

Yukarıda anlatılan olay gerçekleştiğinde, yukarıdaki 3 şart kontrolü yapılacaktır.

If Shift = vbShiftMask Then ve End If satırları arasındaki komutlar eğer "Shift = vbShiftMask " şartı sağlanmış ise gerçekleşecektir. "Shift = vbShiftMask" şartı ise şudur: Fare tıklandığında, klavyede basılı olan tuş (Shift), klavyenin shift tuşu (vbShiftMask) olmalıdır. Yani form üzerinde herhangi bir yerde shift tuşuna basarak farenin herhangi bir butonuna basıldığında, bu iki satır arasında yazılı olan komutlar gerçekleşecektir. Aradaki bu komutların işlevi ise işlem basamağı 12'dekilerle aynıdır.

If Shift = vbCtrlMask And Button = vbLeftButton Then ve End If satırları arasındaki komut ise eğer "Shift = vbCtrlMask " ve "Button = vbLeftButton " şartları sağlanmış ise gerçekleşecektir. Burada yukarıdaki satırdan farklı olarak iki şart istenmiştir. Şartlar ise şunlardır: Fare tıklandığında, klavyede basılı olan tuş (Shift), klavyenin ctrl tuşu (vbCtrlMask) ve farenin tıklanan butonu sol buton " Button = vbLeftButton" olmalıdır. Yani form üzerinde herhangi bir yerde ctrl tuşuna basarak farenin sol butonuna basıldığında, bu iki satır arasında yazılı olan komutlar gerçekleşecektir. Aradaki bu komutun işlevi ise işlem basamağı 14' de anlatıldı.

If Shift = vbAltMask And Button = vbRightButton Then ve End If satırları arasındaki komut ise eğer "Shift = vbAltMask " ve "Button = vbRightButton " şartları sağlanmış ise gerçekleşecektir. Burada ise istenen şartlar şunlardır: Fare tıklandığında, klavyede basılı olan tuş (Shift), klavyenin alt tuşu (vbAltMask) ve farenin tıklanan butonu sağ buton " Button = vbRightButton" olmalıdır. Aradaki komutun anlamı ise işlem basamağı 11' de anlatıldı.

16. Formun yüklendiğinde, shape nesnelerinin bazı özelliklerinin değiştirilmesi için aşağıdaki kodları yazmalıyız. Kodlar form nesnesinin "Load" olayına yazılmalıdır.

```
Private Sub Form_Load()
Shape1.DrawMode = 16
Shape2.FillStyle = 7
Set Shape1.Container = Frame1
Image2.Visible=False
Image3.Visible=False
End Sub
```

İlk satır ile shape1 nesnesinin çizim şekli yani “DrawMode” özelliği 16 ya eşitlenerek “Whiteness” seçilmiştir. Böylece shape1 nesnemiz tamamen beyaz görünecektir. İkinci satır ise shape2 nesnesinin içinin dolduruluş şeklini yani “FillStyle” özelliğini 7 ‘ye eşitleyerek, nesnenin içini çapraz çizgilerle dolduracaktır. Bu iki işlemi kod yazmaya gerek kalmaksızın properties penceresinden de yapabiliriz.

Üçüncü satırda ise shape1 nesnesinin frame1’e bağlı olduğu belirtiliyor. Bu satırla artık frame1, shape1’i kapsar. Shape1, frame1 in dışına çıkamaz.

Son iki satırda, nedeni daha sonraki işlem basamaklarında anlatılacak olan image2 ve image3 nesnelerinin görünmezliği sağlanmıştır.

17. Kod yazımında gerçekleşen son olay ise; fare form üzerinde gezdirilirken, farenin x ve y koordinatlarına bağlı olarak nesneler üzerinde gerçekleşecek olaylardır. Bunlar için aşağıdaki kodlar formun “Mouse Move” olayına yazılmalıdır.

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Shape1.FillColor = RGB(X / 10, Y / 10, 30)
Shape2.FillColor = RGB(Y / 10, X / 10, 30)
Shape1.Move X + 100, Y + 100
Frame1.BackColor = RGB(X / 10, Y / 10, 255)
If X > 3000 Then
Shape2.BorderWidth = X / 500
End If
End Sub
```

Öncelikle şu bilinmelidir ki, formumuzun şimdi kullanmış olduğumuz “Mouse Move” olayında, farenin koordinatları “x” ve “y”, kullanıldıysa fare butonu “Button” ve kullanıldıysa ektuş “Shift” değişkenleriyle tanımlanmıştır ve bu değerler temsil ettikleri olaya göre otomatik olarak değerlerini kazanırlar.

İlk iki satırla shape1 ve shape2 nesnelerinin dolgu renkleri farenin x,y koordinatlarına bağlı kılınmıştır. RGB renk modu üç ana renkten oluşur. Bunlar sırasıyla kırmızı, yeşil ve mavidir. Bütün renkler bu üç rengin çeşitli oranlarda karışımı ile oluşur. Bu üç rengin her biri 0 ve 255 arası değerler alırlar. “255”, O rengin en açık tonunu; “0” ise en koyu tonu olan siyahı; ara değerler ise ara tonları verir. Bunun sonucunda her üçü de 0’a eşit olduğunda ortaya çıkan renk siyah, her biri 255 olursa ortaya çıkan renk beyaz olur.

Biz ilk satırda sahpe1 nesnesinin dolgu renginin; x koordinatının 10 da 1 i tonunda kırmızı, y koordinatının 10 da biri tonunda yeşil ve 30 tonunda da mavi olmasını istedik.

İkinci satırda ise sahpe2 nesnesinin dolgu renginin; y koordinatının 10 da 1 i tonunda kırmızı, x koordinatının 10 da biri tonunda yeşil ve 30 tonunda da mavi olmasını istedik.

X ve y ye bağlı değerler 255 i geçtiğinde, yine 255 gibi işlem görülür.

Üçüncü satırda shape1 nesnesinin move özelliği kullanılarak hareket etmesini sağlıyoruz. Ancak burada dikkat edilmesi gereken konu, frame1 nesnesinin shape1 nesnesini kapsamasıdır. Dolayısıyla shape1’in hareketi için için verdiğimiz koordinatlar frame1 için geçerlidir. Örneğin “Shape1.move 0,0” komut satırını kullansak, shape1 formun sol üst köşesine değil frame1’in sol üst köşesine gider. Bu satırda x+100 ve y+100 kullandığımız için shape1 daima x ve y değerlerinin 100 fazlasında bulunacaktır. Yani fare işaretçisini programımız çalıştırıldığında formun sol üst köşesine götürülse; shape1, frame1 ‘in sol üst köşesinden her iki koordinatta 100 kadar ileride olacaktır.

Frame1.BackColor = RGB(X / 10, Y / 10, 255) satırı ile yaptırılan işlem ilk iki satırdakinin aynıdır. Bu satırda frame1'in zemin rengi x ve y ye bağlı olarak değişecektir. Ancak burada farklı olan ise, daima en açık mavi tonunun zemin rengi içinde bulunacağıdır.

```
If X > 3000 Then
Shape2.BorderWidth = X / 500
End If
```

Mouse move olayında kullandığımız bu son komut satırında, yine bir şart sunuyoruz. Eğer farenin x değeri 3000'den fazla olursa shape2 nin çerçeve kalınlığı, x koordinatının 500 de biri kadar olacaktır. Program ilk çalıştırıldığında çerçeve kalınklığı 1 olacak, farenin x koordinatı 3000'i geçtiğinde kalınlık bu artışla orantılı artacak ve azalacaktır. Eğer formunuzun genişliği 3000 den az ise, çerçeve kalınlığı hiç değişmeyecektir.

18. Herhangi bir nesne image1 nesnesi üzerine sürüklenip bırakıldığında aşağıdaki komutlar gerçekleşecektir. Ancak uygulamamızda yalnızca command1 nesnesi için "Drag-Drop" özelliğini aktif ettiğimizden dolayı, image1 üzerine command1 bırakıldığında bu olay gerçekleşecektir. Bu komutları kod penceresinde image1 nesnesi için "DragDrop" olayına yazacağız.

```
Private Sub Image1_DragDrop(Source As Control, X As Single, Y As Single)
Image3.Picture = Image1.Picture
Image1.Picture = Image2.Picture
End Sub
```

İlk durumda image1 nesnesinin resmi şekil 2.8 'de ki gibi boş bir çöp kutusudur. Image2 nesnesinin resmi ise dolu çöp kutusudur. Image3'e başlangıçta herhangi bir resim ataması yapılmadığı için boştur.

Yapmak istediğimiz command1 nesnesi image1 nesnesine (boş çöp kutusuna) bırakıldığında, image1 nesnesine dolu çöp kutusu resmi atanmasıdır.

Bunun için ilk satırda daha sonra kullanılmak üzere image1 'in resmi(boş çöp kutusu) image3 'e atanır.

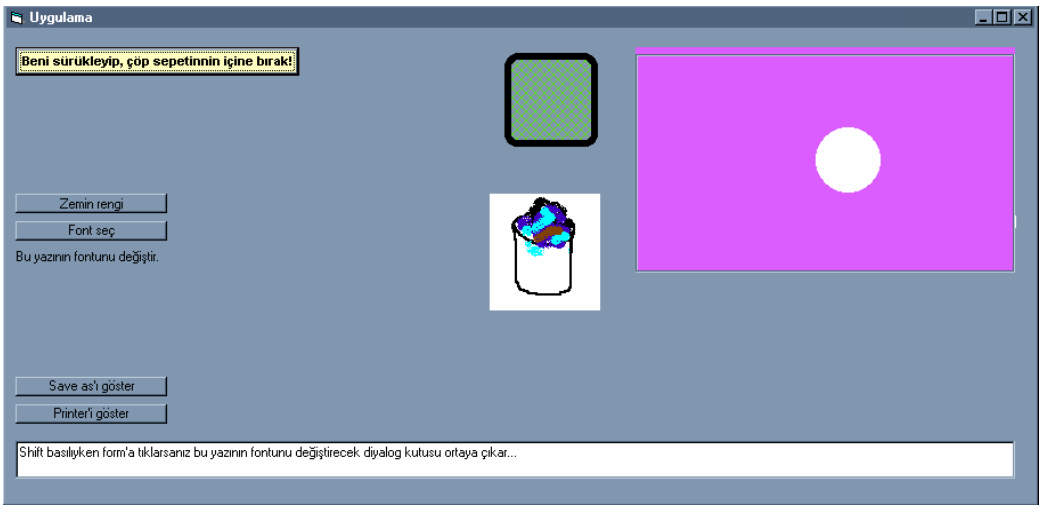
İkinci satırda ise image1 nesnesine image2 nesnesinin resmi (dolu çöp kutusu) atanır.

Böylece sürükle-bırak olayı gerçekleştiğinde image1 nesnesi şekil 2.12 'de görüldüğü gibi dolu çöp kutusu görünümü alır. Program çalıştırıldığında Image2 ve image3 nesneleri şekil 2.12'de görüldüğü gibi görünmezdirler. ("Visible" özellikleri "False" dır.)

19. Son olarak image1 nesnesi üzerine çift tıklandığında, tekrar boş çöp kutusu görünümü kazanmasını sağlayan kodu,image1 nesnesinin "DblClick" olayına yazacağız.

```
Private Sub Image1_DblClick()
Image1.Picture = Image3.Picture
End Sub
```

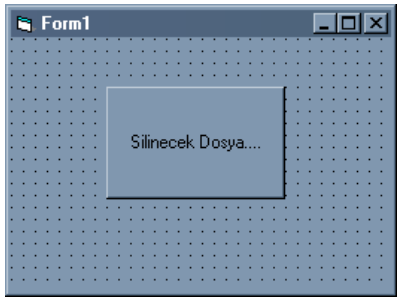
Bu komut satırı ile yukarıda image3 nesnesine atadığımız dolu çöp kutusu resmini image1 nesnesine geri atıyoruz.Böylece image1 tekrar şekil 2.8 de ki görüntüsünü kazanır.


Uygulama
Beni sürükleyip, çöp sepetinnin içine bırak!
Zemin rengi
Font seç
Bu yazının fontunu değiştir.
Save as'ı göster
Printer'ı göster
Shift basılıyken form'a tıklarsanız bu yazının fontunu değiştirecek diyalog kutusu ortaya çıkar...

Şekil 2.12

UYGULAMA 3

Bu uygulamada Windows işletim sisteminin temel taşlarından olan mesaj ve giriş kutularını kullanarak dosya silme işlemi gerçekleştireceğiz. Ayrıca bu uygulama ile, yazdığımız herhangi bir programın kullanımı sırasında oluşabilecek hatalara karşı tedbir alacağız.



Şekil 3.1

1. Şekil 3.1 de görülen ve üzerinde sadece tek bir “Command Button” kontrolü bulunan formu oluşturun.

2. Programımızda işlemlerin hepsi, formumuz üzerinde bulunan butona tıklanması ile gerçekleşecektir. Aşağıdaki kodları “Command1” nesnesinin “Click” olayına yazın.

```
Private Sub Command1_Click()
dosya = InputBox("Silinecek Dosya ismi", "Sil", , 3000, 3000)
If dosya = "" Then  MsgBox "Dosya adı girmelisiniz.", vbOKOnly + vbCritical,
"Dosya Silinemedi"
If dosya <> "" Then
   On Error GoTo Hata
   Kill dosya
Hata:
   MsgBox "Dosya Silinemedi!", vbOKOnly + vbCritical, "Hata Oluştu"
End If
End Sub
```

Command1 nesnesine tıklandığında yukarıdaki kodlar sırasıyla işlemlerini yerine getireceklerdir.

İlk satırda “Dosya” değişkenine “Inputbox” tan elde edilen veri atanır. Inputbox ‘ın söz dizilimine ilk sırada “Prompt” yer alır ve kutuda gösterilecek metni ifade eder.Söz diziliminde daha sonra sırasıyla mesaj kutusunun başlığı (Title), kutuda yer alan ve veri girişini sağlayan textbox’ın ilk durum metni (Default), input mesaj kutusunun x koordinatı (x pos) ve y koordinatı (y pos) yer alır. Söz diziliminde atlamak istediğiniz herhangi bir özellik varsa, bunun için herhangi bir metin koymadan,bir sonraki özelliğe virgül yardımıyla geçmek yeterlidir.

Uygulamamızda kullandığımız kutunun x ve y koordinatları 3000; metni “Silinecek dosya ismi”; başlığı ise “Sil” olacaktır. Default özelliği virgül ile geçildiği için kutu ekrana geldiği ilk durumda şekil 3.2 deki gibi textbox’ı boş olacaktır.

Şekil 3.2

Kullanıcının inputbox ‘ın “Ok” butonuna tıkladıktan sonra iki durum söz konusudur. Kullanıcı ya hiçbir şey girmemiştir. Yada inputbox ın textbox’ına herhangi bir metin yazmıştır. Eğer ilk durum gerçekleşmiş ise kullanıcıya bu durum bildirilerek veri girmesi istenmelidir. Bunun için;

If dosya = "" Then MsgBox "Dosya adı girmelisiniz.", vbOKOnly + vbCritical, "Dosya Silinemedi"

komut satırı yazılmıştır. Inputbox tan elde edilen verinin atandığı “Dosya” değişkeni eğer boş ise mesaj kutumuz ekranda yer alacaktır.

Mesaj kutularının söz dizilimi ise sırasıyla prompt, buttons, title, helpfile, context şeklindedir. Mesaj kutusunda, giriş kutusundan farklı olarak butonlar ve simgeler yer almaktadır.

Örneğin bizim “Dosya silinemedi” başlıklı ve üzerinde “Dosya adı girmelisiniz” metni bulunan mesaj kutumuzda sadece “OK” butonu ve “Critical” simgesi yer alacaktır.(Şekil 3.3) Eğer mesaj kutusunda basılan butonun değerini bir değişkene atamamız gerekiyor ise parametreler, parantez içine yazılmalıdır.

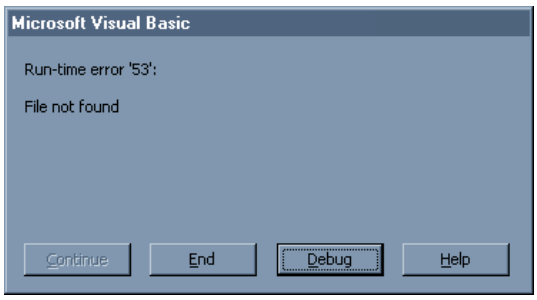
Şekil 3.3

İkinci durum yani inputbox a veri girişi gerçekleşmiş ise, “Dosya” değişkenimiz boş olmayacaktır ve ikinci “IF” bloğumuz çalışacaktır.

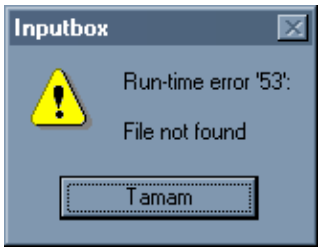
Kullanıcı “InputBox” ‘ın “TextBox”’ına, silmek istediği dosyanın yolunu ve adını girdi ve veri “Dosya” değişkenine atandı. İşte bu safha bir çok hata oluşmasına neden olabilir. Örneğin kullanıcı dosyanın adını veya yolunu yanlış yazmış olabilir, olmayan herhangi bir dosyayı silmek isteyebilir.

Yukarıda da bahsettiğimiz gibi genelde program yazılımı sırasında, programımın kullanım safhasında oluşabilecek hatalar göz önünde bulundurulmalıdır.

Bu hata oluşumu göz önüne alınmazsa ve Visual Basic ortamında program çalıştırılarak olmayan dosyayı silmek istediğinizde Şekil 3.4 de görülen hata mesajı karşınıza çıkar. Eğer program “EXE” yani çalıştırılabilir dosya haline getirilip çalıştırıldığında aynı hata yapılırsa Şekil 3.5 de görülen hata mesajı çıkar. Bu hata mesaj kutusunun butonuna tıklanması ile program Windows tarafından sonlandırılır.



Şekil 3.4



Şekil 3.5

Bu hatayı ortadan kaldırmak için ikinci "If" bloğumuzun başına;

On Error GoTo Hata

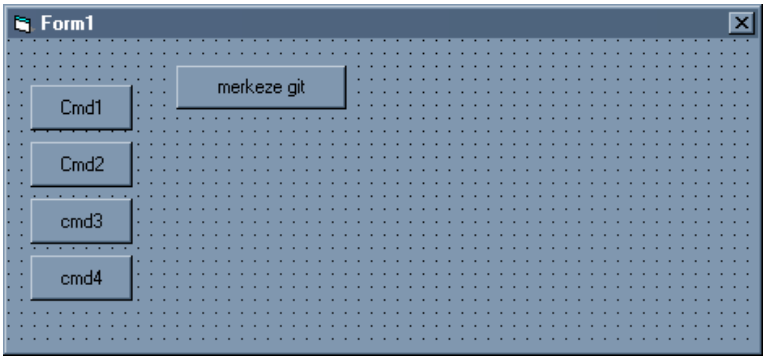
komut satırını ekledik. Bu satırın görevi, "If" bloğu içerisinde oluşacak herhangi bir hata durumunda, programı "HATA" satırından devam ettirmektir. Hata satırında ise ikinci bir mesaj kutusu ekrana getirilir ve kullanıcıya silme işleminin gerçekleştirilemediği bildirilir.(Şekil 3.6) Böylece hata atlatılmış olunur.

Şekil 3.6

Programımız çalıştırıldığında eğer varolan bir dosyanın adı ve yolu inputbox a doğru girilirse ("c:\windows\desktop\ses.wav" şeklinde) , herhangi bir hata oluşmayacak ve ikinci "If" bloğumuz içerisinde yer alan "Kill Dosya" komut satırı ile dosya silinecektir. "Kill" komutu anladığınız gibi dosya silmek için kullanılır ve komutun yanına dosya yolu ve adı yazılır. Dosya adı ve yolunu, inputbox tan aldığımız "Dosya" değişkeni tanımlayacaktır.

UYGULAMA 4

Yazdığımız programların görünür yüzü olan formların ekran üzerindeki koordinatlarını, “Form Layout” penceresinden belirlediğimizde, ancak başlangıç koşulunu belirlemiş oluruz. Program kullanımı sırasında formun koordinatlarının değiştirilmesini, bu uygulama kullanacağız.



Şekil 4.1

1. Şekil 4.1 de görülen ve üzerinde beş tane “Command Button” kontrolü bulunan formu oluşturun. Bu uygulamada form üzerinde bulunan kontrollerin özelliklerini “Properties “ penceresinden değil, kodlarla belirleyeceğiz.

2. Command butonlarının dört teanesinin, “Name” ve “Caption” özelliklerini “Properties” penceresinden aşağıdaki gibi değiştirin.

Caption	Name
Cmd1	cmdTopLeft
Cmd2	cmdTopRight
Cmd3	cmdBottomLeft
Cmd4	cmdBottomRight
Merkeze git	Command1

3. Formumuzun “Load” olayına aşağıdaki kodları yazın. Bu kodlar ile dört butonu formun dört köşesine yerleştireceğiz. Ayrıca yine bu butonların yukarıdaki “Caption” özelliklerini kazandıracağız.

```
Private Sub Form_Load()
   cmdTopLeft.Caption = "Üst Sol"
   cmdTopRight.Caption = "üst sağ"
   cmdBottomLeft.Caption = "alt sol"
   cmdBottomRight.Caption = "alt sağ"
   cmdTopRight.BackColor = RGB(1, 200, 100)
   cmdBottomLeft.BackColor = RGB(250, 200, 100)
   Form1.Caption = "Hareket Ettir"
   Form1.Left = (Screen.Width - Form1.Width) / 2
   Form1.Top = (Screen.Height - Form1.Height) / 2
   cmdTopLeft.Top = 200
   cmdTopLeft.Left = 200
   cmdTopRight.Top = 200
   cmdTopRight.Left = Form1.Width - cmdTopRight.Width - 300
   cmdBottomRight.Top = Form1.Height - cmdBottomRight.Height - 500
```

```
    cmdBottomRight.Left = Form1.Width - cmdBottomRight.Width - 300
    cmdBottomLeft.Top = Form1.Height - cmdBottomLeft.Height - 500
    cmdBottomLeft.Left = 200

End Sub
```

İlk dört satırla yukarıda verilen tablodaki “caption” özellikleri, formun yüklenmesiyle dört butonumuza yerleştirilecektir.ü

Daha sonraki iki satır ile “sağ-üst” ve “sol-alt” butonlarının zemin rengi “RGB” renk modunda tanımlanmıştır.

“Form1.Caption...” satırı ile formun başlığı “Hareket Ettir” olarak değiştirilmiştir.

“Form1.Left.....” satırının işlevi, formun açılışta sol koordinatını atamaktır. Burada “Screen.Width” yani ekranın genişliğinden, “Form1.Widtht” yani formun genişliği çıkarılarak elde edilen sayıyı ikiye bölerek, formun yatayda ortalanması sağlanmıştır.

“Form1.Top.....” satırı ise bir üst satırdan farksızdır. Ancak burada “Top” özelliği kullanılarak, formun dikeyde ortalanması sağlanmıştır.

“cmdTopLeft.Top = 200” ve “ cmdTopLeft.Left = 200 “ satırları “Topleft” isimli butonun “Top” ve “Left” yani “x” ve “y” koordinatları tanımlanmıştır. Böylece bu buton formun sol-üst köşesinde yer alacaktır.

Diğer satırlarda bir önceki gibi butonların “x” ve “y” koordinatlarını tanımlar. Diğer butonlar sırasıyla formun sağ-üst, sağ-alt ve sol-alt köşelerine yerleştirilir.

3. Aşağıdaki komut satırlarını “Merkeze git” başlıklı yani “Command1” nesnesinin “Click” olayına yazın.

```
Private Sub Command1_Click()
Form1.Top = (Screen.Height - Form1.Height) / 2
Form1.Left = (Screen.Width - Form1.Width) / 2
End Sub
```

Bu iki satır, “Form Load” olayında olduğu gibi, formun merkeze gitmesini sağlamaktadır.

4. “cmdTopleft” isimli ve formumuzun sol-üst köşesinde görünmesini sağladığımız buton nesnemizin “Click” olayına aşağıdaki komut satırlarını yazın.

```
Private Sub cmdTopLeft_Click()
Form1.Top = 0
Form1.Left = 0
End Sub
```

Bu nesneye tıklandığında formumuzun “Top” ve “Left” koordinatları sıfır değerini kazanarak, formun ekranın sol-üst köşesinde görünmesi sağlanacaktır.

5. Formun sağ-üst köşesinde bulunan “cmdTopright” nesnesinin “Click” olayına aşağıdaki satırları yazınız.

```
Private Sub cmdTopRight_Click()
Form1.Top = 0
Form1.Left = Screen.Width - Form1.Width
```

```
End Sub
```

Formun üstte yer alması için “Top” özelliği sıfıra eşitlenmiştir. Verdiğimiz koordinatların, formun sol-üst köşesinin koordinatı olduğundan dolayı ekranın genişliğinden, formun genişliği çıkarılarak sağa dayanması sağlanmıştır.

6. “cmdBottomright” nesnesinin “Click” olayına aşağıdaki kodları yazınız.

```
Private Sub cmdBottomRight_Click()
Form1.Top = Screen.Height - Form1.Height
Form1.Left = Screen.Width - Form1.Width
End Sub
```

Önceki satırlarda olduğu gibi, bu seferde ekranın genişlik ve yüksekliğinden, formun genişliğini ve yüksekliğini çıkararak, formun sağ alt köşede görünmesi sağlanmıştır.

7. “cmdBottomleft” nesnesinin “Click” olayına aşağıdaki kodları yazınız.

```
Private Sub cmdBottomLeft_Click()
Form1.Top = Screen.Height - Form1.Height
Form1.Left = 0
End Sub
```

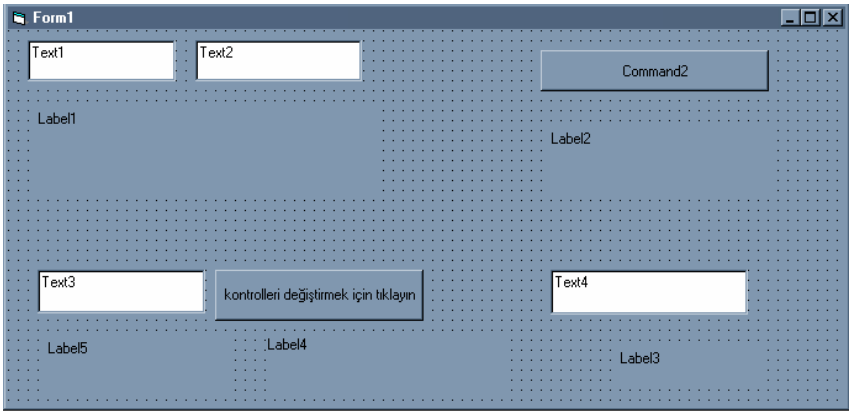
Formun “Left” özelliği sıfıra eşitlenerek, form sola; ekran yüksekliğinden, formun yüksekliği çıkarılarak form aşağıya yerleştirilir.

8. Program çalıştırıldığında ilk olarak formun görünümü şekil 4.2 deki gibi olacaktır. Daha sonra butonlara tıklanması ile form, ekranın köşelerinde gezdirilir.

Şekil 4.2

UYGULAMA 5

Diğer uygulamalarda form üzerinde kullandığımız kontrollerin özelliklerini birer birer değiştirmiştik. Bu uygulama ile ise form üzerindeki bir çok kontrolün özelliğini tek işlemle değiştireceğiz.



Şekil 5.1

1. Şekil 5.1 de görülen formu oluşturun.

2. Kontrollerden "Command1" nesnesinin "Caption" özelliğini "Properties" penceresinden " Kontrolleri değiştirmek için tıklayın" olarak değiştirin.

3. Formumuza yerleştirdiğimiz bu kontollerden sadece "Command1" butonu için kod yazacağız. Aşağıdaki kodları "Command1" nesnesinin "Click" olayına yazın.

```
Private Sub Command1_Click()
Dim iControl As Control
   For Each iControl In Controls
      If (TypeOf iControl Is TextBox) Or (TypeOf iControl Is Label) Or (TypeOf
iControl Is CommandButton) Then
         iControl.BackColor = RGB(200, 200, 0)
         iControl.FontSize = 20
         iControl.Font = "arial"
        End If
   Next
End Sub
```

Programın tüm işlevleri bu satırlardan oluşmaktadır. Amacımız form üzerinde yer alan tüm "Textbox", "Label" ve "Commandbutton" tipi kontrollerin zeminrengini, yazı tipi boyutunu ve yazı tipini değiştirmektedir.

"Dim iControl As Control" satırı ile "iControl" değişkeni tanımlanmaktadır. Ancak diğer uygulamalardan farklı olarak bu değişkenin tipi "Control" tipidir. Yani "iControl" değişkeni "Label", "TextBox" gibi kontrol tiplerine atanacaktır.

"For Each iControl In Controls" ve "Next" satırları ile döngü oluşturulmuştur. Ancak bu döngü, diğer uygulamalarda kullandığımız "For-Next" döngülerinden farklıdır. Çünkü bu döngüde kullandığımız değişken sayısal değerler değil, kontrol isimlerine atanacaktır. "iControl" değişkeni, sırayla form üzerindeki tüm kontrol isimlerine atanacaktır. Yani döngü tekrar sayımız, form üzerindeki kontrol

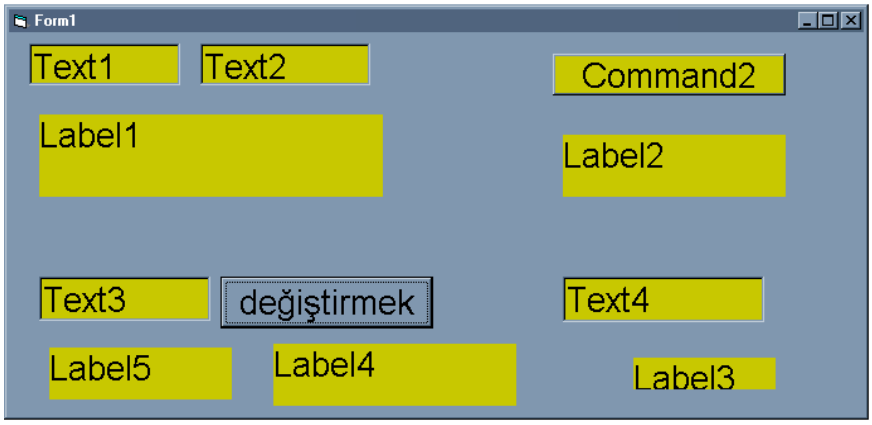
sayısına eşittir ve “iControl” değişkeni her tekrarda farklı bir kontrol ismine sahip olacaktır.

If satırında ise karşımıza yeni bir komut gelmektedir. “TypeOf” komutunun işlevi, birlikte kullanıldığı kontrolün ne tip bir kontrol olduğunu belirtmektir. Uygulamamızda amacımız “Label”, “Textbox” ve “CommandButton” kontrollerinin ortak özellikleri olan zemin rengi, yazı tipi ve boyutlarını değiştirmektir. Örneğin “Image” kontrolünde yazı tipi özelliği bulunmamaktadır. Bu nedenle “iControl” değişkeni form üzerindeki tüm kontrollerin isimlerine atanacağından, istediğimiz kontrollerin dışındaki kontroller için özellik değiştirme satırlaının işlevsiz olması gerekir.

Bu nedenle bir “If” bloğu kullanılmıştır. Şartımız ise “iControl” değişkeninin kontrol tipinin “Label”, “Textbox” veya “CommandButton” olmasıdır. “TypeOf” komutunun (TypeOf iControl Is TextBox) şeklindeki kullanımıyla eğer “iControl” bir “Textbox” tipi kontrol ise, “(TypeOf iControl Is TextBox)=True” yani doğru olacak ve “If” boluğu arasındaki satırlar çalışmaya başlayacak. “TypeOf” komut kullanımı, aynı şekilde istediğimiz diğer iki kontrol tipi içinde kullanılmıştır. Böylece “iControl” istediğimiz üç kontrol tipinden biri ise “If” bloğu çalışacaktır.

“iControl.BackColor” , “iControl.FontSize” , “iControl.Font” komutlarının kullanımı ile “iContol” ün taşıdığı kontrolün sırasıyla zemin rengi, yazı tipi ve boyutu değiştirilecektir.

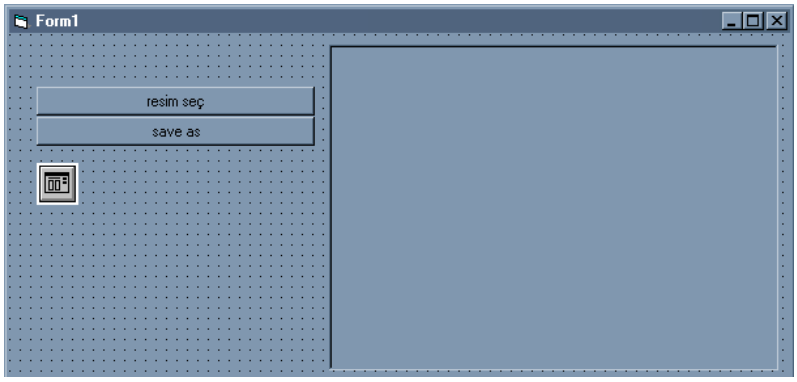
4. Program çalıştırıldığında ve “Command1” nesnesine tıklandığında, form üzerindeki tüm “Label”, “Textbox” ve “CommandButton” kontrollerinin görünümü şekil 5.2 de görüldüğü gibi olacaktır.



Şekil 5.2

UYGULAMA 6

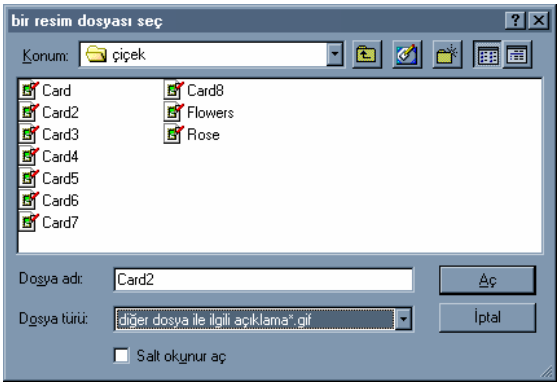
Gerçekleştireceğimiz bu uygulama ile, grafik programlarında sık olarak kullandığımız resim kayıt işlemi gerçekleştireceğiz.



Şekil 6.1

1. Şekil 6.1 de görülen ve üzerinde bir adet “CommonDiaolog”, iki adet “CommandButton” ve bir adet “PictureBox” kontrolü olan formu oluşturun.

2. “Command1” nesnesinin “Caption” özelliğini “Resim Seç”, “Command2” nesnesinin aynı özelliğini ise “Save as” olarak değiştirin.



Şekil 6.2

3. Aşağıda görülen kodları “Command1” nesnesinin “Click” olayına yazınız.

```
Private Sub Command1_Click()
    CommonDialog1.DialogTitle = "Bir resim dosyası seç"
    CommonDialog1.Filter = "diğer dosya ile ilgili açıklama*.gif |*.gif|resim dosyasıyla ilgili açıklama*.bmp|*.bmp"
    CommonDialog1.FilterIndex = 2
    CommonDialog1.ShowOpen
    Picture1.Picture = LoadPicture(CommonDialog1.FileName)
End Sub
```

Yukarıdaki komut satırları ile “Picture1” isimli “PictureBox” nesnesinde görülecek olan resim dosyasının seçimi için “Dosya aç iletişim kutusu” nun ekrana getirilmesini ve bu kutudan seçilen dosyanın “Picture1” nesnesinde gösterilmesi sağlanacaktır.

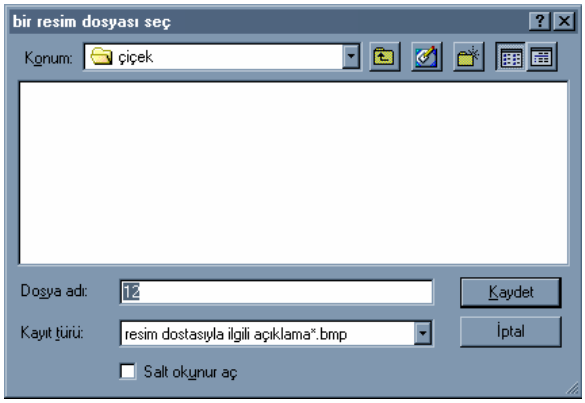
İlk satırda iletişim kutusunun başlığı "Bir resim dosyası seç" olarak belirtilmiştir.

İkinci satırda ise iletişim kutusunun “Filter” özelliği kullanılmıştır. “Filter” özelliği; iletişim kutusunda, sadece belirtilen uzantıya sahip dosyaların gösterilmesidir. Bu özelliğin kullanımında ise ilk olarak uzantı açıklaması yazılır ve “|” karakterinin ardından uzantı tipi yazılır. Birden fazla uzantı tipleri de yine “|” karakteri ile ayrılır.

Üçüncü satırda ise “FilterIndex” özelliği kullanılmıştır. Bu özellik ise iletişim kutusunun, ekrana geldiğinde bir üst satırda tanımladığımız dosya tiplerinden hangisinin seçili olacağını, ilgili dosya tipinin liste numarasına atanarak, belirler.

“CommonDialog1.ShowOpen” satırı ile “Dosya aç iletişim kutusu” ekrana getirilir.(Şekil 6.2)

Son satırla ise “Picture1” nesnesinin “Picture” özelliği yani gösterilecek olan resmi, iletişim kutusundan seçilen dosyanın “LoadPicture” komutu ile yüklenmesiyle ekranda gösterilir.



Şekil 6.3

4. Aşağıda görülen kodları “Command2” nesnesinin “Click” olayına yazınız.

```
Private Sub Command2_Click()
    CommonDialog1.Flags = cdlOFNOverwritePrompt
    CommonDialog1.FileName = "12"
    CommonDialog1.ShowSave
    On Error GoTo hata:
    SavePicture Picture1.Picture, CommonDialog1.FileName
    Exit Sub
    hata:
    MsgBox "çık"
End Sub
```

İkinci uygulamamızda bahsettiğimiz “Flag” kullanımı, bu uygulamamızda da yer alacaktır. İlk satırda “Kayıt iletişim kutusu” için “cdlOFNOverwritePrompt” bayrağı kullanılmıştır. Bu bayrağın amacı, windows kullanırken sık olarak karşılaştığımız, aynı dosya üzerine kayıt işleminin gerçekleştirilmesinde kullanıcıya bunun sorulması işlemidir.

İkinci satırda ise iletişim kutusunu açıldığında, dosya ismi yazılan kutusunda görülecek olan dosya ismidir. İletişim kutusu ilk açıldığında bu kısımda “12” yazacaktır. (Şekil 6.3)

“CommonDialog1.ShowSave” satırı ile “Kayıt iletişim kutusu” ekrana getirilir.

Artık kullanıcı bir kayıt dosya ismi seçmiştir. Ancak bir hata oluşması durumunu göz önüne alarak kayıt işlemine geçmeden önce, Uygulama 3 ‘de olduğu gibi “On Error” komutu kullanılmıştır. Ve bu komut kullanımı ile bir hata oluştuğunda, program “Hata” satırına atlayacaktır.

Hata satırında ise ekrana bir mesaj kutusu getirilerek bu olaydaki işlemler son bulacaktır.

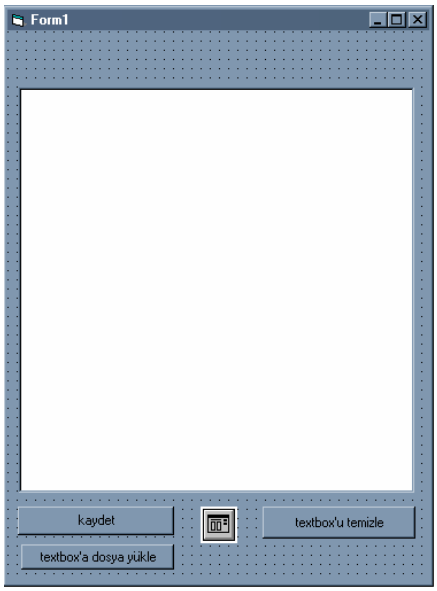
Eğer kayıt için seçilen dosyada bir hata oluşmazsa, “SavePicture” komutu ile “Picture1” deki resim, “CommonDialog1.FileName” adıyla yani iletişim kutusundan seçilen dosya adıyla, seçilen dosya yoluna kaydedilecektir.

5. Program çalıştırılarak “Resim seç” butonuna tıklandığında şekil 6.4 de görüldüğü gibi resim “Picture1” nesnesi üzerinde ekrana getirilecektir.

Şekil 6.4

UYGULAMA 7

Visual Basic kontrolleri arasında zengin işlevselliği nedeniyle ön plana çıkan “RichTextBox” kullanımına bu uygulamada yer veriyoruz. Bu kontrolü kullanarak dosyaları metin kutusuna taşıyacak ve kontroldeki bu metni tekrar kaydedeceğiz.



Şekil 7.1

1. Şekil 7.1 de görülen ve üzerinde bir adet “CommonDiaolog”, üç adet “CommandButton” ve bir adet “RichTextBox” kontrolü olan formu oluşturun.

2. “CommandButton” nesnelerinin “Caption” özelliklerini sırasıyla “Kaydet”, “TextBox’a Dosya Yükle” ve “TextBox’ı Temizle” olarak değiştirin.

3. Aşağıda görülen kodları “Command2” nesnesinin “Click” olayına yazınız

```
Private Sub Command2_Click()
        If CommonDialog1.FileName = "" Then
        CommonDialog1.ShowOpen
        End If
        RichTextBox1.LoadFile CommonDialog1.FileName, 1
End Sub
```

Uygulamamız ilk çalıştırıldığında, zengin metin kutusuna herhangi bir dosyayı iletişim kutusu yardımıyla yükleyeceğiz. Bu işlemden sonra sürekli aynı dosya ile çalışacağız. Yani bir daha iletişim kutusu ekrana gelmeyecek. Aynı dosya yüklenecek ve yine aynı dosyaya kayıt yapılacak.

Bunun sağlanması için “If” bloğu kullanıyoruz. Eğer “CommonDialog1.FileName” a herhangi bir atama yapılmamışsa yani herhangi bir dosya henüz yüklenmemiş ise “CommonDialog1.ShowOpen” satırı ile iletişim kutusu ekrana getirilir.

Eğer daha önce bir dosya açılmış ise “End if” ten sonra gelen satır işlem görecektir. Böylece zengin metin kutusunun “LoadFile” özelliği kullanılarak iletişim kutusundan seçilen dosya, metin kutusuna yüklenir.

4. Aşağıda görülen kodları “Command1” nesnesinin “Click” olayına yazınız.

```
Private Sub Command1_Click()
    If CommonDialog1.FileName = "" Then
    CommonDialog1.ShowSave
    End If
    RichTextBox1.SaveFile CommonDialog1.FileName, 1
End Sub
```

Eğer program çalıştırıldığında herhangi bir dosya yüklenmemiş, metin kutusuna yeni metin yazılmış ise, “CommonDialog1.FileName” ye herhangi bir atama yapılmamış olacak ve “Kayıt iletişim kutusu” ekrana getirilecektir. Kutudan seçilen dosyaya kayıt yapılacaktır.

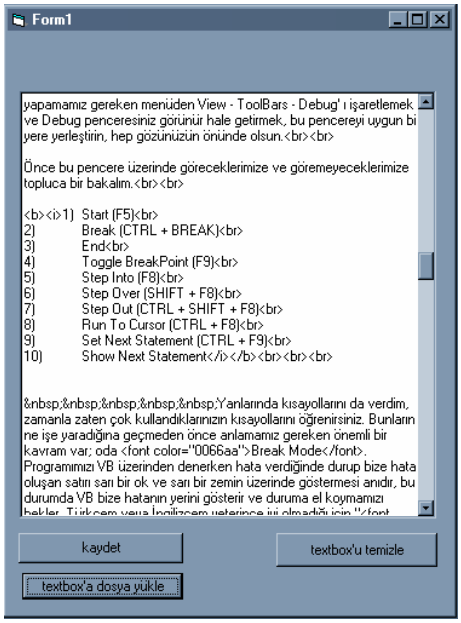
Ancak daha önce bir dosya seçimi yapılmış ise bu satırlar atlanarak ve metin kutusunun “SaveFile” özelliği kullanılarak kayıt işlemi yapılır.

5. Aşağıda görülen kodları “Command3” nesnesinin “Click” olayına yazınız.

```
Private Sub Command3_Click()
    RichTextBox1.Text = ""
End Sub
```

Bu buton kullanıldığında metin kutusuna boş atama yapılarak temizlenmesi sağlanır.

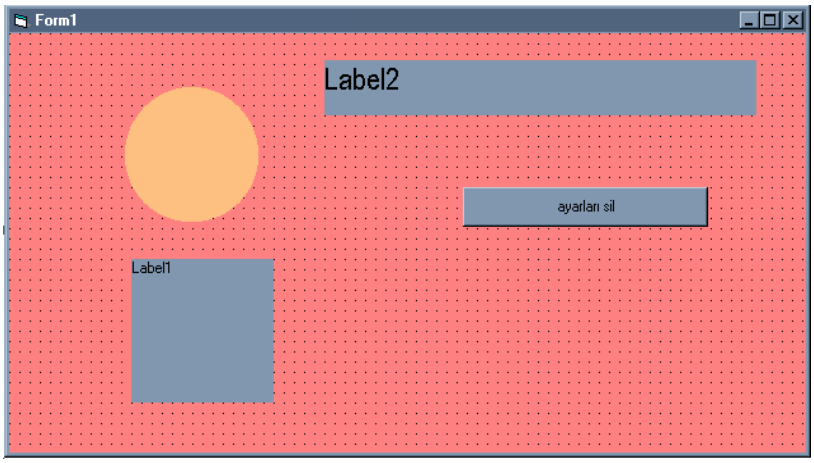
6. Program çalıştırılarak herhangi bir dosya yüklendiğinde, yüklenen dosya şekil 7.2 deki gibi metin kutusunda görünecektir.



Şekil 7.2

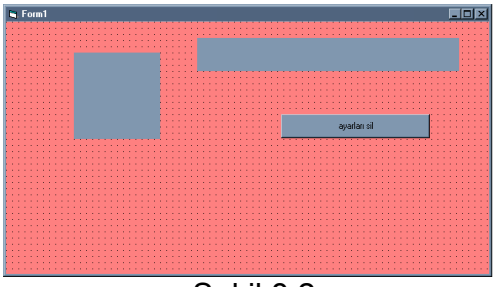
UYGULAMA 8

Programların deneme sürümlerinin, sınırlı gün uygulaması ile birçok defa karşılaşmışsınızdır. Program kurulduktan belirli gün sonra, sürenin tükendiğini belirterek çalışmaz. Bu işlem gibi bütün programlar ayarlarını “Registry” ‘e kayıt ederler. “Registry” Windows’un bu kullanım amacı için sunduğu bir kayıt düzenleyicisidir. Bu uygulama ile kayıt düzenleyicisine veri kaydı yapacak ve daha sonra bu veriyi kullanacağız.

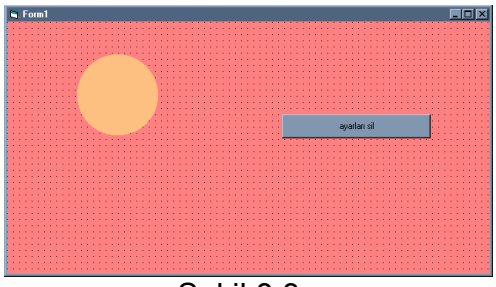


Şekil 8.1

1. Şekil 8.1 de görülen ve üzerinde bir adet “Shape”, bir adet “CommandButton” ve iki adet “Label” kontrolü olan formu oluşturun.



Şekil 8.2



Şekil 8.3

2. “Label1” nesnesini şekil 8.2 de görüldüğü gibi “Shape” nesnesinin üzerine getirin. Daha sonra “Label1” ve “Label2” nesnelerinin “Backstyle” özelliklerini “Transparent” olarak değiştirin.(Şekil 8.3)

3. Formun “Load” olayına aşağıdaki kodları yazın.

```
Private Sub Form_Load()
Dim giriş, sayı
giriş = GetSetting("kayıt", "bizimayar", "giriş")
If giriş = "" Then
   SaveSetting "kayıt", "bizimayar", "giriş", Date
   Label2.Caption = "bu programı ilk defa çalıştırdınız..."
   SaveSetting "enesfurkan", "bizimayar", "sayı", "1"
Else
   If (Date - CDate(giriş)) > 3 Then
   MsgBox "süreniz doldu"
```

```
    End
    End If
    sayı = GetSetting("kayıt", "bizimayar", "sayı")
    Label2.Caption = "programı " & sayı & " defa çalıştırdınız..."
    SaveSetting "kayıt", "bizimayar", "sayı", sayı + 1
    If CDate(giriş) > Date Then
    MsgBox "tarihi değiştirmişsin "
    End
    End If
  End If
End Sub
```

Yukarıdaki kodların açıklamasına geçmeden önce kayıt düzenleyicisi hakkında bazı bilgilerin verilmesi gerekmektedir. Öncelikle “Başlat menüsü” nden “Çalıştır” ı seçerek “Regedit” ‘i çalıştırdığınızda kayıt düzenleyicisi ekrana gelir. Ancak daha önce kayıt düzenleyicisini kullanmadıysanız burada işlem yapmaktan kaçının. Çünkü herhangi bir veriyi silmeniz durumun ilgili program yada windows çalışmayabilir. Genel bir anlatımla kayıt düzenleyicisine yaptığınız her kayıt işlemi için adres, veri adı ve veri belirtilmelidir. Visual Basic ile yaptığımız kayıtların adresi “HKEY_CURRENT_USER\software\vb and VBA settings” olmaktadır. Son “vb and VBA settings” klasörünün içine kendi klasör yada klasörlerimizi oluşturarak içlerine veri kaydı yaparız.

İlk satırda “Giriş” ve “Sayı” değişkenleri tanımlanmıştır.

İkinci satırda ise “GetSetting” komutu kullanılmıştır. Bu komut yardımıyla kayıt düzenleyicisinden veri okunur. Bu satırda “HKEY_CURRENT_USER\software\vb and VBA settings\kayıt\bizimayar\” adresinden giriş ismine sahip veri, giriş değişkenine atanmıştır.

Giriş değişkenine atama yapıldıktan sonra iki durum söz konusudur. Ya program ilk defa çalıştırılmıştır ve giriş değişkeni herhangi bir değer almamıştır yada daha önce çalıştırılmıştır ve giriş değişkenine atama yapılmıştır.

Giriş değişkenine herhangi bir atama yapılmamış olması durumu için “If giriş = "" Then” şeklindeki ilk “If “ bloğu yazılmıştır.

Bu şartın sağlanmış olması, programın ilk defa çalıştırılmış olması anlamına geldiğinden “SaveSetting "kayıt", "bizimayar", "giriş", Date” komut satırı ile aynı adrese ve giriş isimli veriye o günün tarihi kayıt edilir. Böylece program bir dahaki çalıştırılmasında giriş değişkeni ilk çalıştırılma tarihinin değerini alarak bu satırlar atlanacak.

Daha sonra Label2 nesnesine "Bu programı ilk defa çalıştırdınız..." metni atanır.

Bunun ardından programın kaç defa çalıştırıldığını öğrenebilmek için kullanacağımız “Sayı” isimli veri kaydı “SaveSetting "kayıt", "bizimayar", "sayı", "1"” komutu satırı ile ilgili adrese “sayı” ismiyle “1” olarak kaydedilir.

Eğer giriş değişkenine bir atama yapılmışsa ki bu programın daha önce çalıştırılmış olduğu anlamına gelir, ilk çalıştırılma tarihinin yani giriş değişkeninin değerinin üzerinden, belirlemiş olduğumuz üç gün sınır süresinin geçip geçmediğini kontrol edeceğiz.

Bunun için yine “If” bloğundan yararlanarak “If (Date - CDate(giriş)) > 3 “ şartını sunacağız. Programın çalıştırıldığı tarihten, kayıt düzenleyicisinden elde ettiğimiz ve giriş değişkenine atadığımız ilk çalıştırılma tarihini “cdate” komutu yardımıyla sayısal olarak çıkararak üç günü geçip geçmediğini kontrol edeceğiz.

Şekil 8.4

Eğer bu şart sağlanmışsa “MsgBox "süreniz doldu" “ komut satırı ile şekil 8.4 de ki gibi mesaj kutusu ekrana getirilecek ve tamam butonuna tıklanmasıyla, “End” komutunun yardımıyla program sona erdirilecektir. Böylece programımız ilk çalıştırılma tarihinden üç gün geçtikten sonra bir daha çalışmayacaktır.

Programın daha önce çalıştırılmış olması şartı hala devam etmektedir. Bu durumda kullanıcıya programı kaç defa çalıştırdığını belirtmek için “Sayı” isimli veri kaydı, “sayı = GetSetting("enesfurkan", "bizimayar", "sayı")” komut satırı ile sayı değişkenine atanır.

Daha sonra ise “Label2.Caption = "Programı " & sayı & " defa çalıştırdınız..." satırı ile “Label2” nesnesinin metni “Programı 3 defa çalıştırdınız...” şeklinde değiştirilir.

Program bir kez daha çalıştırıldığına göre sayı isimli veri kaydının bir arttırılması gerekir. Bunun için ise “SaveSetting "kayıt", "bizimayar", "sayı", sayı + 1” kodu yazılmıştır.

Şekil 8.5

Sizin de tahmin edebileceğiniz gibi kullanıcı, giriş ve sayı verilerimizi kayıt düzenleyicisinden değiştirebilir. Ve bu sık karşılaşılan bir durumdur. Bunu yapan kullanıcılar, her gün bu işlemle uğraşmamak için bizim giriş adını verdiğimiz ve ilk çalıştırma tarihinin kaydedildiği veriyi ileri tarih olarak değiştirirler. Uygulamamıza bunun da güvenlik önlemini “If CDate(giriş) > Date” satırı ile ekliyoruz. Eğer kayıt düzenleyicisinden aldığımız giriş değeri, günün tarihinden büyükse şekil 8.5 deki mesaj kutusunu “MsgBox "tarihi değiştirmişsin " komut satırı ile ekrana getiririz. Tamam butonuna tıklanması ile program “End” komutu etkisinde sona erer.

4. Aşağıda görülen kodları “Command1” nesnesinin “Click” olayına yazınız.

```
Private Sub Command1_Click()
        DeleteSetting "kayıt", "bizimayar", "giriş"
        DeleteSetting "kayıt", "bizimayar", "sayı"
End Sub
```

“Command1” butonunun kullanım amacı kayıt düzenleyicisine yapmış olduğumuz giriş ve sayı isimli veri kayıtlarını silmektir. Bunun için ise yukarıda görüldüğ gibi “DeleteSetting” komutu kullanılır. Bu komutun kullanımı, “Getsetting” ve “Savesetting” komutlarının kullanımıyla aynıdır.

6. Formun “MouseMove” olayına aşağıdaki komutları yazın.

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Shape1.FillColor = RGB(233, 323, 123)
End Sub
```

Bu satırla yaptığımız işlemi önceki uygulamalarımızda da gerçekleştirdik. Farenin form üzerinde gezinirken, “Shape1” nesnemizin rengini belirledik.

7. “Label1” nesnesinin “MouseMove” olayına aşağıdaki komutları yazın.

```
Private Sub Label1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Shape1.FillColor = RGB(123, 12, 150)
End Sub
```

Bu satırla ise farenin “Label1” nesnesi üzerinde gezinirken “Shape1” nesnemizin kazanacağı dolgu rengini belirledik.

8. “Label1” nesnesinin “Click” olayına aşağıdaki komutları yazın.

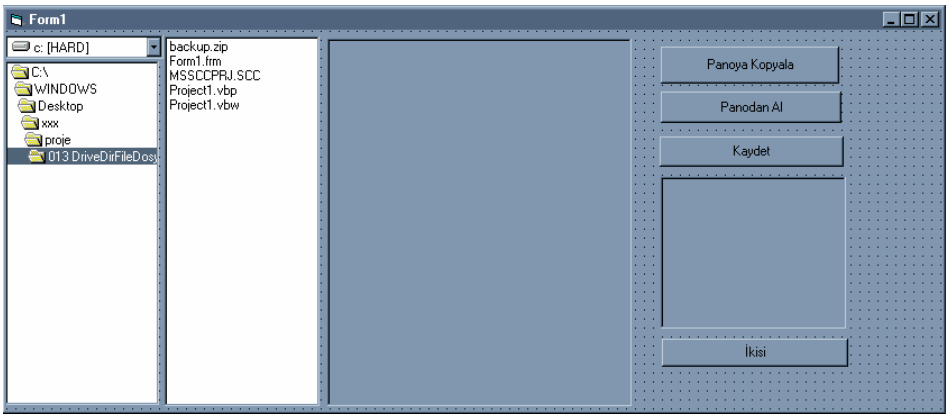
```
Private Sub Label1_Click()
Shape1.FillColor = RGB(123, 260, 150)
MsgBox "İlk çalıştırma tarihi: " & GetSetting("kayıt", "bizimayar", "giriş")
End Sub
```

“Label1” nesnesine tıklandığında ilk satır yardımıyla “Shape1” nesnesinin dolgu rengi değişecek. Daha sonra ise mesaj kutusu yardımıyla, yine “GetSetting” komutu yardımıyla elde ettiğimiz giriş yani ilk çalıştırma tarihi ekranda şekil 8.6 daki gibi gösterilir.

Şekil 8.6

UYGULAMA 9

Microsoft Word programında sık olarak karşımıza çıkan pano kullanımını, bu sefer kendi programımızda gerçekleştireceğiz. Resim dosyasını panoya kaydederek, daha sonra bunu kullanacağız.



Şekil 9.1

1. Şekil 9.1 de görülen ve üzerinde bir adet "Drive List Box", bir adet "Dir List Box", bir adet "File List Box", dört adet "Command Button" ve iki adet "Picture Box" kontrolü olan formu oluşturun.

2. Kontroller için aşağıdaki değişiklikleri yapın.

Nesne	İsim	Caption
Command1	cmdPanoKopya	Panoya Kopyala
Command2	cmdPanodanAl	Panodan Al
Command3	cmdKaydet	Kaydet
Command4	cmdİkisi	İkisi

3. "Drive1" nesnesinin "Change" olayına aşağıdaki kodları yazın.

```
Private Sub Drive1_Change()
    Dir1.Path = Drive1.Drive
End Sub
```

"Change" olayındaki komutlar, "Drive1" nesnesinde herhangi bir değişiklik olduğunda gerçekleşecektir. Bu değişiklik gerçekleştiğinde "Dir1" nesnesinin yolu, "Dir1.Path = Drive1.Drive" komut satırı ile "Drive1" de yeni seçilen yola eşlenecektir.

4. "Dir1" nesnesinin "Change" olayına aşağıdaki kodları yazın.

```
Private Sub Dir1_Change()
    File1.Path = Dir1.Path
    ChDir (File1.Path)
End Sub
```

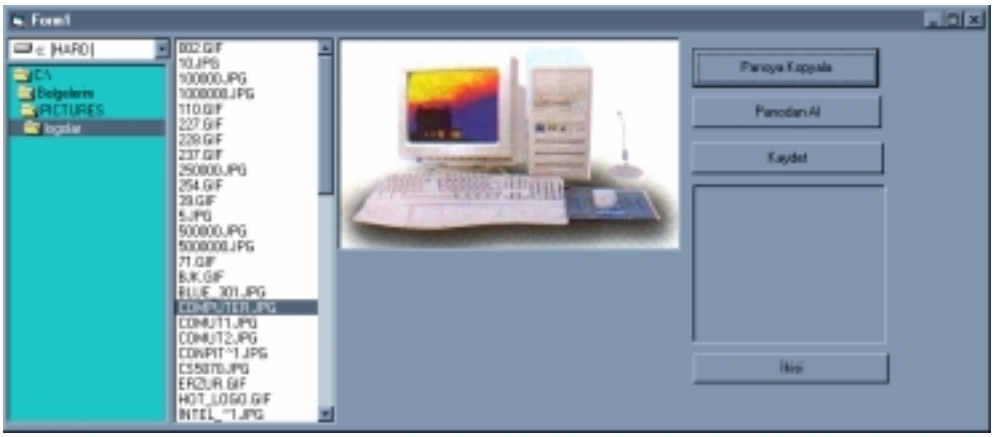
Yukarıda olduğu gibi, "Dir1" nesnesinde herhangi bir değişiklik gerçekleştiğinde bu sefer "File1" nesnesinin yolu, "Dir1" nesnesinin yoluna

eşlenecektir. “ChDir (File1.Path)” komutu yardımıyla “File1” nesnesinin bulunduğu yolunun klasörü değiştirilir.

5. Formun “Load” olayına aşağıdaki kodları yazın.

```
Private Sub Form_Load()
    Dir1.BackColor = RGB(20, 200, 200)
End Sub
```

Formumuz yüklendiğinde, “Dir1” nesnemizin zemin rengini, bu komut satırı ile değiştirmiş olacağız.



Şekil 9.2

6. “File1” nesnesinin click olayına aşağıdaki kodları yazın.

```
Private Sub File1_Click()
    On Error GoTo rhatası
    Picture1.Picture = LoadPicture(File1.FileName)
    Exit Sub
rhatası:
MsgBox "hatalı: resim dosyası seçiniz!", vbExclamation, "Resim Dosyası Seçimi"
End Sub
```

“File1” nesnesinde görülen dosyalardan herhangi birine tıklandığında, işlemler yaptırılmadan önce “On Error GoTo rhatası” komut satırı kullanılmıştır. Çünkü kullanıcının seçtiği dosya; resim dosyası olmayabilir yada uygun resim dosyası olmayabilir. Bu nedenle hata durumuyla karşılaşıldığında program “rHatası” satırına yönlendirilecek ve daha önceki uygulamalarımızda olduğu gibi mesaj kutusuyla bu hata bildirilerek, olay sona erdirilecek.

İkinci satırda ise yine daha önceki uygulamalarda olduğu “Picture1” nesnesine resim yükleniyor. Ancak burada dosya olarak “File1” den seçilen dosya yani “File1.FileName” kullanılır.(Şekil 9.2)

“Exit Sub” kullanılarak olaydan çıkılır ve “rHatası” satırının çalışması engellenir.

7. “cmdPanoKopya” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub cmdPanoKopya_Click()
    Clipboard.Clear
```

```
    Clipboard.SetData Picture1.Picture
End Sub
```

İlk satırda bizim pano olarak adlandırdığımız “Clipboard” ın “Clear” özelliği kullanılarak, pano temizlenir. Bunun ardından “SetData” özelliği ile, “File1” nesnesinden seçilerek “Picture1” nesnesine yüklenen resim, panoya kopyalanır.

8. “cmdKaydet” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub cmdKaydet_Click()
    SavePicture Picture1.Picture, InputBox("Dosya Adı")
End Sub
```

Panoya “Picture1” nesnesinden atadığımız resmi, “Picture1” nesnesinin “SavePicture” özelliği ile kaydedeceğiz. Dosya adını ise, ekrana gelecek olan “InputBox” dan alacaktır.

9. Formun nesnesinin “Resize” olayına aşağıdaki kodları yazın.

```
Private Sub Form_Resize()
    Picture1.Left = (Form1.ScaleWidth - Picture1.Width) / 2
    Picture1.Top = (Form1.ScaleHeight - Picture1.Height) / 2
End Sub
```

Formun boyutları değiştirildiğinde, “Picture1” nesnesinin, formun ortasında yer alması için, daha önceki uygulamalarımızda olduğu gibi “left” ve “top” koordinatlarını atadık.

10. “cmdKaydet” nesnesinin “Click” olayına aşağıdaki kodları yazın.

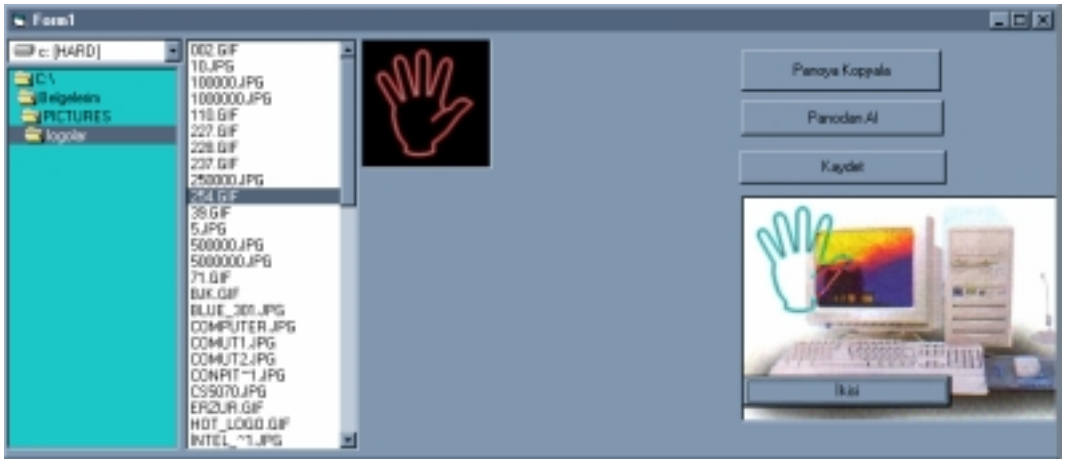
```
Private Sub cmdPanodanAl_Click()
    Picture1.Picture = Clipboard.GetData
End Sub
```

Yedinci işlem basamağında, “Picture1” nesnesindeki resmi, panoya atamıştık. Adı geçen nesnenin üzerindeki resim değiştikten sonra, panodaki resmi bu nesneye yüklemek için yukarıdaki satırı ekledik. Resim panodan “Getdata” özelliği ile geri alınmıştır.

11. “cmdİkisi” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub cmdİkisi_Click()
    Picture2.Picture = Clipboard.GetData
    Picture2.PaintPicture Picture1.Picture, 0, 0, Picture1.ScaleWidth,
    Picture1.ScaleHeight, 0, 0, Picture1.ScaleWidth, Picture1.ScaleHeight,
    &H660046
End Sub
```

Bu işlem basamağına kadar, “PictureBox” nesnelerine resimleri ya atama yada yükleme ile aktardık. Bu işlemlerle “PictureBox” nesnesi sadece atanan resmi gösterdi. Şimdi ise birden fazla resmi aynı nesne üzerinde göstereceğiz.



Şekil 9.3

İlk satır ile, “Picture2” nesnesine panoda bulunan resmi atıyoruz. İkinci satır da ise yine “Picture2” nesnesine resim atıyoruz. Ancak bu ikinci durumda “PaintPicture” özelliği kullanarak alttaki resmin üzerine ikinci resmi boyatıyoruz. Böylece iki resim şekil 9.3 deki gibi üst üste görünecektir.

“PaintPicture” özelliğinin sözcük dizilimi şöyledir:

Picture, x_1, y_1, $width_1$, $height_1$, x_2, y_2, $width_2$, $height_2$, opcode

“Picture” ile üste eklenen resim, “x” ve “y” ile resimlerin üst köşe koordinatları, “width” ile genişlik, “height” ile yükleklikler belirtilir. Bu özelliklerin “1” indisli olanları resmin yerleştirilmesi için kullanılır. “2” indisli olanlar ise gösterilecek kısmı için kullanılır.

11. “Picture1” nesnesinin “Click” olayına aşağıdaki kodları yazın

```
Private Sub Picture1_Click()
    Picture2.PaintPicture Picture1.Picture, 0, 0, Picture1.ScaleWidth / 2,
    Picture1.ScaleHeight / 2, 0, 0, Picture1.ScaleWidth,
    Picture1.ScaleHeight, &HCC0020
End Sub
```

“Picture1” nesnesine tıklandığında, yukarıda olduğu gibi “Picture2” nesnesine, “Picture1” nesnesindeki resim boyanacaktır. Ancak yukarıdan farklı olarak “$Width_1$” ve “$Height_1$” değerleri, resmin değerlerinin ikiye bölünerek, ½ oranında küçük gösterilecektir. Ayrıca “opcode” özelliğine “&HCC0020” değeri atanarak orijinal rekleri ile boyanması sağlanmıştır.

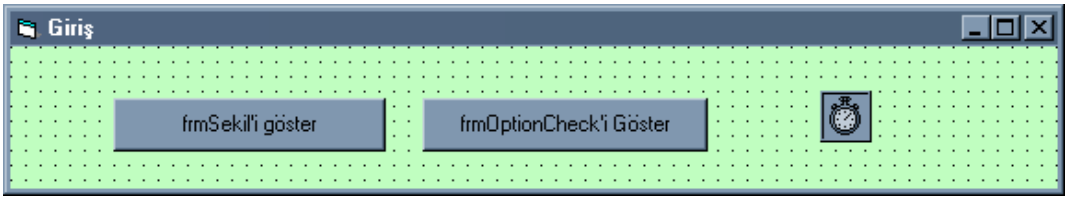
12. “Picture1” nesnesinin “DblClick” olayına aşağıdaki kodları yazın

```
Private Sub Picture1_DblClick()
    Picture2.PaintPicture Picture1.Picture, 0, 0, Picture1.ScaleWidth / 2,
    Picture1.ScaleHeight / 2, 0, 0, Picture1.ScaleWidth,
    Picture1.ScaleHeight, &H550009
End Sub
```

“Picture1” nesnesine çift tıklandığında, yukarıda olduğu gibi “Picture2” nesnesine, “Picture1” nesnesindeki resim boyanacaktır. Ancak yukarıdan farklı olarak “opcode” özelliğine “&H550009” değeri atanarak orijinal reklerinin terslerinin tersleri ile boyanması sağlanmıştır.

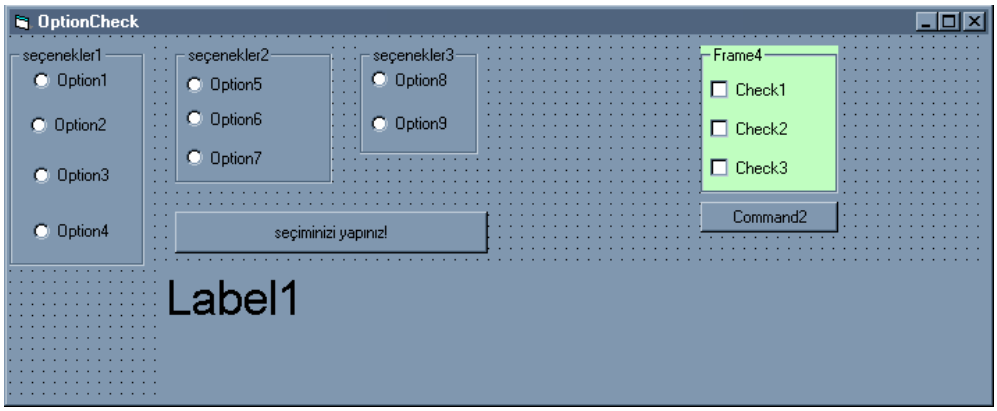
UYGULAMA 10

Diğer uygulamalarımızda kullanmadığımız ve Windows ‘un kullanıcıya büyük kolaylık sağlayan kontrollerine bu uygulamada yer veriyoruz. “Frame”, “CheckBox”, “OptionButton”, “ScrollBar” kontrollerinin yanı sıra “Timer” kontrolü de uygulamada yer alacak. Ayrıca bu uygulamada birden fazla formun kullanımı da gerçekleşecek.



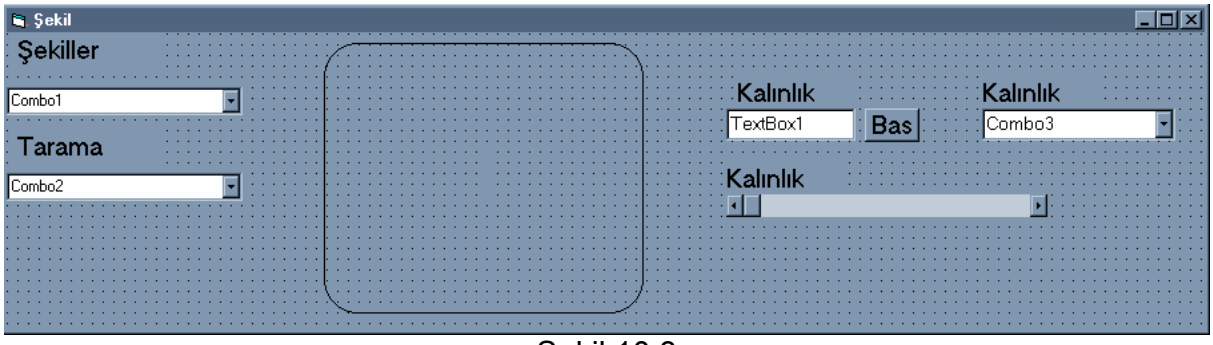
Şekil 10.1

1. Şekil 10.1 de görülen formu oluşturun. “Form1” ismini “frmGiris” olarak değiştirin. Ayrıca “Timer1” nesnesinin “Interval” özelliğini “30” yapın.



Şekil 10.2

2. Şekil 10.2 de görülen formu oluşturun. Ancak şekilde de görüldüğü gibi “OptionButton” nesneleri seçeneklerin yani “Frame” nesnelerinin içinde yer almaktadır. Bunun yapılması için öncelikle “Frame” kontrollerini forma ekleyin, ardından “Optionbutton” nesnelerini bu “Frame” nesnelerinin içinde oluşturun. Formun ismini “frmOptionCheck” olarak değiştirin.



Şekil 10.3

3. Şekil 10.3 de görülen formu oluşturun. “TextBox” kontrolünün “Text” özelliğini boşaltın. Formun ismini “frmSekil” olarak değiştirin.

4. “frmGiris” isimli formun “Load” olayına aşağıdaki kodları yazın.

```
Private Sub Form_Load()
    frmGiris.Height = 20
    frmGiris.Width = 20
End Sub
```

Formumuz ilk açıldığında genişlik ve yükseklik değerleri “20” olacaktır.

5. “Timer1” isimli nesnenin “Timer” olayına aşağıdaki kodları yazın.

```
Private Sub Timer1_Timer()
    Static hacim As Integer
    hacim = hacim + 1
    frmGiris.Height = 90 * hacim
    frmGiris.Width = 600 * hacim
    If hacim = 20 Then Timer1.Interval = 0
End Sub
```

Kodların açıklanmasına başlamadan önce “Timer” kontrolü hakkında bazı bilgilerin verilmesi gerekmektedir. Bu kontrol zamanlayıcı olarak çalışır. Örneğin kontrolün “Interval” özelliğine “1000” değeri atarsak, her “1” sanide bir, “Timer1” nesnesinin “Timer” olayı gerçekleşecek ve yukarıdaki satırlar çalışacaktır. Anlaşıldığı gibi Interval özelliğine atanan değerler, milisaniye birimindedir ve “1000” milisaniye, “1” saniyedir. Biz bu kontrolün “Interval” özelliğini ilk işlem basamağında “30” olarak belirledik.

İlk satırda “Hacim” değişkeni tanımlanmıştır. Ancak alışagelmiş şekilde “Dim” terimi ile değil de “Static” terimiyle tanımladık. Çünkü bir değişkenin “Static” terimiyle tanımlanması, bu değişkenin projenin tüm çalışma süresi boyunca değerini korumasını sağlar. Eğer “Dim” terimi kullanmış olsaydık, “Hacim” değişkeni yukarıdaki satırların sona ermesiyle yani yordam duduğunda tekrar “0” değerini alacaktı.

Zamanlayıcının her tekrarında “Hacim” değişkenin değerinin “1” artması “hacim = hacim + 1” satırıyla sağlanmıştır.

Daha sonraki iki satır ile formumuzun büyümesi sağlanır. Yükseklik “90” ın katlarıyla, genişlik “600” katlarıyla artacaktır.

Bu artış “Hacim” değişkeninin “20” olması ile sonlandırılacaktır. Değişken “20” olduğunda “Timer1” in “Interval” özelliği sıfır olacak ve bu yordam bir daha çalışmayacaktır.

Böylelikle program çalıştırıldığında formun giderek büyüdüğünü ve bir süre sonra büyümenin durduğunu göreceksiniz.

6. “Command1” isimli nesnenin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Command1_Click()
    frmSekil.Show
    frmSekil.Top = frmGiris.Top + frmGiris.Height
    frmSekil.Left = frmGiris.Left
End Sub
```

Amacımız butona tıklandığında “frmSekil” formunun ekrana getirilmesidir. Bu nedenle “frmSekil.Show” satırı kullanılmıştır.

Diğer bir isteğimiz ise ekrana getirilen “frmSekil” formunun üstünün “frmGiris” formunun altına dayalı olması ve her ikisininde sol kenarlarının aynı hizada olmasıdır.

İkinci satırda “frmSekil” formunun üst noktası, “frmGiris” formunun üst başlangıcı ile yüksekliğinin toplamına eşitlenerek, şartımız sağlanmıştır.

Son satırla “frmSekil” formunun sol koordinatı, “frmGiris” formunun sol koordinatına eşitlenmiştir.

7. “Command2” isimli nesnenin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Command2_Click()
    frmOptionCheck.Show
    frmOptionCheck.Top = frmSekil.Top + frmSekil.Height
    frmOptionCheck.Left = frmGiris.Left
End Sub
```

Bu satırlarla yapılan işlem bir önceki işlem basamağı ile aynıdır.

8. “frmSekil” formunun “Load” olayına aşağıdaki kodları yazın.

```
Private Sub Form_Load()
    Combo1.AddItem "Dikdörtgen"
    Combo1.AddItem "Kare"
    Combo1.AddItem "Elips"
    Combo1.AddItem "Çember"
    Combo1.AddItem "Oval dikdörtgen"
    Combo1.AddItem "Oval kare"
    Combo1.ListIndex = 0
    Combo2.AddItem "Tam dolu"
    Combo2.AddItem "Şeffaf"
    Combo2.AddItem "Yatay çizgili"
    Combo2.AddItem "Dikey çizgili"
    Combo2.AddItem "Sola eğik"
    Combo2.AddItem "Sağa eğik"
    Combo2.AddItem "Kareli"
    Combo2.AddItem "Çapraz"
    Combo2.ListIndex = 0
    Combo3.AddItem "Kalınlık 1"
    Combo3.AddItem "Kalınlık 2"
    Combo3.AddItem "Kalınlık 5"
    Combo3.AddItem "Kalınlık 20"
    Combo3.ListIndex = 0
    Shape1.BorderStyle = 5
    Shape1.BorderWidth = 1
    Shape1.FillStyle = 1
    HScroll1.Max = 50
    HScroll1.Min = 1
End Sub
```

Bu satırlarla ilk olarak “Combo” nesnelerini dolduruyoruz. “Combo” nesnelerini “AddItem” özelliği yardımıyla, listeye sözcükler ekliyoruz.

Bu satırlarda kullanılan “Combo” nesnelerinin “ListIndex” özelliğinin amacı, ilk durumda kontrolde seçili olacak sözcüğün belirlenmesidir. Bu belirlemeyi sözcüğü listedeki numarası yardımıyla yaparız. Örneğin bu uygulamada bu özelliğe “0” değeri verilerek listenin ilk sırasındaki sözcük seçili olacaktır.

“Shape1 “ nesnesinin özelliklerinin belirtildiği satırların işlevlerine daha önceki uygulamalarımızda yer vermiştik.

“HScroll1.Max = 50” ve “HScroll1.Min = 1” satırları ile nesnenin alacağı en fazla ve en az değerler girilmiştir.

9. “frmSekil” formundaki “Combo1” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Combo1_Click()
If Combo1.Text = "Dikdörtgen" Then
   Shape1.Shape = 0
   ElseIf Combo1.Text = "Kare" Then Shape1.Shape = 1
         Label3.Caption = "Bu bir karedir!"
   ElseIf Combo1.Text = "Elips" Then Shape1.Shape = 2
       Label3.Caption = "Bu bir elipsdir!"
   ElseIf Combo1.Text = "Çember" Then Shape1.Shape = 3
   Label3.Caption = "Bu bir çemberdir!"
   ElseIf Combo1.Text = "Oval dikdörtgen" Then Shape1.Shape = 4
   Label3.Caption = "Bu bir oval dikdörtgendir!"
   ElseIf Combo1.Text = "Oval kare" Then Shape1.Shape = 5
   Label3.Caption = "Bu bir oval karedir!"
   End If
End Sub
```

Kullanıcı “Combo1” nesnesinin listesinden herhangi bir sözcük seçtiğinde, “Combo1.Text” bu sözcüğü tanımlayacak ve yukarıdaki kodlar çalışacaktır.

Kullanılan “If” blokları ile seçilen sözcüğün hangisine eşit olduğu yani hangisi olduğu tek tek kontrol edilir.

Eşleme gerçekleştiğinde, eşlemenin gerçekleştiği blokta bulunan komutlar işlevlerini yerine getirecektir. Öncelikle seçilen sözcüğe göre “Shape1” nesnesinin “Shape” özelliği kullanılarak biçimi değiştirilecek, daha sonra “Label3” nesnesinin metni değiştirilecektir.

10. “Combo2” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Combo2_Click()
If Combo2.Text = "Tam dolu" Then
   Shape1.FillStyle = 0
   ElseIf Combo2.Text = "Şeffaf" Then Shape1.FillStyle = 1
   ElseIf Combo2.Text = "Yatay çizgili" Then Shape1.FillStyle = 2
   ElseIf Combo2.Text = "Dikey çizgili" Then Shape1.FillStyle = 3
   ElseIf Combo2.Text = "Sola eğik" Then Shape1.FillStyle = 4
   ElseIf Combo2.Text = "Sağa eğik" Then Shape1.FillStyle = 5
   ElseIf Combo2.Text = "Kareli" Then Shape1.FillStyle = 6
   Else: Shape1.FillStyle = vbDiagonalCross
   End If
End Sub
```

Bu işlem basamağında yapılanlar bir önceki ile aynıdır. Bu sefer “Combo2” nesnesinden seçilen sözcüğe göre “Shape1” nesnesinin dolgu stili değiştirilmektedir.

11. “Combo3” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Combo3_Click()
   If Combo3.Text = "Kalınlık 1" Then
   Shape1.BorderWidth = 1
   ElseIf Combo3.Text = "Kalınlık 2" Then Shape1.BorderWidth = 2
   ElseIf Combo3.Text = "Kalınlık 5" Then Shape1.BorderWidth = 5
   ElseIf Combo3.Text = "Kalınlık 20" Then Shape1.BorderWidth = 20
   Else
   Shape1.BorderWidth = 35
   End If
End Sub
```

Bu basmakta ise “Combo3” nesnesinden yapılan seçime göre “Shape1” nesnesinin çerçeve kalınlığı değiştirilmektedir.

12. “Command1” nesnesini “Click” olayına aşağıdaki kodları yazınız.

```
Private Sub Command1_Click()
      If Text1.Text > 50 Then
      MsgBox "50'den küçük  sayı yazınız."
      End If
      If Text1.Text <= 50 Then
      Shape1.BorderWidth = Text1.Text
      End If
End Sub
```

Bir önceki işlem basamağında “Shape1” nesnesinin çerçeve kalınlığını “ComboBox” nesnesi yardımıyla değiştirmiştik. Bu basmakta ise aynı değişikliği “Text1” ve “Command1” nesnelerini kullanarak yapacağız.

Öncelikle “Text1” nesnesine girilen değerin “50” den az olmasını sağlayacağız. Bunun için “If Text1.Text > 50” şartını kullandık. Eğer bu şart sağlanırsa, kullanıcıya mesaj kutusu yardımıyla girilen değeri düzeltmesi gerektiği bildirilecek.

“If Text1.Text <= 50” şartı yardımıyla “Shape1” nesnesinin, çerçeve kalınlık değişimi sadece girilen değerin “50” veya altına olması ile gerçekleşecek.

13. ”Hscroll1” nesnesinin “Change” olayına aşağıdaki kodları yazın.

```
Private Sub HScroll1_Change()
Shape1.BorderWidth = HScroll1.Value
Label3.Caption = HScroll1.Value
End Sub
```

“Shape1” in çerçeve kalınlık değişimini bu seferde “Hscroll1” nesnesi ile gerçekleştireceğiz. “Hscroll1” nesnesinin değeri değiştirildiğinde bu satırlar işlev kazanacaktır. Bu nesnenin alacağı en az ve en yüksek değerleri daha önce “0-50” olarak değiştirmiştik. Nesnenin aldığı değer “Value” özelliği yardımıyla “Shape1” in çerçeve kalınlığı olarak atanır ve bu değer “Label3” nesnesi yardımıyla ekrana getirilir.

14. ”Hscroll1” nesnesinin “Scroll” olayına aşağıdaki kodları yazın.

```
Private Sub HScroll1_Scroll()
      Shape1.BorderWidth = HScroll1.Value
```

```
    Label3.Caption = HScroll1.Value
End Sub
```

Görüldüğü üzere yazacağımız bu kodlar bir önceki basmaktakilerle aynıdır. Ancak bu sefer kodların çalışacağı olay farklıdır. “Hscroll1” nesnesinin “Scroll” olayı, farenin sol tuşu basılı tutularak “Hscroll” un konum belirticisinin taşınması ile gerçekleşir. Yani, konum belirticinin taşıma işlemi bitmeden çerçeve kalınlığı değişecektir.

15. “Text1” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Text1_Click()
    If IsNumeric(Text1.Text) Then
    Text1.SelStart = 0
    Text1.SelLength = Len(Text1.Text)
    Else
    MsgBox "lütfen sayısal bir değer giriniz!"
    End If
End Sub
```

Windows’ta kullandığımız çoğu programda, programların “TextBox” kontrollerine tıkladığımızda, kutu içerisindeki metinin seçili duruma geçtiğini görürüz. Bizde uygulamamızda kullandığımız “TextBox” için aynı özelliği ekleyeceğiz.

Öncelikle bu kutuya girilen değeri çerçeve kalınlığı olarak kullandığımız için, bu değerin sayısal olması gerekir. Bunun kontrolü için “IsNumeric” terimini kullanacağız. Bu terimde parantez içinde yer alan değişken sayısal ise, terim “True” yani doğru değerini alacaktır. Bunu “If” şartında kullanarak, sayısal değer şartı sağlanmışsa “Text1.SelStart = 0” ve “Text1.SelLength = Len(Text1.Text)” satırları işlev kazanacaktır. “SelStart” terimi seçimin yapılacağı ilk karakteri belirler. “SelLenght” ise seçimin uzunluğunu tanımlar. “Len” terimi ise parantez içindeki metinin karakter sayısını belirtir.

Böylece “Text1” e tıklandığında, ilk karakterden itibaren, toplam karakter sayısı kadar yani hepsi seçili duruma geçecek.

Eğer “If IsNumeric(Text1.Text)” şartı sağlanmamış ise, mesaj kutusu yardımıyla kullanıcıya sayısal değer girmesi bildirilecek.

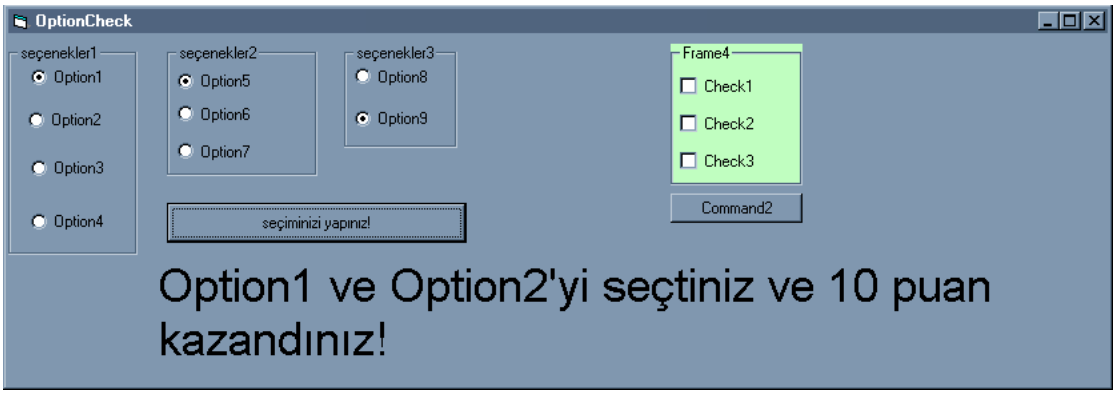
16. “Text1” nesnesinin “KeyPress” olayına aşağıdaki kodları yazın.

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
    Text1.SelStart = 0
    Text1.SelLength = Len(Text1.Text)
    Command1_Click
    End If
End Sub
```

Windows ortamında ki programlarda “Textbox” lara girilen değerlerin ardından “Enter” tuşuna basılması ile işlem gerçekleştirilir. Bu özelliği de “TextBox” a yukarıdaki satırlar yardımıyla ekleyeceğiz.

“Text1” için “KeyPress” olayı, “Text1” kullanımda iken klavyenin herhangi bir tuşuna basılmasıdır. Bu yordamda otomatik olarak “KeyAscii” değişkeni tanımlanır ve bu değişken basılan tuşun “ascii” değerini alır. “Enter” tuşunun “Ascii” kodu “13” tür.

Eğer klavyeden basılan tuş “Enter” ise, “KeyAscii” değişkeni “13” değerini alacak ve “If” bloğumuzun şartı sağlanmış olacaktır. Daha sonraki satırlar ise bir önceki basamakta olduğu gibi yazılı metnin seçilmesini sağlayacaktır.



Şekil 10.4

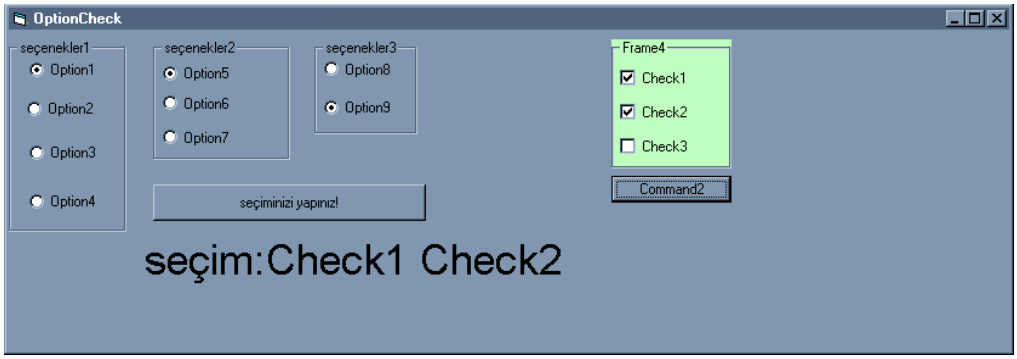
17. Artık “frmOptionCheck” formunun nesnelerine kod yazacağız. Öncelikle bu formun “Command1” nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Command1_Click()
    Label1.Caption = ""
    If Option1.Value = True And Option5.Value = True Then
    Label1.Caption = "Option1 ve Option2'yi seçtiniz ve 10 puan kazandınız!"
    ElseIf Option2.Value = True And Option7.Value And Option8.Value Then
    Label1.Caption = "Option2, Option7 ve Option8'yi seçtiniz ve 20 puan kazandınız!"
    ElseIf Option3.Value = True Or Option9.Value = True Then
    Label1.Caption = "Option3 veya Option9'u seçtiniz ve 50 puan kazandınız! (hangisini seçtiğinizi bilemiyorum!)"
    Else
    Label1.Caption = "ne yazık ki puan alacak bir seçim yapmadınız!"
    End If
End Sub
```

“frmSekil” formunun kod yazımı sona ermiştir. Artık “frmOptionCheck” formu için kod yazacağız. Bu iki form arasında yapılan işlemler arasında herhangi bir bağlantı yoktur.Yani her iki formu, birbirinden bağımsız iki ayrı program gibi düşünebilirsiniz.

“Option” nesnelerinin özelliği, seçili konumda iken “True”, değilken “False” değeri almasıdır.

Kullanıcının seçmiş olduğu “Option” nesnelerine göre puanlama yapacağız. Önceki uygulamalarda olduğu gibi her şartımızı “If” bloklarıyla tek tek kontrol ediyoruz. Örneğin ilk şartımız “Option1” ve “Option5” nesnelerinin seçilmiş olmasıdır. Bu durumda kullanıcı “10” puan kazanmış olacaktır. Kullanıcıya kazandığı puanı, “Label1” nesnesine yazdırarak bildiririz.(Şekil 10.4) Son şart diğerlerinden farklıdır. “Option3.Value = True Or Option9.Value = True” satırında “Or” terimi yani “veya” kullanıldığından “Option3” yada “Option9” dan birisi seçili ise yeterli olacaktır. Eğer şartlarımızdan hiçbiri sağlanmamış ise “Else” terimi yardımıyla kullanıcıya “Label1” nesnesiyle bu durum bildirilir.



Şekil 10.5

18. Command2 nesnesinin “Click” olayına aşağıdaki kodları yazın.

```
Private Sub Command2_Click()
    Dim sonuc As String
    If Check1.Value = 1 Then
    sonuc = "Check1"
    End If
    If Check2.Value = 1 Then
    sonuc = sonuc & " Check2"
    End If
    If Check3.Value = 1 Then
    sonuc = sonuc & "  Check3"
    End If
    Label1.Caption = "seçim:" & sonuc & ""
End Sub
```

İlk satırda “Dim” terimiyle “Sonuc” değişkeni, “String” yani bir text kutusuna yazabileceğimiz tüm karakterleri içerebilen tipte tanımlanmıştır. “Sonuc”, “Dim” ile tanımlandığından, “Command2” ye her tıklandığında yeniden tanımlanacak ve boş (“”) olacaktır.

“CheckBox” kontrolleri seçili durumda iken “Value” “1”, seçili değilken “0” değerlerini alır.

İlk olarak “Check1” nesnesinin seçili olup olmadığı kontrol edilecek ve eğer seçili ise “Sonuç” değişkenine “Check1” metni atanacaktır. Aynı işlemler “Check2” ve “Check3” nesneleri içinde gerçekleşecektir. Ancak dikkat edilmesi gerek konu, “Sonuç” değişkenine her sağlanan şartta yapılan atama; kendisi ve seçili olan “Check” nesnesinin ismidir.

Son olarak ta “Label1” nesnesine seçili kutuların isimleri yazdırılır.

19. "Frame1" nesnesinin "MouseMove" olayına aşağıdaki kodları yazın.

```
Private Sub Frame1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Label1.Caption = ""
    End Sub
```

Böylelikle fare "Frame1" nesnesi üzerinde gezinirken "Label1" nesnesini üzerinde herhangi bir metin olmayacaktır.

20. Bir önceki işlem basamağındaki işlemi "Frame2","Frame3" ve "Frame4" nesneleri için de gerçekleştirin.