

T.C.
MİLLÎ EĞİTİM BAKANLIĞI



MEGEP

(MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMİNİN
GÜÇLENDİRİLMESİ PROJESİ)

ENDÜSTRİYEL OTOMASYON
TEKNOLOJİLERİ

BİLGİSAYARLI KONTROL 4

ANKARA 2008

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşılabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	1
ÖĞRENME FAALİYETİ – 1	3
1. TEMEL GRAFİK İŞLEMLERİ	3
1.1 Çizim Aletleri ile Çalışma	3
1.1.1. Line Bileşeni	4
1.1.2. Shape Bileşeni	7
1.2. Grafik Metotları	10
1.2.1 Line Metodu	11
1.2.2 Pset Metodu	19
1.2.3 CLS Metodu	21
1.2.4 Circle Metodu	21
1.2.5 Fare ile Çalışma	27
1.3. PictureBox	30
1.3.1 Resim Yükleme	31
1.3.2 PictureBox Olayları	32
1.3.4. PictureBox Grafik Metotları	34
1.3.5 Resimleri Saklamak	45
1.4. Çizelgeler	49
1.4.1. Bir Koordinat Sistemi Tanımlamak	49
1.4.2. PictureBox'ta Yazı Yazma	51
1.4.3. Çizgi Çizelgesi (Line Chart)	53
1.4.4. Çubuk Çizelgeleri	56
UYGULAMA FAALİYETİ	60
ÖLÇME VE DEĞERLENDİRME	61
ÖĞRENME FAALİYETİ – 2	62
2. API ile GRAFİK	62
2.1. Apı Text Viewer	63
2.2. Apı Çeşitleri	64
2.3. Piksel Çizimi ve Okuması	66
2.4. Çizgi Çizimi	68
2.5. Dikdörtgen Çizimi	69
2.6. Daire ve Elips Çizimi	78
2.7. Metin Yazmak	82
2.8. BitBlt API Fonksiyonu	83
2.9. Sprite ve Maskeleyme	87
2.10. DirectX	90
UYGULAMA FAALİYETİ	91
ÖLÇME VE DEĞERLENDİRME	92
ÖĞRENME FAALİYETİ – 3	94
3. CANLANDIRMA(ANİMASYON)	94
3.1. Çizgilerle Canlandırma	94
3.2. Image Kontrol ile Basit Animasyon	98
3.3. StretchBlt API Fonksiyonu	110
3.4. Titremeyi Önleme	116

3.5. Shape Kontrolü ile Canlandırma.....	120
3.8. Oyun.....	126
UYGULAMA FAALİYETİ	134
ÖLÇME VE DEĞERLENDİRME	135
MODÜL DEĞERLENDİRME	136
CEVAP ANAHTARLARI	137
KAYNAKÇA	143

AÇIKLAMALAR

KOD	523EO0315
ALAN	Endüstriyel Otomasyon Teknolojileri
DAL/MESLEK	Alan Ortak
MODÜLÜN ADI	Bilgisayarlı Kontrol 4
MODÜLÜN TANIMI	Görsel programda grafik yazım metodlarını anlatan öğretim aracıdır.
SÜRE	40/32
ÖN KOŞUL	Bilgisayarlı Kontrol 3 modülünü almış olmak.
YETERLİK	Grafik programı yazmak.
MODÜLÜN AMACI	Genel Amaç Görsel programlama grafik programı yazabileceksiniz. Amaçlar 1. Görsel programlamada grafiksel araçları doğru olarak kullanabileceksiniz. 2. Görsel programlamada API fonksiyonlarını doğru olarak kullanabileceksiniz. 3. Görsel programlamada animasyon oluşturma işlemini yapabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Ortam: Bilgisayar laboratuvarı Donanım: Bilgisayar, bilgisayar çevre birimleri
ÖLÇME VE DEĞERLENDİRME	Her faaliyetin sonunda ölçme soruları ile öğrenme düzeyinizi ölçeceksiniz. Araştırmalarla grup çalışmaları ve bireysel çalışmalarla öğretmen rehberliğinde ölçme ve değerlendirmeyi gerçekleştirebileceksiniz.

GİRİŞ

Sevgili Öğrenci,

Bu modülde, görsel dünyaya attığınız adımı daha da ileri götürecek, verilerinizi nasıl şekillerle ifade edeceğinizi öğreneceksiniz. Grafik, insan üzerinde çok daha fazla etki bırakan bir güzelliştir. Ele alacağınız her konu, grafiksel olarak bir şekle dönüşecektir.

Bu modül üç öğrenme faaliyetinden oluşmaktadır.

- Temel Grafik İşlemleri
- API Fonksiyonları ile Grafik
- Animasyon (Canlandırma)

Öğrenme faaliyetlerinde konu, teorik bilgiden daha çok örnekler üzerinde anlatılmıştır. Yapılan örneklerde, öğrenilen konuların yanında eksik kalan noktalar da belirlenerek bir sonraki örnekle yeni bilgilerin öğrenilmesi amaçlanmıştır.

Örnekler, çoğunlukla adım adım anlatılmıştır. Bu yöntem problem çözümüne katkı sağlayacaktır. Ayrıca, sorular ve sorunlar karşısında çaresizliğe düşmeden, adımlara bölerek, problem çözme kabiliyeti kazanılacaktır.

ÖĞRENME FAALİYETİ-1

AMAÇ

Görsel programlamada grafiksel araçları doğru olarak kullanabileceksiniz.

ARAŞTIRMA

- Görsel programlama dilinde metot, özellik ve olay kavramlarını hatırlayınız.
- Dosya açma, dosyadan veri okuma ve dosyaya yazma yöntemleri hakkında bilgi sahibi olunuz.
- VB ortamına alet kutusunda yer almayan bileşenlerin nasıl yüklendiğini araştırınız.
- Verileri düzenli bir şekilde gösteren çizelgelerin mantığı konusunu inceleyiniz.

1. TEMEL GRAFİK İŞLEMLERİ

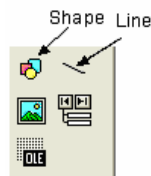
Grafik, görsel haberleşmenin bir biçimidir. Verilmek istenen haberin, bilginin çizgi ve noktalardan oluşan şekillerle muhatabına ulaştırma süreci diyebileceğimiz grafik, bilgisayar ortamında resimleri ve simgeleri gösterme ve işleme faaliyetine denir.

VB ortamında grafik, form üzerinde, PictureBox içinde ya da yazıcıda oluşturulur. Bu üçü grafik için bir ortam ya da yüzey oluşturur. VB ile gelen diğer kontroller grafik ile ilgili bir özellik sunmazlar. VB ortamında çizim, VB'nin sunduğu araçlarla iki yolla yapılır. Birisi metot adı verilen kodlarla çizim oluşturma, diğeri VB alet kutusunda yer alan çizim aletlerini kullanma. Windows'un hazır kütüphanesi olan API fonksiyonlarını ayrı tutuyoruz. Metotlarda çalışma görsel etki bakımından daha etkileyici ve çalışma alanı çok daha geniştir.

Metotların etkisi çalışma zamanında görünür. Çizim aletleri ise tasarım zamanında oluşturulur. Form üzerine bir ızgara çizilmesi istendiğinde alet kutundaki çizgi bileşeni ile her bir ızgara çizgisini yerleştirmek gerekirken kod ile birkaç satır da çizilebilir.

1.1 Çizim Aletleri ile Çalışma

VB alet kutusunda form üzerine çizim için kullanılmak üzere iki araç sunar. Bunlar:

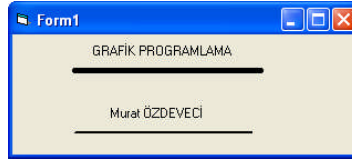


Şekil 1.1 : Çizim aletleri

- Line bileşeni: Belirtilen iki nokta arasında düz çizgiler için kullanılır.
- Shape bileşeni: Seçilen tipe bağlı olarak çeşitli şekiller çizer.

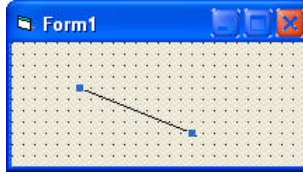
1.1.1. Line Bileşeni

Gerçek anlamdaki grafik uygulamalarında bu bileşen sık kullanılmasa bile form üzerindeki yazıların vurgulanmasında (Şekil 1.2) ya da form üzerine ızgara çizme gibi küçük uygulamalarda fevkalade faydalıdır.



Şekil 1.2 : Çizgi vurgusu

Line bileşenin üzerine fareyle çift tıkladığında VB, form üzerine çizgiyi yerleştirir.



Şekil 1.3 : Form üzerinde çizgi

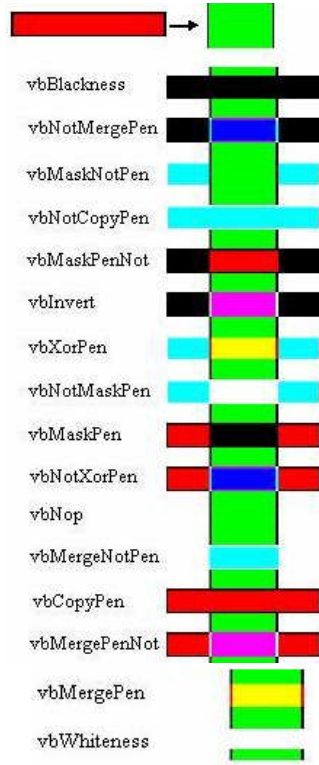
Çizginin her iki ucunda görünen iki küçük kare, çizginin uzatılması ya da daraltılmasını sağlar. Çizginin üzerinde farenin sağ tuşu basılı tutularak sağa-sola yada aşağı yukarı hareket ettirilir. Boyut ve konum ayarı yapılırken VB, özellikler penceresinde çizgiye ait değişkenleri günceller.

Çizginin Özellikleri:

- BorderColor: Çizgi rengini tayin eder.
- BorderStyle: Tablo 1.1’de görüldüğü gibi çizgi biçimini belirler.
- BorderWidth: Çizginin nokta cinsinden kalınlığı (Şekil 1.12). Çizginin görünmesi için değerinin en az 1 olması gerekir.
- X1,Y1,X2,Y2: Çizginin başlangıç ve bitiş koordinatları.
- Index: Çizginin dizi olarak tanımlanması durumunda sıra numarası.
- DrawMode: Çizginin ekran ya da yazıcıdaki görünümünü belirler. Çizimin yapıldığı ortam ile çizimin etkileşimini belirler. Geçerli değer olan 13-CopyPen, BorderColor ile belirlenen çizgiyi çizer. Alabileceği modlar Şekil 1.4’te görülmektedir.

No	İsim	Açıklama
0	Transparent	Şeffaf Çizgi
1	Solid	Düz Çizgi
2	Dash	Kesikli Çizgi
3	Dot	Noktalı Çizgi
4	Dash-Dot	Kesikli- Noktalı Çizgi
5	Dash-Dot-Dot	Kesikli-Noktalı-noktalı Çizgi

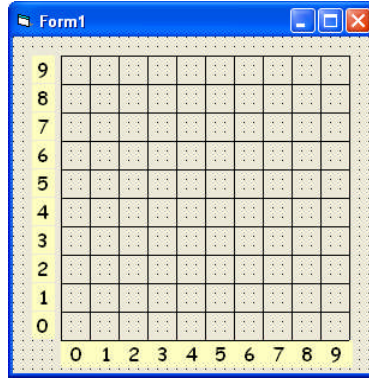
Tablo1.1 Çizgi şekilleri



Şekil 1.4 : DrawMode özelliği

ÖRNEK 1.1: Izgara Görüntüsü

Hedef noktamız Şekil 1.4 olacaktır. Bu amaçla aşağıdaki adımları takip edelim.

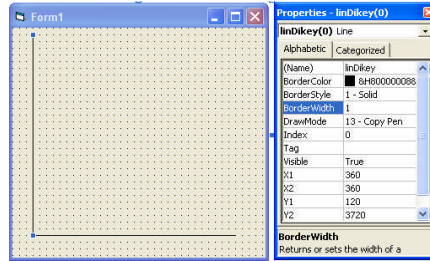


Şekil 1.5: Izgara deseni

ADIM 1: Form üzerine iki çizgi koyarak özelliklerini tabloda görüldüğü gibi ayarlayalım.

	Name	Index	X1	X2	Y1	Y2
1.Çizgi	linYatay	0	360	3960	3720	3720
2.Çizgi	linDikey	0	360	360	120	3720

Tablo1.2: Örnek 1.1 ayarları



ADIM 2: Diğer dikey çizgilerin X1 ve X2 değerlerini 360 artırarak 9 çizgi kopyalayalım. İndeks numarası verildiğinden çizgiler bir dizi olarak kabul edilecektir (en son çizginin koordinatları: X1: 3960, X2: 3960, Y1: 120, Y2: 3720).

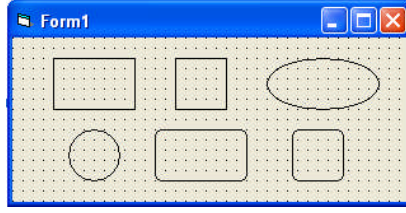
ADIM 3 : Diğer yatay çizgilerin Y1 ve Y2 değerlerini 360 azaltarak 9 çizgi kopyalayalım (en son çizginin koordinatları: X1: 360, X2: 3960, Y1: 120, Y2: 120).

ADIM 4: Her çizginin başlangıç noktasına Label bileşeni koyarak Caption özelliklerini ayarlayalım.

1.1.2. Shape Bileşeni

Line bileşeni sadece çizgi çizmesine karşılık Shape bileşeni birkaç şekil çizer. Şekil 1.5'te görüldüğü gibi bileşenin Shape özelliğine verilen değerlere göre değişik geometrik şekiller çizilir. Bunlar:

- 0-Rectangle: Dikdörtgen
- 1-Square: Kare
- 2-Circle: Daire
- 3-Oval: Elips
- 4-Rounded Rectangle: Köşeleri Yuvarlatılmış Dikdörtgen
- 5-Rounded Square: Köşeleri Yuvarlatılmış Kare



Şekil 1.6: Geometrik şekiller

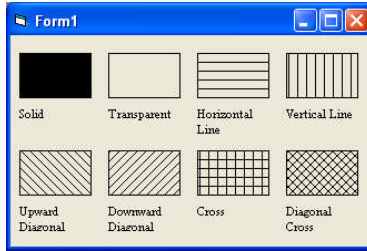
Shape özelliğinin yanı sıra Tablo 1.3'te görülen özellikler bileşenin form üzerindeki görünürlüğüne etki etmektedir.

Özellik	Açıklama
BackStyle	True değeri verilirse şekil şeffaf olur.
BorderColor	Şeklin kenar rengi
BorderStyle	Tablo 1.1 de görülen değerleri alarak şeklin kenar desenlerini tayin eder.
BorderWidth	Twip değeri olarak şeklin kenar çizgi kalınlığı
FillColor	Şeklin içini doldurma rengi
FillStyle	Şeklin dahili desen çeşidi (Tablo 1.4)
Height	Şeklin en yüksek noktası
Width	Şeklin en geniş noktası

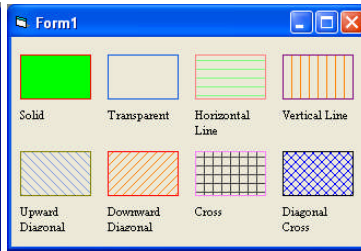
Tablo 1.3: Shape bileşenin görünürlük değişkenleri

FillStyle	Açıklama
Solid	Katı
Transparent	Şeffaf
Horizontal Line	Yatay Çizgi
Vertical Line	Dikey Çizgi
Upward Diagonal	Sola Yatık
Downward Diagonal	Sağa Yatık
Cross	Kareli
Diagonal Cross	Baklava Dilimi

Tablo 1.4: FillStyle desenleri



Şekil 1.7: FillStyle görünümleri

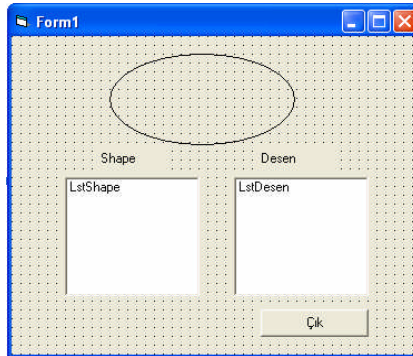


Şekil 1.8: FillColor özelliği

ÖRNEK 1.2: Shape nesnesinin şeklini ve desenini liste kutularından seçme.

ADIM 1: Yeni bir proje başlatın.

ADIM 2: Form üzerine nesnelere yerleştiriniz.



Şekil 1.9: Form yapısı

ADIM 3: Form üzerindeki bileşenlerin özelliklerini tabloda görüldüğü gibi ayarlayınız.

Kontrol İsmi	Özellik	Değer
Form	Height	4845
	Width	5610
Shape	Name	ShpOrnek
	Left	1320
	Width	2505
	Top	240
ListBox1	Height	1245
	Name	LstShape
	Left	720
	Width	1815
ListBox1	Top	1920
	Height	1620
	Name	LstDesen
	Left	3000
CommandButton1	Width	1815
	Top	1920
	Height	1620
	Name	cmdTerk
	Caption	Çık

ADIM 4: Formun kod bölümüne aşağıdaki kodları yazınız.

```

Private Sub Form_Load()
    'Şekilleri Ekle
    LstShape.AddItem "0-Rectangle"
    LstShape.AddItem "1-Square"
    LstShape.AddItem "2-Oval"
    LstShape.AddItem "3-Circle"
    LstShape.AddItem "4-Rounded Rectangle"
    LstShape.AddItem "5-Rounded Square"

    'Desenleri Ekle
    LstDesen.AddItem "0-Solid"
    LstDesen.AddItem "1-Transparent"
    LstDesen.AddItem "2-Horizontal Line"
    LstDesen.AddItem "3-Vertical Line"
    LstDesen.AddItem "4-Upward Diagonal"
    LstDesen.AddItem "5-Downward Diagonal"
    LstDesen.AddItem "6-Cross"
    LstDesen.AddItem "7-Diagonal Cross"

```

```

Liste Kutularının İlk Deęerini Ata
LstShape.ListIndex = 0
LstDesen.ListIndex = 0
End Sub

Private Sub LstDesen_Click()
'Seęime gre deseni seę
ShpOrnek.FillStyle = LstDesen.ListIndex
End Sub

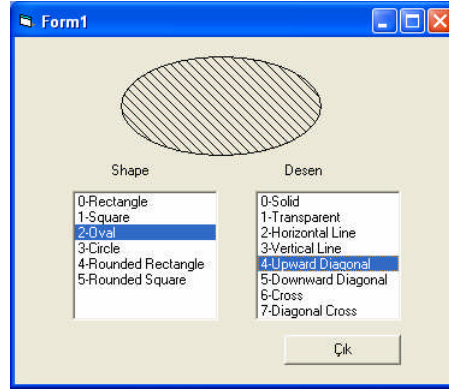
Private Sub LstShape_Click()
'Seęime gre Őekil seę
ShpOrnek.Shape = LstShape.ListIndex
End Sub

Private Sub cmdTerk_Click()
End
End Sub

```

Program Aıklaması

Programı alıřtırarak bir Őekil ve ona ait bir desen seęiniz. Seęilen renk ve desende form üzerindeki Őekil deęiřir.



Őekil 1.10: Program ıktısı

1.2. Grafik Metotları

VB, grafiksel nesnelerin yzey üzerinde izimini saęlamak iin birkaç metot sunar. Bu metotlar, sayısal deęerlerin grafiksel rneklerini yazıcı, form ya da PictureBox üzerine izer. Metotlar her uu iin de geerlidir. rneęin Line metodu form üzerinde yaptığı etkiyi aynı Őekilde PictureBox ya da yazıcıda da yapar.

Tablo 1.5, VB'nin destekledięi grafik metotları gstermektedir.

Bu metotların birkaçı X ve Y koordinatları ile form üzerinde çizim yapan kalem mantığını kullanır. Formun sıfır noktası (0,0), tahmin edildiği gibi sol alt köşe değil formun sol üst köşesidir. X değerleri formun sağına doğru, Y değerleri yukardan aşağı doğru indikçe artar. Çizim yapan hayali kalemin konumu formun CurrentX ve CurrentY değerleri ile denetlenir.

Metot	Açıklama
Line	Düz bir çizgi çizer.
Circle	Daire yada elips çizer.
Point	Bir noktanın rengini okur.
Print	Bir metni yazıcıya gönderir.
Pset	Bir pixel(resim noktası) koyar.
Cls	Tüm grafik öğelerini siler.
PaintPicture	Grafik dosyasının içeriğini çizer

Tablo 1.5: Grafik metotları

Koordinat sistemi neden kartezyen koordinat sistemine göre terstir? Sebebi ekran kartının görüntü elementlerini saklama biçimidir. Soldan sağa ve yukarıdan aşağı hareket eden bir taramalı elektron tabancası, ekran üzerindeki fosforlara gönderdiği ışınlarla görüntü oluşur. Ekranın sol üst köşesindeki ilk nokta ekran kartının hafızasındaki ilk noktadır. Bu yüzden ekran kartında tutulan bilgi ile ekran üzerindeki görüntü aynı olmaktadır.

Aşağıdaki metotlar hem form üzerinde hem de PictureBox üzerinde geçerlidir. Burada form üzerinde metotlar örneklerle anlatılacak. PictureBox kısmında teorik bilgi verilmeden uygulamaları yapılacaktır.

1.2.1 Line Metodu

Form üzerinde ve PictureBox dahilinde Line komutu ile çizgi ve kutular çizilir. Çizginin nerede başlayıp nerede bitmesi gerektiğini bildiren ifadeler içerir.

ÖRNEK 1.3: Çizgi Çizme

ADIM 1: Form üzerine bir düğme koyalım.

ADIM 2: Düğmenin Click olayına,

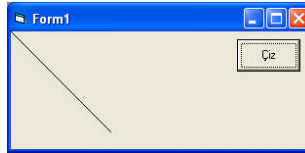
```
Private Sub Command1_Click()
```

```
Form1.Line (0, 0)-(1540, 1540)
```

```
End Sub
```

kodunu yazalım.

ADIM 3: Programı çalıştıralım.



Şekil 1.11: Program çıktısı

Görüldüğü gibi sol üst baştan aşağı doğru bir çizgi çizildi.

Line komutunun kullanımı:

Nesne.Line [Step] (X1,Y1) – [Step] (X2,Y2), Renk, B[F]

Line ifadesinin başındaki Nesne, çizim yapılacak ortamı temsil etmektedir. Picture1.Line ya da Form1.Line gibi.

(X1,Y1): Çizginin başlangıç noktasıdır ve isteğe bağlıdır. Başlangıç noktası ihmal edilirse bir önceki çizginin bitiş noktasından itibaren devam edilir. Single tipinde sayılar kullanılır.

(X2,Y2): Çizginin bitiş noktasıdır.

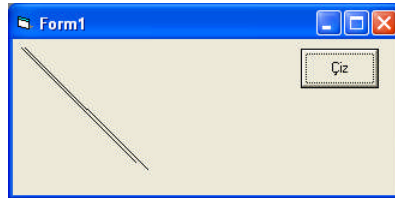
Step: Başlangıç ve bitiş noktalarının bir önceki noktadan olan uzaklığını söyler. Mutlak koordinat sisteminden eklemeli koordinat sistemine geçişi temsil eder. Yukarıdaki kodları şöyle değiştirelim.

```
Private Sub Command1_Click()  
Form1.Line (100, 100)-(1540, 1540)  
Form1.Line (150, 100)-Step(1540, 1540)  
End Sub
```

Programı çalıştırdığımızda ikinci çizginin y koordinatı değişmediği halde birinciden daha aşağıda olduğu görülür. Dolayısıyla ikinci çizginin bitim noktasının sıfır noktasına göre X ve Y değerleri:

$$X=150+1540=1690$$

$$Y=100+1540=1640 \text{ olacaktır.}$$



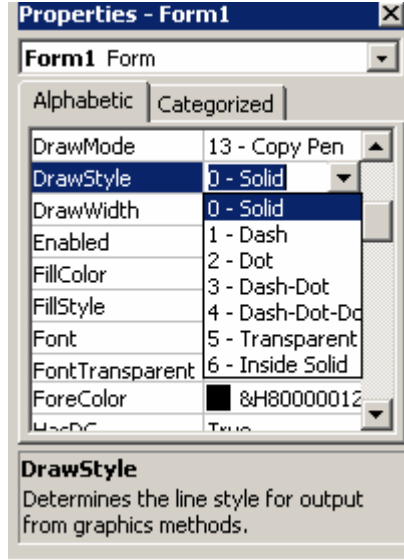
Şekil 1.12: Örnek çıktı

Renk: Çizim rengi

B[F]: Dikdörtgen çizilmesini sağlar. İçi boş dikdörtgen için B, içi dolu dikdörtgenler için BF kullanılır. Bu durumda (X2,Y2) noktası dikdörtgenin karşı köşesini gösterir. B olmadan F tek başına kullanılamaz.

DrawStyle özelliği çizginin çizim şeklini, aynen FillStyle özelliğinin kapalı alanların desen şeklini değiştirdiği gibi değiştirmektedir.

Şekil 1.11'de ve Tablo 1.6'da görüldüğü gibi DrawStyle yedi değerle temsil edilmektedir.



Şekil 1.13 DrawStyle özellikleri

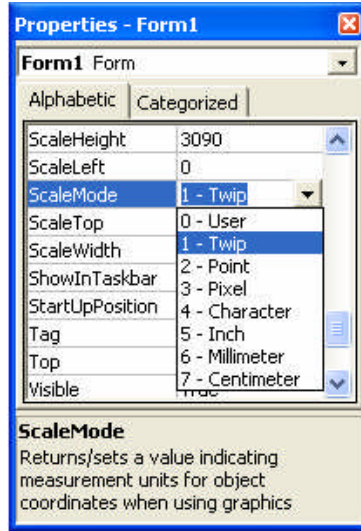
DrawStyle	Temsil	Program Kodu
Solid		vbSolid
Dash		vbDash
Dot		vbDot
DashDot		vbDashDot
DashDotDot		vbDashDotDot
Transparent		vbInvisible
Inside		vbInsideSolid

Tablo 1.6: DrawStyle çizgi tipleri

Burada çizilen çizginin uzunluğunu düşünelim. VB'nin standart uzunluk birimi twip'dir. Yirminci nokta anlamındadır. Bir twip bir parmak (inch=25.4 mm) uzunluğun 1/1440 oranındaki karşılığıdır. Bu birim donanım çözünürlüğünden bağımsız olacak şekilde yeterince küçük seçilmiştir. Birim piksel başına twip uzunluğu ekran çözünürlüğüne göre değişmez. 14" ekranda 1473 twip uzunluğunda olan çizgi uzunluğu, 17" ekranda aynı görünür.

Yukarıdaki örnekte (100,100) noktasından (1540,1540) noktasına bir çizgi çizdik. 768x1024 çözünürlüklü bir ekranda, çizginin X ve Y değerleri, 1540-100=1440 twip yani 1 parmaktır. Eğer çizgi birimi piksel olsaydı çizgi ekrana sığmazdı.

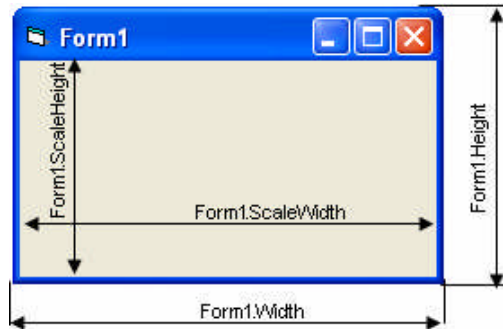
Bu ölçüm birimleri formun ScaleMode özelliğinden ayarlanır. ScaleMode özelliği tıklandığında VB'nin 8 değişik birimi desteklediğini görürüz.



Şekil 1.14: ScaleMode tipleri

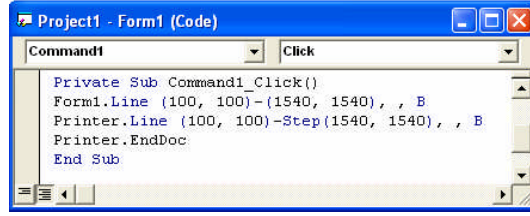
- **User:** Kullanıcı tanımlı ölçü birimidir. ScaleHeight, ScaleWidth, ScaleLeft ve ScaleTop ile negatif ve pozitif değerlikli kullanıcı tanımlı bir koordinat sistemi tanımlanabilir (Şekil 1.13).
- **Characters:** 120 twip genişliğinde ve 240 twip yüksekliğinde karakter sabiti.
- **Points:** Matbaacıların kullandığı puntoyla(nokta) eş değerdir. 72 nokta 1 inch uzunluğa tekabül eder. Buna göre twip, punto birim sisteminde 1/40 inçtir.
- **Centimeter:** Santimetre. 1 cm 567 twip değerindedir.

VB ortamında grafik oluşturulduğunda formun dahili sınırları içinde oluşturulmaktadır. Formun dış sınırları burada bileşenin yerleştirildiği X-Y koordinatlarına dahil edilmemektedir. Formun dahili sınırları formun ScaleWidth ve ScaleHeight ile temsil edilmektedir. Bu alanının başlangıç koordinatı(sıfır noktası) ScaleTop ve ScaleLeft özellikleri ile belirlenmektedir.



Şekil 1.15: Formun çalışma alanı

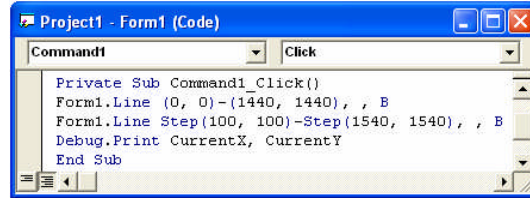
Yukarıdaki kod aşağıdaki gibi değiştirilir.



```
Private Sub Command1_Click()  
Form1.Line (100, 100)-(1540, 1540), , B  
Printer.Line (100, 100)-Step(1540, 1540), , B  
Printer.EndDoc  
End Sub
```

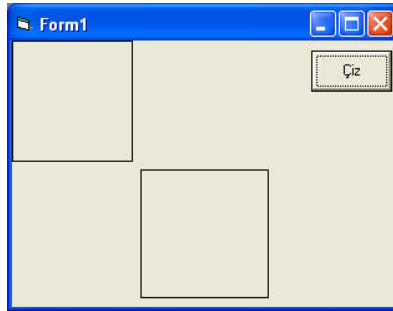
Burada Printer.Line ifadesi ile çizim doğrudan yazıcıya yönlendirilmektedir. Çizimin başlangıç ve bitiş koordinatları kenarları bir inç olan bir karenin köşegenleridir. Bilgisayara bağlı yazıcının hazır olduğundan emin olun. EndDoc ise yazıcıya gönderilecek verinin olmadığı anlamındadır.

Yukarıdaki kod bir daha değiştirilirse;



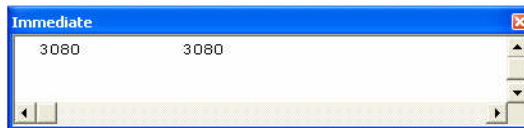
```
Private Sub Command1_Click()  
Form1.Line (0, 0)-(1440, 1440), , B  
Form1.Line Step(100, 100)-Step(1540, 1540), , B  
Debug.Print CurrentX, CurrentY  
End Sub
```

Program çalıştırıldığında;



Şekil 1.16: Ekran çıktısı

Görüldüğü gibi ikinci dikdörtgenin başlangıç noktası ilk dikdörtgenin bitiş noktasından 100 twip uzaklıktadır. Renk parametresi verilmediği için komut yazımında yeri boş bırakılmıştır. Debug.Print ifadesi değişkenlerin durumlarını izlemek için kullanılan Immediate penceresine son noktanın X ve Y değerlerini yazar. View menüsünden Immediate Window seçilerek yada Ctrl+G tuşlarına basılarak bu pencere ekrana getirilir.



Şekil 1.17: Debug görüntüsü

Örnek 1.4: Form üzerine kutular çizdirme.

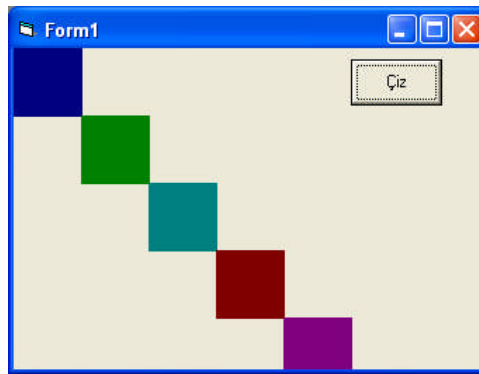
Bu amaçla, form üzerine “Çiz” başlıklı bir düğme yerleştirdikten sonra aşağıdaki kodlar yazılır.

```
Dim intBasX As Integer
Dim intBasY As Integer
Dim intBitX As Integer
Dim intBitY As Integer
Dim sayac As Integer
```

```
Private Sub Form_Load()
intBasX = 0
intBasY = 0
intBitX = 720
intBitY = 720
End Sub
```

```
Private Sub Command1_Click()
For sayac = 1 To 6
Form1.Line (intBasX, intBasY)-(intBitX, intBitY), QBColor(sayac), BF
'Diğer kutuları kaydır
intBasX = intBasX + 720
intBasY = intBasY + 720
intBitX = intBitX + 720
intBitY = intBitY + 720
Next sayac
End Sub
```

Program çalıştırıldığında birbirinden farklı renkte içi dolu kareler form üzerinde görünür.



Şekil 1.18: Ekran çıktısı

Eğer altıncı kutuyu görmek için form alt kenarından tutarak genişletilmeye çalışılırsa 5. karenin yarım ve 6. karenin ise yerinde olmadığı görülür. Formun özellikler penceresinde

“AutoRedraw” seçeneğini “True” yapılarak program yeniden çalıştırılır. Daha sonra, form genişletilip daraltılır. Şekiller hâlâ yerinde duruyor mu? AutoRedraw özelliği formun kendini sürekli olarak tazelemesini sağlar. Fakat bu şekilde kod yazmak hafıza israfına sebep olabilir. Farklı bir yöntem şekilleri formun Paint olayına çizmektir. Paint olayı, form üzerine herhangi bir bileşen konulduğunda sürekli formu tazelemektedir. Yukarıdaki kodu aşağıdaki gibi değiştirerek sonucu karşılaştırınız.

```
Dim intBasX As Integer
Dim intBasY As Integer
Dim intBitX As Integer
Dim intBitY As Integer
Dim sayac As Integer
```

```
Private Sub Command1_Click()
Form_Paint
End Sub
```

```
Private Sub Form_Load()
intBasX = 0
intBasY = 0
intBitX = 720
intBitY = 720
End Sub
```

```
Private Sub Form_Paint()
For sayac = 1 To 6
Form1.Line (intBasX, intBasY)-(intBitX, intBitY), QBColor(sayac), BF
'Diğer kutuları kaydır
intBasX = intBasX + 720
intBasY = intBasY + 720
intBitX = intBitX + 720
intBitY = intBitY + 720
Next sayac
End Sub
```

Yukarıda kullanılan renk parametrelerini biraz açalım.

VB’de renk ataması için dört yöntem vardır.

- RGB(Red,Gren,Blue) fonksiyonunu kullanarak,
- QuickBasic’ten gelen, 16 rengi QBColor fonksiyonu ile kullanarak,
- VB’nin içine gömülü renk sabitlerini kullanarak,
- Renk kodlarını doğrudan yazarak.

Yukarıdaki kodda QBColor fonksiyonu kullanıldı. Tablo 1.7’de bu renkler görülmektedir.

Numara	Renk
0	Black
1	Dark blue
2	Dark green
3	Dark cyan
4	Dark red
5	Dark lila
6	Dark yellow
7	Dark white/grey
8	Dark grey
9	Bright blue
10	Bright green
11	Bright cyanogen
12	Bright red
13	Bright lila
15	Bright yellow
16	Bright white

Tablo 1.7 : QBColor renk deęerleri

RGB fonksiyonuna Tablo 1.8'de gsterilen temel renk deęerleri yazılabildięi gibi RGB(112,222,98) gibi karıřımlarda yazılabilir.

Sabit Deęer	Renk	RGB
vbBlue	Blue	RGB(0,0,255)
vbYellow	Yellow	RGB(255,255,0)
vbGreen	Green	RGB(0,255,0)
vbMagenta	Magenta	RGB(255,0,255)
vbRed	Red	RGB(255,0,0)
vbBlack	Black	RGB(0,0,0)
vbWhite	White	RGB(255,255,255)
vbCyan	Cyan	RGB(0,255,255)

Tablo 1.8: RGB renk deęerleri

Program koduna sabit deęer ataması Shape1.FillColor = vbYellow yadaForm1.Line (0, 0)-(1440, 1440), vbRed, BF řeklinde kullanılabilir.

VB'nin iine ggmmlu renk sabitleri temel renkleri ve sistem renklerini (menu ubuęu, glge rengi) ayrı ayrı ifade etmektedir.

Tablo 1.9'da VB renk sabitlerini, Tablo 1.10'da ise sistem renk sabitlerini gstermektedir.

Sabit	HEX Deęeri	Renk
vbBlue	&HFF0000	Blue
vbBrown	&H80&	Brown
vbYellow	&HFFFF&	Yellow
vbGray	&HC0C0C0	Gray
vbGreen	&HFF00&	Green
vbOrange	&H80FF&	Orange
vbMagenta	&HFF00FF	Magenta
vbRed	&HFF&	Red
vbBlack	&H0&	Black
vbWhite	&HFFFFFF	White
vbCyan	&HFFFF00	Cyan

Tablo 1.9: Renk sabitleri

Sabit	HEX Deęeri	Açıklama
vbScrollBars	&H8000000	ScrollBar Rengi
vbDesktop	&H80000001	Desktop Rengi
.....
vbWindowText	&H80000008	Windows Metin Rengi
vbButtonText	&H80000012	Düğme Rengi
.....

Tablo 1.10: Sistem renkleri

1.2.2 Pset Metodu

Pset (Point Set: Nokta yerleřtir), çizim alanındaki herhangi bir noktaya belirli bir renkte bir piksel koyar.

Kullanımı: Nesne.Pset (X1,Y1), Renk

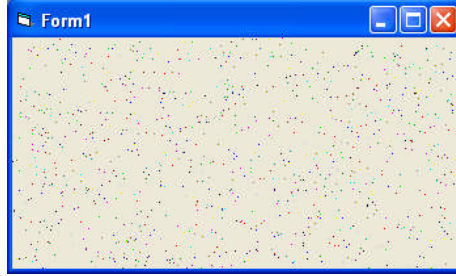
ÖRNEK 1. 5: Form üzerine rastgele piksel koyma.

ADIM 1: Boş bir form oluşturulur.

ADIM 2: Formun Paint olayına ařaęıdaki kodlar yazılır.

```
Private Sub Form_Paint()
For i = 1 To 1000
'0-15 arasında rasgele bir renk seę
RenkSec = (Int(15 * Rnd))
'0-ScaleWidth arasında bir X deęeri seę
X = (Int(Form1.ScaleWidth * Rnd))
'0-ScaleHeight arasında bir Y deęeri seę
Y = (Int(Form1.ScaleHeight * Rnd))
Form1.PSet (X, Y), QBColor(RenkSec)
Next
End Sub
```

ADIM 3: Program çalıştırılır.



Şekil 1.19 : Ekran çıktısı

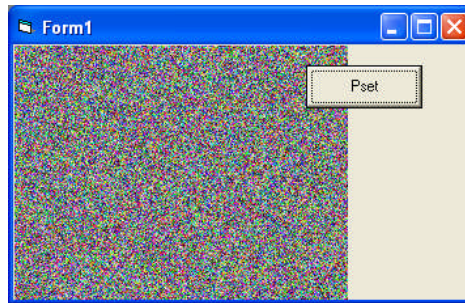
ÖRNEK 1.6: Pset ile karlama deseni oluşturma.

ADIM 1: Boş bir form üzerine bir düğme yerleştirerek, Caption özelliği Pset yapılır.

ADIM 2: Düğmenin Click olayına aşağıdaki kodlar yazılır.

```
Private Sub Command1_Click()  
ScaleMode = 3 'Piksel  
For i = 1 To 255  
For j = 1 To 255  
Red = CInt(Rnd * 255)  
Green = CInt(Rnd * 255)  
Blue = CInt(Rnd * 255)  
Form1.PSet (i, j), RGB(Red, Green, Blue)  
Next j  
Next i  
End Sub
```

ADIM 3: Program çalıştırılır.



Şekil 1.20: Ekran çıktısı

ADIM 4: ScaleMode değerine 0-7 arasında değişik değerler vererek sonuçları karşılaştırınız.

1.2.3 CLS Metodu

Form ya da PictureBox yüzeyini siler.Yukarıdaki örneğe bir düğme eklenerek aşağıdaki kod yazılır.

```
Private Sub Form_Click()  
Form1.Cls  
Picture1.Cls  
End Sub
```

1.2.4 Circle Metodu

Belirtilen noktaya belli bir renkte ve yarıçapta çember çizer. İfadesi:

Nesne.Circle [Step] (x, y), Radius, [Color, Start, End, Aspect]

Burada :

X;Y: Çemberin merkezi.

Radius: Çemberin yarıçapı.

Color: Çizim rengi.

Start,End: Yay ya da elips çizmek için kullanılır. Yayın başlangıç ve bitiş değerlerini radyan cinsinden verir. Varsayılan değeri Start için 0, End için 2*pi yani çemberin çevresidir.

Step: Çemberin merkez noktası, CurrentX ve CurrentY ile gösterilen koordinattır.

Aspect: Basıklık oranı. Bir eksenin diğer eksene olan oranıdır. Bu değer çember için birdir. Farklı olursa elips olur. Birden küçük olursa yatay elips, büyük olursa dikey elips olur.

Kullanım örnekleri:

Circle (36,54), 25, RGB(Rnd * 255, Rnd * 255, Rnd * 255)

Circle (234,234),65,QBColor(11),

Circle (1234, 723), 548, QBColor(12), 0.5, 5.9

Yukarıdaki son komut (0.5-5.9) radyan arasında bir yay çizer.Bunun başka bir ifadesi de katsayılarla çemberin çevresini çarpmaktır.

$c = 2 * 3.14159$

Circle (CX, CY), 548, QBColor(4), $c * 0.5$, $c * 0.9$

Circle (CX, CY), 548, QBColor(4), 0, 0, 1'Daire çizer

Circle (CX, CY), 548, QBColor(4), 0, 0, 0.8'Elips çizer

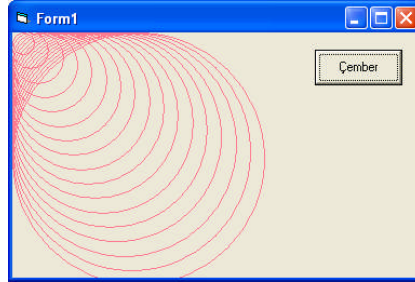
ÖRNEK 1,7: Büyüyen çember çizimi.

ADIM 1: Boş bir forma bir düğme yerleştirerek, Caption özelliğini “Çember” yapılır.

ADIM 2: Düğmenin Click olayına aşağıdaki kodlar yazılır.

```
Project1 - Form1 (Code)
Command1 Click
Private Sub Command1_Click()
For i = 150 To 1800 Step 100
Me.Circle (i, i), i, RGB(255, 123, 145)
Next i
End Sub
```

ADIM 3: Program çalıştırılır.



Şekil 1.21: Ekran çıktısı

RGB(255, 123,145) ifadesini RGB(Rnd * 255, Rnd * 255, Rnd * 255) ifadesi ile yer değiştirilir.Burada Form1 yerine Me kelimesi yazıldı. Me sözcüğü, burada çalışan formu temsil etmektedir.

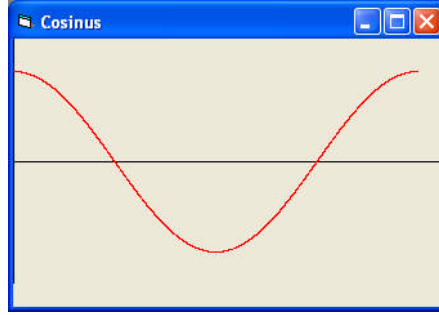
ÖRNEK 1.8: Cosinüs fonksiyonunun grafiği.

ADIM 1: Height: 3630 ve Width: 5130 boyutlu bir form açılır.

ADIM 2: Formun kod bölümüne ilgili kodlar yazılır.

```
Project1 - Form1 (Code)
Form Load
Const pi = 3.14
Private Sub Form_Click()
For i = 0 To 2 * pi Step 0.001
x = i * 50
y = 95 - (70 * Cos(i))
PSet (x, y), &HFF
Next i
End Sub
Private Sub Form_Load()
Form1.ScaleMode = 3
Form1.AutoRedraw = True
Form1.Line (0, 95)-(350, 95)
Form1.Line (0, 0)-(0, 190)
End Sub
```

ADIM 3: Program çalıştırılır.



Şekil 1.22: Ekran çıktısı

Burada

$$y = 95 - (70 * \text{Cos}(i))$$

satırında hesaplanan kosinüs değerleri çok küçük olduğundan sabit bir sayı ile çarpılmakta ve çıkan sayı yine bir sabit sayıdan çıkarılmaktadır. Neden? Koordinat sisteminde Y değeri ters olduğu için. Bu çıkarma işlemi ile grafiği günlük hayatta gördüğümüz gibi görüyoruz.

ÖRNEK 1.9: Trigonometrik hesaplamayla çember çizimi.

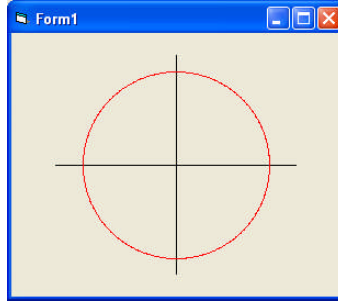
Daha önce Circle metodunu kullanarak çember çizilmişti. Kutupsal koordinatlardan yararlanarak bir çemberin nasıl çizileceği aşağıda görülmektedir .

ADIM 1: Yüksekliği 4035 piksel, genişliği 4650 piksel olan boş bir form açılır.

ADIM 2: Formun kod bölümüne aşağıdaki kodlar yazılır.

```
Const pi = 3.14
Private Sub Form_Click()
For i = 0 To 2 * pi Step 0.01
x = 150 + (85 * Sin(i))
y = 120 - (85 * Cos(i))
PSet (x, y), &HFF
DoEvents
Next i
End Sub
Private Sub Form_Load()
Form1.ScaleMode = 3
Form1.AutoRedraw = True
Form1.Line (40, 120)-(260, 120)
Form1.Line (150, 20)-(150, 220)
End Sub
```

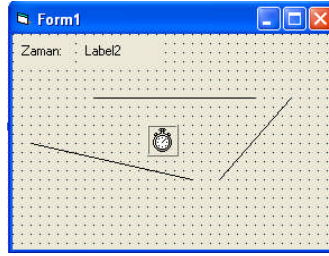
ADIM 3: Program çalıştırılır.



Şekil 1.23: Ekran çıktısı

ÖRNEK 1.10: Analog saat.

ADIM 1: Yüksekliği 4410, genişliği:5280 olan bir form açarak, üzerine alet kutusundan üç tane Line bileşeni, bir adet Timer bileşeni ve iki adet Label nesnesi yerleştirilir. Label bileşenlerinden birincisinin Caption özelliğini Zaman yapılır.



Şekil 1.24: Form görüntüsü

ADIM 2: Pi sabiti tanımlanır.

Const pi = 3.14

Pi sayısı şu şekilde de tanımlanabilir: pi = 4 * Atn(1)

ADIM 3: Formun Load olayına başlangıç değerleri yüklenir.

```
Private Sub Form_Load()  
Form1.ScaleMode = 3'Piksel modu  
Form1.AutoRedraw = True  
Timer1.Interval = 1000'Zaman aralığı 1 saniye  
With Line1  
X1 = 150  
X2 = 150  
Y1 = 150  
Y2 = 150  
BorderColor = &HFF  
BorderWidth = 1  
End With
```

```

With Line2
X1 = 150
X2 = 150
Y1 = 150
Y2 = 150
BorderColor = &H0
BorderWidth = 2
End With
With Line3
X1 = 150
X2 = 150
Y1 = 150
Y2 = 150
BorderColor = &H0
BorderWidth = 4
End With
End Sub

```

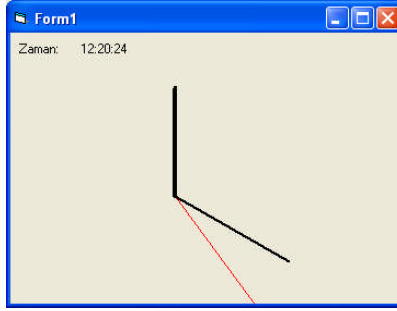
ADIM 4: Saat, dakika ve saniyeyi gösterecek çizgilerin konumu hesaplanarak grafiksel olarak gösterilir.

```

Private Sub Timer1_Timer()
Label2.Caption = Time
Saat = Hour(Time)
Dakika = Minute(Time)
Saniye = Second(Time)
Saat = (Saat / 12) * 2 * pi
Dakika = (Dakika / 60) * 2 * pi
Saniye = (Saniye / 60) * 2 * pi
With Line3
X2 = 150 + (100 * Cos(Saat - 1.57))
Y2 = 150 + (100 * Sin(Saat - 1.57))
End With
With Line2
X2 = 150 + (120 * Cos(Dakika - 1.57))
Y2 = 150 + (120 * Sin(Dakika - 1.57))
End With
With Line1
X2 = 150 + (125 * Cos(Saniye - 1.57))
Y2 = 150 + (125 * Sin(Saniye - 1.57))
End With
End Sub

```

ADIM 5: Program çalıştırılır.



Şekil 1.25: Program çıktısı

Saat, dakika ve saniyeyi gösteren değişkenler Time fonksiyonundan sırayla saat (Hour), dakika(Minute) ve saniye(Second) bilgilerini almaktadır.

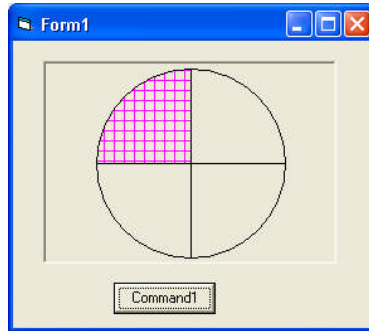
ÖRNEK 1.11: Daireyi dörde bölerek ikinci çeyreğinin bir desenle doldurulması.

ADIM 1: Form üzerine bir düğme ve picturebox yerleştirilir.

ADIM 2: Aşağıdaki kodlar yazılır.

```
Private Sub Command1_Click()  
Dim c As Single  
c = 2 * 3.14159  
Picture1.Cls  
CX = Picture1.ScaleWidth / 2  
CY = Picture1.ScaleHeight / 2  
Picture1.Circle (CX, CY), 75, , 0, -0.25 * c  
Picture1.FillStyle = 6  
Picture1.FillColor = vbMagenta  
Picture1.Circle (CX, CY), 75, , -0.25 * c, -0.5 * c  
Picture1.FillStyle = 1 "Transparent"  
Picture1.Circle (CX, CY), 75, , -0.5 * c, -0.75 * c  
Picture1.Circle (CX, CY), 75, , -0.75 * c, -1 * c  
End Sub
```

ADIM 3: Program çalıştırılır.



Şekil 1.26: Program çıktısı

Programı daha iyi anlamak için

```
Picture1.Circle (CX, CY), 75, , -0.25 * c, -0.5 * c  
Picture1.FillStyle = 1 'Transparent  
Picture1.Circle (CX, CY), 75, , -0.5 * c, -0.75 * c  
Picture1.Circle (CX, CY), 75, , -0.75 * c, -1 * c
```

satırlarını etkinlikten çıkartarak, tekrar çalıştırılır.

1.2.5 Fare ile Çalışma

Şu ana kadar form üzerinde Windows'un en güzel ve güçlü yanlarından birisi olan fare kullanılmadı. Aşağıda form üzerine gelişmiş güzel çizim yapacak bir program üzerinde durulmuştur.

Fare Textbox gibi özellikleri ve metotları olan bir nesne değildir. Aksine farenin eylemleri diğer VB bileşenlerinin içine gömülmüştür. Sözgelimi Label bileşenin ile formun aynı fare olayları vardır. Farenin üç olayı vardır. Bunlar:

- Form_MouseDown (Button As Integer, Shift As Integer, X as Single, Y As Single)
- Form_MouseUp (Button As Integer, Shift As Integer, X as Single, Y As Single)
- Form_MouseMove (Button As Integer, Shift as Integer, X as Single, Y As Single)

Farenin hareketi için MouseDown, düğmesinin tıklanması için MouseDown ve serbest bırakılması için MouseUp olayları meydana gelir. Farenin hareketi yada tıklanması kod ile desteklenmediği sürece tek başına anlam ifade etmez. Alacağı bir dizi parametreye göre kendisinden beklenen görevi yerine getirmeye hazırdır. Bu parametreler:

X,Y: ScaleMode ile belirlenen koordinat sistemindeki fare işaretçisinin bulunduğu X,Y değerleri. Bulunan son nokta bu değerler okunarak bulunur.

Button: Farenin varolan üç tuşundan hangisine basıldığı hususunda bilgi verir. Bu 0 ile 7 arasında değişen bir sayıdır.

Aşağıdaki kod hangi tuşa basıldığını göstermektedir.

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, _  
Y As Single)  
Select Case Button  
Case 0 'Hiçbir düğmeye basılmadı.  
Case 1 'Sadece sol tuşa basıldı.  
Case 2 'Sadece sağ tuşa basıldı.  
Case 3 'Sadece sol ve sağ tuşa basıldı.  
Case 4 'Sadece orta düğmeye basıldı.  
Case 5 'Sadece sol ve orta düğmelere basıldı.
```

Case 6 'Sadece sađ ve orta düğmelere basıldı.

Case 7 'Tüm düğmelere basıldı.

End Select

End Sub

Eđer basitçe hangi düğmeye basıldığını görmek istersek aşağıdaki kodu yazabiliriz.

```
Private Sub Form_MouseMove(Button As Integer, _  
Shift As Integer, X As Single, Y As Single)  
If Button AND 1 then Print "Sol düğmeye basıldı"  
If Button AND 2 then Print " Sađ düğmeye basıldı "  
IF Button AND 4 then Print " Orta düğmeye basıldı "  
End Sub
```

Shift: Hangi Shift tuşuna basıldığını bildirir. Sol shift, sađ shift, Ctrl+Shift gibi. Özellikle Windows uygulamalarında sık karşılaşılır. Örneğin Shift tuşuna basılı tutarken Geri Dönüşüm Kutusu'na sürüklenen bir dosya veya klasör tamamen silinir.

Deđerleri:

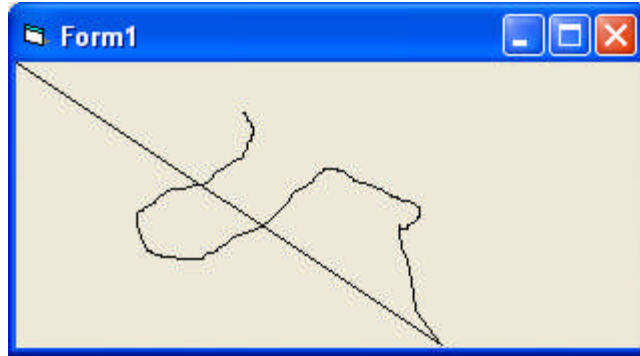
- 0 - SHIFT, CTRL, yada ALT tuşlarının hiçbirine basılmadı
- 1 - SHIFT basılı
- 2 - CTRL basılı
- 3 - ALT basılı
- 4 - SHIFT ve CTRL basılı
- 5 - SHIFT ve ALT basılı
- 6 - CTRL ve ALT basılı
- 7 – Tümü(SHIFT, CTRL ve ALT) basılı

Farenin hangi tuşuna basılı olduğunu sınamak için sabit numaralar yerine ifadesel sabitleri kullanmak daha uygundur. Örneğin If Button = vbLeftButton satırı If Button = 1 satırından daha anlaşılırdır.

Diđer sabitler vbRightButton ve vbMiddleButton sabitleridir.Şimdi yeni bir form açarak, Form_MouseMove olayına aşağıdaki kodu yazalım.

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
Line -(X, Y)  
End Sub
```

Program çalıştırılıp fare hareket ettirildiğinde hemen çizime başladı mı? Çizime (0,0) noktasından başlaması dođal. Çünkü Line -(X,Y) komutu (Step parametresi ihmal edilebilir) en son noktadan itibaren çizgi çizmektedir. Son nokta da program ilk defa çalıştığı için başlangıç noktasıdır. Fakat bir sıkıntı var. Alışageldiğimiz gibi çizimin farenin sol tuşuna basıldığında olmasını bekliyoruz.



Şekil 1.27: Program çıktısı

Kod aşağıdaki gibi değiştirilir.

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
If Button = vbLeftButton Then Line -(X, Y)  
End Sub
```

Şimdi farenin sol tuşu basılı tutulduğu sürece çizim yapmaktadır. Fakat birbirinden ayrı çizgiler de çizilememektedir. Yapılması gereken fareyi her tıkladığımızda farenin son konumunu CurrentX ve CurrentY değerlerine eşitlemektir. Bu Form_MouseDown olayına yazılacaktır. Farenin tuşuna basılı tutulup hareket ettirildiğinde MouseDown olayı içinde farenin son konumu CurrentX ve CurrentY değerlerine aktarır.

Aşağıdaki kod projeye ilave edelir.

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
CurrentX = X  
CurrentY = Y  
End Sub
```

Program çalıştırıldığında, çizgi istenilen yerden başlatılabilir. Fakat farenin tıkladığı nokta boş geçilmektedir. Bu noktaya da bir nokta koydurmak için aşağıdaki kod eklenir.

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
CurrentX = X  
CurrentY = Y  
PSet (X, Y)  
End Sub
```

Program çalıştırılır.



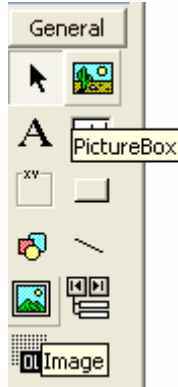
Şekil 1.28: Program çıktısı

Bu arada fare ile yazı yazmanın ne kadar zor olduğunu göreceksiniz.

1.3. PictureBox

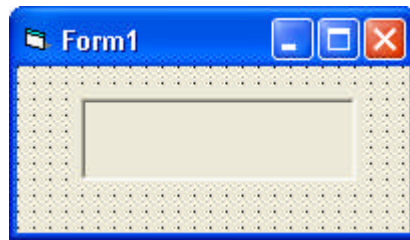
PictureBox, en basit ifade ile resimleri gösteren bir ara yüzdür. Bunun yanında kendisine yüklenen resim ya da grafikleri değişik resim formatlarına kaydedebilir. Yazıcıya bastırabilir, piksel piksel işleyebilir. Alet kutusunun sağ üst köşesinde bulunur.

PictureBox, kendisi gibi resimleri gösterme yeteneğine sahip Image bileşeninden çok daha fazla işlevselliğe sahiptir. Image bileşeni Öğrenme Faaliyeti 3'te incelenektir.



Şekil 1.29: PictureBox simgesi

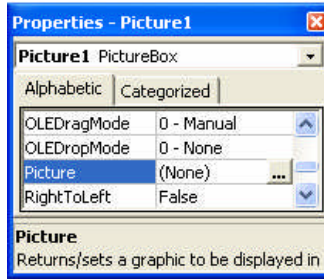
Form üzerine PictureBox yerleştirdiğimizde BorderStyle özelliği varsayılan olarak "1- Fixed Single" değerine ayarlandığından 3 boyutlu görünür. 0-None seçilirse Image bileşeninde olduğu gibi düz bir yüzey görünür.



Şekil 1.30:BorderStyle=0

1.3.1 Resim Yükleme

PictureBox'a tasarım zamanında ve çalışma zamanında ayrı ayrı resim yüklenebilir. Tasarım zamanında *Picture* özelliği ile açılan pencereden resim seçilerek form üzerine aktarılır. Bu esnada PictureBox, kendi boyutlarını resmin büyüklüğüne göre ayarlamaz. AutoSize özelliği True yapılırsa resmin boyutuna göre resim çerçevesi ayarlanır. Yine AutoRedraw özelliği True yapılarak resmin önüne bir pencere getirilip çekilirse resmin tekrar görünmesini yani resim kutusu içeriğinin tazelenmesi sağlanmış olur.



Şekil 1.31: Picture özelliği

PictureBox içine bitmap(.bmp) dosyalarından metafile(.wmf) dosyalarına kadar her çeşit resim dosyası yüklenebilir.Çalışma zamanında resim yüklemek için LoadPicture metodu kullanılır. Kullanımı:

LoadPicture([ResimDosyasıAdı], [Ölçü], [RenkDerinliği], [x,y])

Burada:

- **ResimDosyasıAdı:** Seçimlidir. Resim dosyasının ismi. Bu isim, sürücü ve klasör isimlerini ihtiva edebilir. Herhangi bir isim belirtilmezse PictureBox ya da Image kutusunu temizler. Program dahilinde bir düğmenin tıklama olayına eklenebilir.

```
Private Sub Command1_Click()  
Picture1.Picture = LoadPicture("c:\okul.jpg")  
End Sub
```

- **Ölçü:** Seçimlik. Dosya ismi bir cursor ya da icon dosyası ise arzulanan resim ölçüsünü betimler.
- Alacağı değerler: vbLPSmall, vbLPLarge, vbLPSmallShell, vbLPLargeShell ve vbLPCustom(resmin kendi boyutları).
- **RenkDerinliği:** Seçimlik. Dosya ismi bir cursor ya da icon dosyası ise arzulanan renk doygunluğunu ayarlar.
- Alacağı değerler: vbLPDefault, vbLPMonochrome, vbLPVGAColor(16 renk), vbLPColor(256 renk)
- **x,y:** Seçimlik. Dosya ismi bir cursor ya da icon dosyası ise istenilen genişlik ve yükseklik değerleri. RenkDerinliği vbLPCustom'a ayarlanırsa geçerlidir.

ÖRNEK 1.12: LoadPicture metodu

Form üzerine bir buton ve bir resim kutusu yerleştirin. Kod listesi:

```
Private Sub Command1_Click()  
Dim Msg As String  
Height = 3990  
Width = 4890 'Yükseklik ve Geniřlięi Ayarla  
'Cursor'u ykle  
Picture1.Picture = LoadPicture("c:\WINDOWS\Cursors\HAND-IL.CUR", _  
vbLPCustom, vbLPColor, 32, 32)  
Msg = "Resmi temizlemek iin Tamam deyin."  
MsgBox Msg  
Picture1.Picture = LoadPicture() 'Resim kutusunu temizle.  
End Sub
```

Program alıřtırıldığında;



Őekil 1.32: Program sonucu

Burada kullanılan cur uzantılı dosyanın adresini, siz kendi bilgisayarınızdan ayarlayınız.

1.3.2 PictureBox Olayları

PictureBox da dięer tm kontroller gibi eřitli olaylara tepki verir. Szgelimi fareyle resim kutusu zerinde tıklamak bir Click(hatta DblClick) olayı meydana gelmesine sebep olur. Ancak nerede tıklandığına dair bir bilgi iermez. Bu durumda MouseDown olayını kullanmak gerekir. Bu durumda nce hangi l sistemi kullanılacağına karar vermek gerekir.

```
Private Sub Form_Load()  
Picture1.ScaleMode = vbPixels  
End Sub
```

MouseDown olayı ile farenin basıldıęı noktanın koordinatları, X ve Y deęiŐkenlerine aktarılır.

PictureBox'ın ScaleMode özelliği "6-Millimeter" olarak değiştirildiğinde aşağıdaki kod kutu içerirse tıklanan yere göre mesaj vermektedir.

```
Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, _
X As Single, Y As Single)
If X > 1 And X < 234 And Y > 12 And Y < 234 Then
MsgBox "Yanlış yerde Geziniyorsun"
Else
MsgBox "Doğru yerdesin"
End If
End Sub
```

PictureBox klavye tuşlarına basılmasıyla meydana gelen KeyDown, KeyUP KeyPress gibi olaylara karşı da oldukça hassastır. KeyPress tuşa basılıp bırakılmasıyla KeyDown basılma anında, KeyPress tuşun bırakılması anında meydana gelir.

ÖRNEK 1.13: Resim kutusunu klavye ile hareket ettirme.

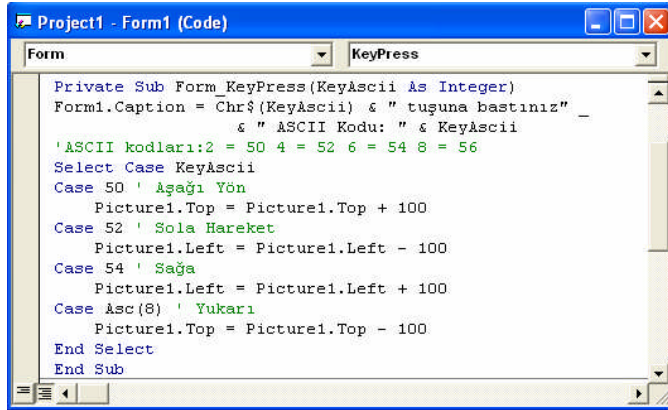
ADIM 1:Boş bir forma bir picturebox ekleyerek BackColor özelliğini &H0000FFFF& (sarı) olarak ayarlanır.

ADIM 2: Formun Load olayına aşağıdaki kod yazılır.

```
Private Sub Form_Load()
KeyPreview = True
MsgBox ("PictureBox'ı gezdirmek için 2, 4, 6, 8 rakamlarına ya da yön tuşlarına basınız")
End Sub
```

Buradaki KeyPreview True yapılarak formun, üzerine konulan bileşenlerden daha önce klavye olaylarına tepki vermesi sağlanır.

ADIM 3 : Klavyenin sağ bölümünde bulunan belirli tuşlara karşı tepki verilmesi.



```
Project1 - Form1 (Code)
Form
KeyPress
Private Sub Form_KeyPress(KeyAscii As Integer)
Form1.Caption = Chr$(KeyAscii) & " tuşuna bastınız" _
& " ASCII Kodu: " & KeyAscii
'ASCII kodları:2 = 50 4 = 52 6 = 54 8 = 56
Select Case KeyAscii
Case 50 ' Aşağı Yön
Picture1.Top = Picture1.Top + 100
Case 52 ' Sola Hareket
Picture1.Left = Picture1.Left - 100
Case 54 ' Sağa
Picture1.Left = Picture1.Left + 100
Case Asc(8) ' Yukarı
Picture1.Top = Picture1.Top - 100
End Select
End Sub
```

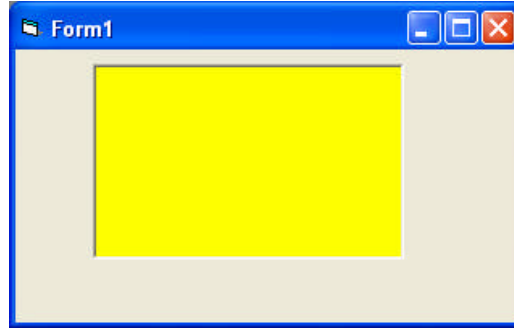
Form üzerinde işaret edilen tuşlara basıldığında resim kutusu hareket edecektir. Yukarıya doğru hareket etmek için diğerlerinden farklı olarak ASC(8) kullanıldığına dikkat ediniz. Burada sadece farklı kullanım göstermek için yazıldı. Bunun yerine “Case 56 “ da yazılabilirdi.

ASC, karakter kodunu getirir. Örneğin Asc("A") komutu, A'nın ASCII kodu olan 65 sayısını döndürecektir. Resim kutusunu yukarı çıkarmak için Picture1.Top değerinden 100 sayısının çıkarıldığına dikkat ediniz. Neden eklenmiyor? Koordinatın kartezyen koordinat sistemine göre ters olduğunu hatırlayınız.

ADIM 4: Resim kutusunu yön tuşlarıyla hareket ettirelim.

```
Project1 - Form1 (Code)
Form
KeyPress
! Ok tuşları
Private Sub Picture1_KeyDown(KeyCode As Integer, _
    Shift As Integer)
Select Case KeyCode
Case vbKeyDown
    Picture1.Top = Picture1.Top + 100
Case vbKeyLeft
    Picture1.Left = Picture1.Left - 100
Case vbKeyRight
    Picture1.Left = Picture1.Left + 100
Case vbKeyUp
    Picture1.Top = Picture1.Top - 100
End Select
End Sub
```

ADIM 5: Program çalıştırılır.



Şekil 1.33: Program çıktısı

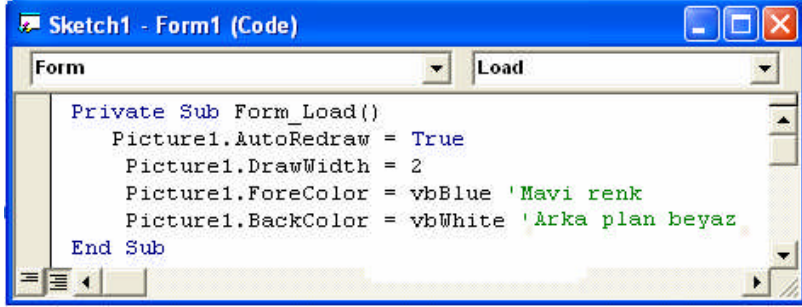
1.3.4. PictureBox Grafik Metotları

Picturebox bileşeninde de Form için geçerli olan grafiksel metotlar aynen geçerlidir. Tek fark metodun başına picturebox bileşenine atanan Name değerinin yazılmasıdır. Picture1.Cls gibi.

ÖRNEK 1.14: Fare ile serbest çizim.

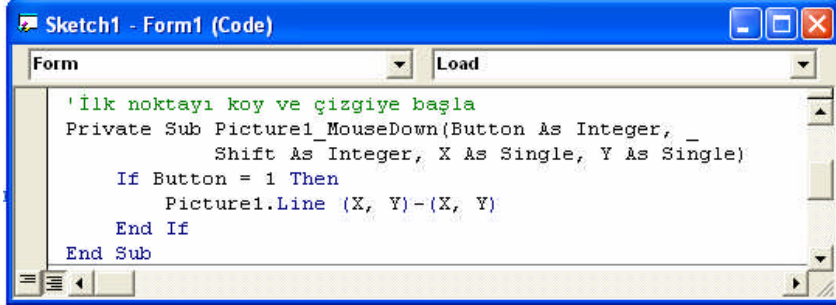
ADIM 1: Form üzerine bir PictureBox koyarak, sınırları formun dahili sınırlarına kadar genişletilir.

ADIM 2: Load Olayı.



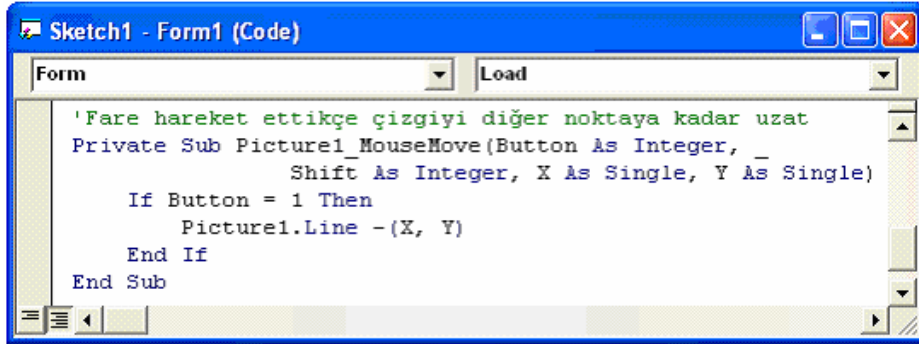
```
Sketch1 - Form1 (Code)
Form Load
Private Sub Form_Load()
    Picture1.AutoRedraw = True
    Picture1.DrawWidth = 2
    Picture1.ForeColor = vbBlue 'Mavi renk
    Picture1.BackColor = vbWhite 'Arka plan beyaz
End Sub
```

ADIM 3: Fare ile hareket.



```
Sketch1 - Form1 (Code)
Form Load
' İlk noktayı koy ve çizgiye başla
Private Sub Picture1_MouseDown(Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        Picture1.Line (X, Y)-(X, Y)
    End If
End Sub
```

ADIM 4: Fare ile harekete devam.



```
Sketch1 - Form1 (Code)
Form Load
' Fare hareket ettikçe çizgiyi diğer noktaya kadar uzat
Private Sub Picture1_MouseMove(Button As Integer, _
    Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        Picture1.Line -(X, Y)
    End If
End Sub
```

ADIM 5: Sonuç.



Şekil 1.34: Programın çalışması

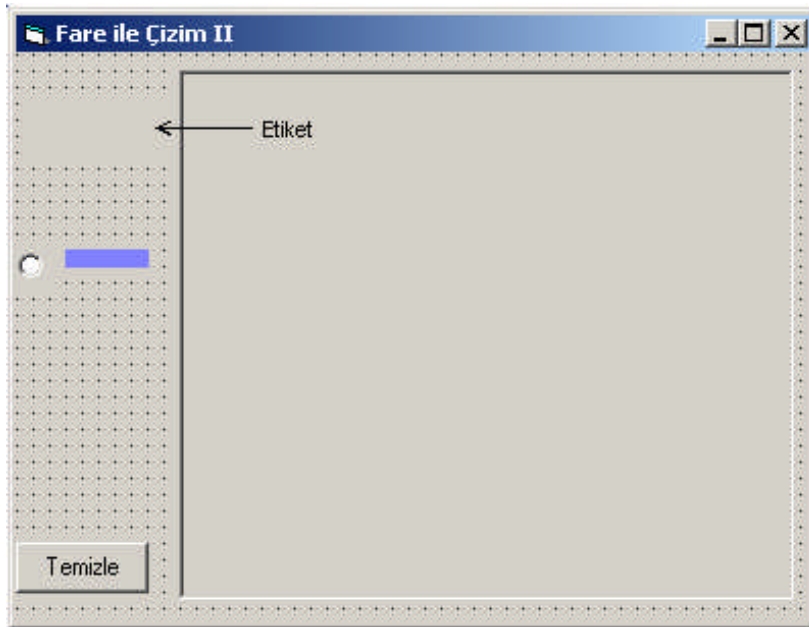
ÖRNEK 1.15: Kullanıcıya çizgi kalınlığını ve rengini deęiřtirme kabiliyetini sunma.

Bir önceki programda çizgi kalınlığı ve rengi kod içinde ayarlanmıştı. Bu örnekte kullanıcının bunları kendisinin seçmesi istenirse;

Burada birkaç nokta dikkate alınacaktır.

- Birincisi bileşenler dinamik olarak formun Load olayında yüklenir. Böylece programcı renk ve kalınlık seçeneklerinin sayısını artırabilir ya da azaltabilir. Bu da bu deęerleri tutan dizide deęişiklik yaparak sağlanır.
- İkincisi dinamik bileşenler program yoluyla konumlandırıldığından istenilen yerde görünmesi sağlanır.
- Üçüncüsü renk tercihlerini sunmak için label bileşeni kullanılır. Border özellięi ile seçildięi ya da seçilmedięi ayarlanır.
- Dördüncüsü pencerenin önüne herhangi bir nesnenin gelmesi ya da yeniden boyutlandırılması resim alanını etkilememektedir.

ADIM 1:Form tasarımı ařaęıdaki tablodan yararlanarak, řekilde görüldüęü gibi yapılır.



Şekil 1.35: Form tasarımı

Kontrol	Özellik	Değer
Form1	Top	1500
	Width	6666
	Height	5333
	Left	1236
	Caption	Fare ile Çizim II
Label	Name	lblRenk
	Index	0
	Caption	
	Top	360
	Width	975
	Height	375
Label	Left	120
	Name	lblCizgiKalinlik
	Index	0
	Caption	
	Top	1445
	Width	615
	Height	135
	Left	360
OptionButton	BackColor	&H00FF8080&
	Index	0
	Name	optCizgiKalinlik
	Caption	
	Top	1445
	Width	255
	Height	255
Button	Left	0
	Index	0
	Name	cmdTemizle
	Caption	Temizle
	Top	4075
Button	Left	120
	Width	975
	Height	375
	Height	375

ADIM 2: Değişken tanımlama.

Option Explicit

Const Had As Integer = 10 'Görüntü alanının sol/sağ/üst/alt köşeleri arasındaki sınır

Const Narin_Bosluk As Integer = 2 'Çizgi kalınlık sembolleri arasındaki yatay boşluk

Const Buyuk_Bosluk As Integer = 10 'Çizgi kalınlık sembollerinin renk sembollerinden ayrıklığı

```
Const RenkCumbusu As Integer = 4 'Görüntülenecek renk sayısı 0..4
Dim Renk(RenkCumbusu + 1) As Long 'Renk paletindeki renkleri tutacak dizi
değişkeni
```

```
Const Kalinlik_Degeri As Integer = 4 'Görüntülenecek çizgi kalınlık sayısı 0..4
Dim CizgiKalinlik(Kalinlik_Degeri + 1) As Integer 'Çizgi kalınlık paletindeki
kalınlıkları 'tutan dizi
```

ADIM 5: Formun Load olayı ve bağlı yordamlar.

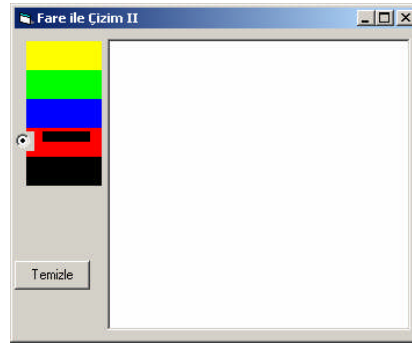
```
Sub Form_Load()
Dim I As Integer
Picture1.AutoRedraw = True
Picture1.BackColor = vbWhite 'Renk kutusunun arka planı beyaz
Picture1.ToolTipText = "Çizim için farenin sol tuşuna basınız"
cmdTemizle.ToolTipText = "Resim alanını temizlemek için tıklayınız"
Form1.ScaleMode = vbPixels 'Tüm ölçüler pixel
End Sub
```

ADIM 6: Form üzerine konulan lblRenk isimli etiketten yararlanarak, renkleri çalışma zamanında gösterecek bir renk paleti oluşturup Formun kod bölümüne bununla ilgili RenkDeryası yordamı eklenir.

```
Sub RenkDeryası()
Dim I As Integer
Renk(0) = vbYellow 'Renk paletini tanımla
Renk(1) = vbGreen
Renk(2) = vbBlue
Renk(3) = vbRed
Renk(4) = vbBlack
lblRenk(0).Left = Had 'İlk rengi sol üst köşeye yerleştirir
lblRenk(0).Top = Had 'Diğer renkler buna göre konumlanır
lblRenk(0).ToolTipText = "Çizim rengi için seçim yapınız"
lblRenk(0).BackColor = Renk(0) 'İlk label bileşeni ilk renge ayarlanır ve
For I = 1 To RenkCumbusu 'Diğer renkler için yeni label bileşenleri oluşturulur.
Load lblRenk(I)
lblRenk(I).BackColor = Renk(I) 'Renkleri göster ve konuşlandır
lblRenk(I).Left = Had
lblRenk(I).Top = lblRenk(I - 1).Top + lblRenk(I - 1).Height 'Renk yüksekliği
lblRenk(I).Visible = True 'Geçerli değer olarak visible özelliği True olsun
Next I
End Sub
```

Çalışma zamanında bileşen yüklemenin en kolay yolu bir bileşen dizisi oluşturmaktır. Bunun için de form üzerine örnek teşkil edecek bir bileşen konular ve index özelliğine 0 atanır. Load komutu ile istenildiği anda bileşen form üzerinde oluşturulur. Kurulacak bir döngü ile bileşen adedi ayarlanır. Yeni bileşenlerin indeks sayıları otomatik olarak 1 arttırılır.

Bu yordamın ismi formun Load olayına yazılır ve program çalıştırılır. Kodun hatalı olup olmadığını denir.



Şekil 1.36: Tasarımın çalışması

Daha önce rastgele bir OptionButton ve lblCizgiKalinlik isimli bir etiket yerleştirildiğinden, şekil istenilen tarza uymadı. Dağınık haldeki bu şekilleri düzene sokmak için;

ADIM 7: CizgiGenislikOlustur isimli yordamla çizgi kalınlığını seçme imkanı sunulur.

'Çizgi kalınlığı paleti

'Çizgi kalınlığı için bir option kutusu ve kalınlığı temsil için bir label bileşeni kullanılır

Sub CizgiGenislikOlustur()

Dim I As Integer

CizgiKalinlik(0) = 1 'Çizgi kalınlıklarını tanımla

CizgiKalinlik(1) = 2

CizgiKalinlik(2) = 4

CizgiKalinlik(3) = 7

CizgiKalinlik(4) = 10

lblCizgiKalinlik(0).ToolTipText = "Çizgi kalınlığı için tıklayınız"

optCizgiKalinlik(0).ToolTipText = lblCizgiKalinlik(0).ToolTipText

For I = 0 To Kalinlik_Degeri

If I > 0 Then

Load optCizgiKalinlik(I) 'Option ve label bileşenlerini yükle

Load lblCizgiKalinlik(I)

optCizgiKalinlik(I).Top = optCizgiKalinlik(I - 1).Top + optCizgiKalinlik(I - 1).Height

+ Narin_Bosluk

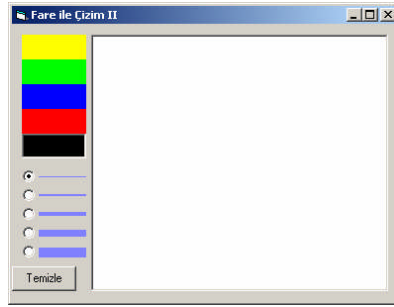
Else

```

    optCizgiKalinlik(I).Top = lblRenk(RenkCumbusu).Top+
lblRenk(RenkCumbusu).Height + Buyuk_Bosluk
    End If
    optCizgiKalinlik(I).Left = Had 'Kontrolleri konuştandı
    optCizgiKalinlik(I).Visible = True
    lblCizgiKalinlik(I).Left = Had
    lblCizgiKalinlik(I).Height = CizgiKalinlik(I)
    lblCizgiKalinlik(I).Width = lblRenk(0).Width
    lblCizgiKalinlik(I).Top = optCizgiKalinlik(I).Top + ((optCizgiKalinlik(I).Height -
CizgiKalinlik(I)) / 2)
    lblCizgiKalinlik(I).Visible = True
    Next I
    End Sub

```

Yordam isminin formun Load olayına eklenmesi unutulmamalıdır. Bir kere daha çalıştırılırsa;



Şekil 1.37: Tasarımın çalışması

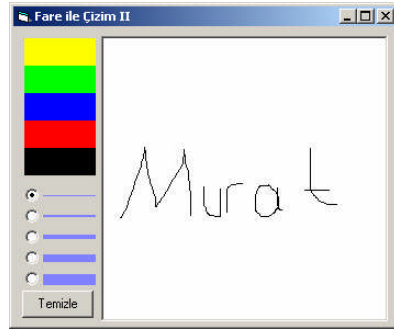
ADIM 8: Program çalışmasına rağmen fare ile çizim yapılamamaktadır. Bu amaçla daha önce gösterilen fare olayları yazılır. İlk noktayı koy ve taslak çizimi başlat

```

Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    If Button = 1 Then
        Picture1.Line (X, Y)-(X, Y)
    End If
End Sub
Fare ile hareket ettikçe çiz
Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    If Button = 1 Then
        Picture1.Line -(X, Y)
    End If
End Sub

```

Program çalıştırılır.



Şekil 1.38: Tasarımın çalışması

ADIM 9: Fareyle çizim yapabilmemize karşılık, renk ve çizgi seçme imkanı yoktur. Önce renk seçmek için, renkleri temsil eden etiketin tıklama olayına aşağıdaki kodlar yazılır.

```
'Renk etiketine tıkladığında renk seç
Sub lblRenk_Click( Sec As Integer)
Dim I As Integer
Picture1.ForeColor = Renk( Sec)
For I = 0 To RenkCumbusu 'Kenar desenlerini kaldır
lblRenk(I).BorderStyle = 0
Next I
lblRenk( Sec).BorderStyle = 1'Kenar desenini değiştirerek seçili görünsün
End Sub
```

Program çalıştırılır.

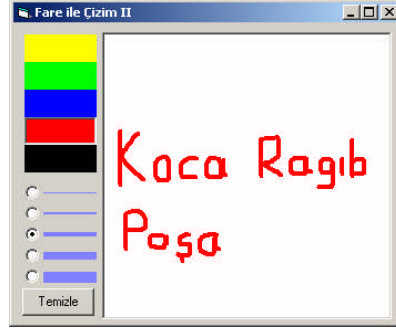


Şekil 1.39: Tasarımın çalışması

ADIM 10: Çizgi kalınlığı seçilerek, aşağıdaki kod ilave edilir.

```
Option düğmesine tıklanınca yeni çizgi kalınlığını ata
Private Sub optCizgiKalinlik_Click( Sec As Integer)
Picture1.DrawWidth = CizgiKalinlik( Sec)
End Sub
```

Program çalıştırıldığında, sadece option düğmesini tıklayarak çizgi kalınlığının değiştirilebildiği görülür.



Şekil 1.40: Programın çalışması

ADIM 11: Çizgi sembolleri tıklanarak kalınlık seçilir.

Çizgi kalınlığını seç

```
Sub lblCizgiKalinlik_Click( Sec As Integer)
```

```
Picture1.DrawWidth = CizgiKalinlik( Sec)
```

```
optCizgiKalinlik( Sec) = True 'Seçili olduğuna emin ol
```

```
End Sub
```

ADIM 12: Formun Load olayının kodu düzenlenerek, varsayılan ayarlamalar yapılır.

```
Sub Form_Load()
```

```
Dim I As Integer
```

```
Picture1.AutoRedraw = True
```

```
Picture1.BackColor = vbWhite 'Renk kutusunun arka planı beyaz
```

```
Picture1.ToolTipText = "Çizim için farenin sol tuşuna basınız"
```

```
cmdTemizle.ToolTipText = "Resim alanını temizlemek için tıklayınız"
```

```
Form1.ScaleMode = vbPixels 'Tüm ölçüler pixel
```

```
RenkDeryası 'Form üzerinde renk ve
```

```
CizgiGenislikOlustur 'kalınlık paleti oluştur
```

```
Paletin başlangıç değerlerini ata
```

```
lblRenk_Click RenkCumbusu 'Çizgiler siyah
```

```
optCizgiKalinlik(0).Value = True 'İlk çizgi kalınlığını seç
```

```
optCizgiKalinlik_Click 0
```

```
End Sub
```

ADIM 13: Formun Resize olayına kod yazarak pencere büyüklüğü ile oynansa bile şekil kaybolmasın.

Form yeniden boyutlandırılrsa dahi çizilen şekli kabul etme

```
Sub Form_Resize()
```

```
Picture1.Width = Form1.ScaleWidth - Picture1.Left - Had
```

```
Picture1.Top = Had
```

```
Picture1.Height = Form1.ScaleHeight - Picture1.Top - Had
```

```
cmdTemizle.Top = Form1.ScaleHeight - cmdTemizle.Height - Had
```

```
End Sub
```

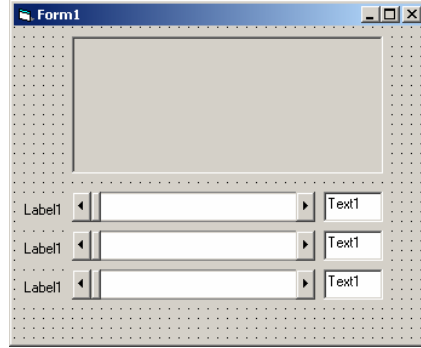

ADIM 14: Temizle başlıklı düğme tıklandığında PictureBox temizlensin.

```
Çizim alanını temizle  
Private Sub cmdTemizle_Click()  
Picture1.Cls  
End Sub
```

ADIM 15: Program tekrar çalıştırılır.

ÖRNEK 1.16: RGB renkleri kaydırma çubukları ile ayarlanır.

ADIM 1: Form tasarımı.



Şekil 1.41: Form tasarımı

Burada Label, HscrollBar ve Text bileşenleri bir dizi olarak tanımlanacaktır.

ADIM 2: Formun Load olayı

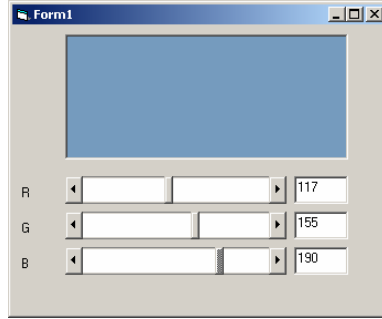
```
Private Sub Form_Load()  
Picture1.BackColor = RGB(0, 0, 0)  
Label1(0).Caption = "R"  
Label1(1).Caption = "G"  
Label1(2).Caption = "B"  
HScroll1(0).Max = 255  
HScroll1(1).Max = 255  
HScroll1(2).Max = 255  
End Sub
```

ADIM 3: Kaydırma çubuklarıyla oynandıkça renk değerleri değişsin.

```
Private Sub HScroll1_Change(Index As Integer)  
Dim R As Byte '(0~255)  
Dim G As Byte '(0~255)  
Dim B As Byte '(0~255)  
R = HScroll1(0).Value
```

```
G = HScroll1(1).Value
B = HScroll1(2).Value
Text1(0).Text = R
Text1(1).Text = G
Text1(2).Text = B
Picture1.BackColor = RGB(R, G, B)
End Sub
```

ADIM 4: Program çalıştırılır.



Şekil 1.42: Programın çalışması

Bir de teknikte CMY renk modeli vardır.

CMY modelinde harfler C:'Cyan' (Camgöbeği), M:'Magenta' (Eflatun), Y:'Yellow' (Sarı) anlamına gelir ve bunların üçü boya veya mürekkep olarak karıştırıldığında siyah oluşur. Bu model, matbaa makineleri, yazıcı gibi baskı yapan cihazlarda kullanılır. RGB modeli gibi ışınların karışımından değil mürekkeplerin karışımlarından oluşan renkleri ifade eder.

ADIM 5: CMY renklerini kullanarak, aşağıdaki kod değişikliği yapılır. Program çalıştırılıp aradaki fark gözlenir.

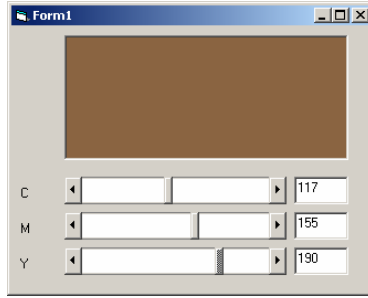
```
Private Sub Form_Load()
Picture1.BackColor = RGB(255, 255, 255)
Label1(0).Caption = "C"
Label1(1).Caption = "M"
Label1(2).Caption = "Y"
HScroll1(0).Max = 255
HScroll1(1).Max = 255
HScroll1(2).Max = 255
End Sub
Private Sub HScroll1_Change(Index As Integer)
Dim R As Byte ' (0~255)
Dim G As Byte ' (0~255)
Dim B As Byte ' (0~255)
Dim C As Byte ' (0~255)
```

```
Dim M As Byte ' (0~255)
Dim Y As Byte ' (0~255)
```

```
C = HScroll1(0).Value
M = HScroll1(1).Value
Y = HScroll1(2).Value
Text1(0).Text = C
Text1(1).Text = M
Text1(2).Text = Y
```

```
R = 255 - C
G = 255 - M
B = 255 - Y
Picture1.BackColor = RGB(R, G, B)
End Sub
```

Program çalıştırılır.



Şekil 1.43: Programın çalışması

1.3.5 Resimleri Saklamak

LoadPicture ile sabitdiskte bulunan bir resmi PictureBox içine getiriyorduk. PictureBox içindeki bir resim ise sabitdiske SavePicture metodu kullanılarak kaydedilir. Kullanımı:

SavePicture picture yada image kontrol, grafik dosyanın adı

SavePicture, PictureBox ya da image kontrolde yüklü bulunan bmp, ico ve wmf dosyalarını kendi formatlarında saklar. Eğer dosya jpeg yada gif formatında ise bmp formatına dönüştürür. Aşağıdaki kod, Picture1 isimli resim kutusunda bulunan resmi sabitdiske kaydetmektedir.

```
Private Sub Command1_Click()
SavePicture Picture1.Picture, "c:\TEST.BMP"
End Sub
```

Picture kontrolü içine yapılan çizimleri temsil etmek için PictureBox'ın sadece okunabilir olan Image isimli bir özelliği daha vardır. Line, Circle gibi metotlar kullanılarak elde edilen çizimler Picture özelliği yerine Image özelliğinde tutulur.

ÖRNEK 1.18: İç içe daireler çizme.

ADIM 1: Form üzerine bir PictureBox yerleştirilir.

ADIM 2: Aşağıdaki kod yazılır.

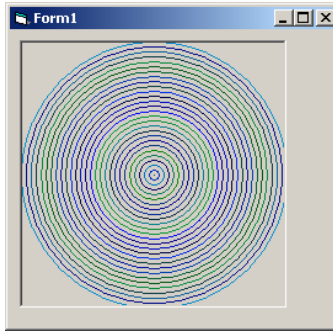
```
Dim CX, CY, Had, Yaricap As Integer, Peyam As String
Private Sub Form_Load()
With Picture1
ScaleMode = vbPixels
AutoRedraw = True
Width = .Height ' En ile boyu eşitle
CX = .ScaleWidth / 2 ' Daire merkezinin X koordinatı.
CY = .ScaleHeight / 2 ' Daire merkezinin Y koordinatı
End With

Had = CX ' Daire çapına sınırlama getir.
For Yaricap = 0 To Had / 4
Picture1.Circle (CX, CY), Yaricap * 4, RGB(Rnd * 15, Rnd * 155, Rnd * 242)
Next Yaricap

Peyam = "Resmi bmp uzantılı kaydetmek "
Peyam = Peyam & "için onaylayın"
MsgBox Peyam
SavePicture Picture1.Image, "c:\TEST.BMP"
End Sub

Private Sub Form_Unload(Cancel As Integer)
MsgBox "Resminiz kaydedildi"
End Sub
```

ADIM 3: Program çalıştırılır.

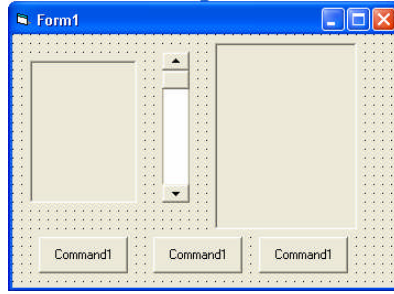


Şekil 1.44: Programın çalışması

Formun Unload olayına da bir mesaj yazarak kullanıcıya bir hatırlatma yapılmaktadır. Program dahilinde rastgele bir renk eldesi için Rnd fonksiyonu kullanılmaktadır. Dairelerin arasını açmak için yarıçap çarpma ve bölme işlemine tabi tutulmaktadır.

ÖRNEK 1.19: Çizilen rastgele bir çizgiyi döndürme.

ADIM 1: Form üzerine iki picturebox, bir dizi olarak tanımlanmış üç düğme ve dikey bir kaydırma çubuğu yerleştirilir.



Şekil 1.45: Form tasarımı

ADIM 2: Kod listesi aşağıda verilmiştir.

```
Private renk As Long  
Private ang As Single  
Dim BasOran As Single
```

```
Private Sub cmdPatrona_Click(Index As Integer)  
Select Case Index  
Case 0  
Picture1.Cls  
Picture2.Cls  
Case 1  
Dondur  
Case 2  
End  
End Select
```

```
End Sub  
Private Sub Form_Load()  
cmdPatrona(0).Caption = "Sil"  
cmdPatrona(1).Caption = "Döndür"  
cmdPatrona(2).Caption = "Çık"  
Picture1.Scale (0, 0)-(100, 100)  
Picture2.Scale (-100, -20)-(200, 220)  
Picture1.BackColor = QBColor(15)  
Picture2.BackColor = QBColor(15)  
renk = QBColor(0)
```

```
VScroll1.Max = 90
VScroll1.Min = 0
ang = 20
VScroll1.Value = ang
End Sub
```

```
Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As Single,
Y As Single)
Picture1.PSet (X, Y), renk
End Sub
```

```
Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y
As Single)
If Button = 1 Then Picture1.Line -(X, Y), renk
End Sub
```

```
Sub Dondur()
radian = (2 * 3.1416 * ang) / 360
For py = 0 To 100 Step 2
For px = 0 To 98
scolor = Picture1.Point(px, py)
If scolor = renk Then
z = py * Cos(radian)
BasOran = Sin(radian)
Picture2.Circle (0, z), 2 * px, QBColor(12), , , BasOran
Exit For
End If
Next px
Next py
End Sub
```

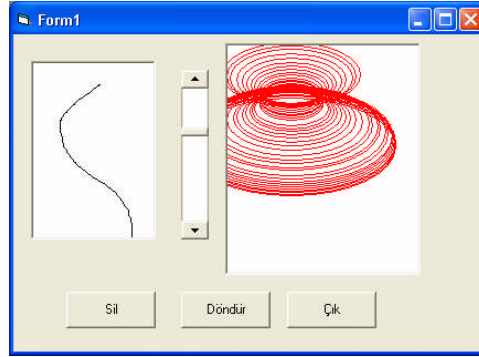
```
Private Sub VScroll1_Change()
ang = VScroll1.Value
Picture2.Cls
Dondur
End Sub
```

```
Burada
radian = (2 * 3.1416 * ang) / 360
satırı ile gelen radyan bilgisi,
z = py * Cos(radian)
satırı ile hesaplanmaktadır. Bu değer çemberin yarıçapıdır.
BasOran = Sin(radian)
```

satırından hesaplanan değer, basıklık oranıdır ve

```
Picture2.Circle (0, z), 2 * px, QBColor(12), , , BasOran
```

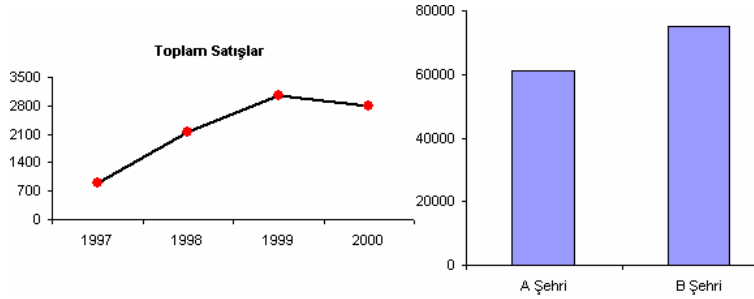
satırında yerine konulduğunda, bir perspektif görüntüsü vermektedir. Programın ekran çıktısı aşağıda görülmektedir.



Şekil 1.46: Programın çalışması

1.4. Çizgeler

Çizgeler, sayısal verilerin kullanıcıya daha görkemli olarak sunulmasıdır. Verilerin artma ve azalma eğilimleri ya da diğerleri ile özellikle de zamanla olan bağlantıları çizgeler yardımıyla daha kolay anlaşılır.



Şekil 1.47: Çizelge Örnekleri

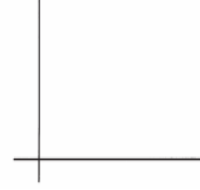
Her çizelgenin çizimi üç aşamayı gerektir:

- Koordinat sisteminin tanımı
- Grafiği çizmek için uygun geometrik şekilleri kullanmak
- Çizelgenin uygun alanlarına yazı yerleştirmek

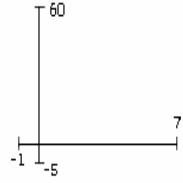
1.4.1. Bir Koordinat Sistemi Tanımlamak

Kağıt üzerinde (2,45) ve (6,55) noktaları arasında cetvelle çizgi çizmek için herkesin kendine göre bir yöntemi olabileceği gibi, genellikle aşağıdaki üç madde daha çok tercih edilir.

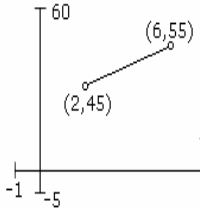
➤ Cetvelle x ve y eksenleri çizilir.
Ölçüler pozitif olduğundan sadece ilk çeyreği çizmek yeterlidir.



➤ Her iki eksen için de ölçek tayin edilir.
Örnek olarak x eksenini için -1...7, y eksenini için -5...60 sayıları tayin edilebilir.



➤ İki nokta işaretlenerek cetvelle çizgi dosdoğru çizilir.



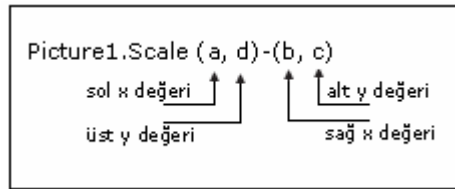
Şekil 1.48: Koordinat sistemi

İlkokuldan beri bilinen bir konu, resim kutusunda yeni bir koordinatın nasıl tanımlanacağını gözünüzde canlandırmanız için anlatılmıştır.

Ekran üzerinde de çizim yapmak için bu üç adım uygulanır. Tek fark bilgisayarın kendine özgü işleyişinden dolayı önce ikinci adım, sonra birinci ve üçüncü adım uygulanır. VB'nin Scale metodu x ve y aralıklarını tanımlamak için, Line metodu da cetvelin yerine çizgiyi çizmek için kullanılır.

Picture1.Scale (a, d)-(b, c) ifadesi resim kutusunda x eksenini için (a, b) ve y eksenini için (c,d) noktalarını tanımlar.

Burada aralıklar:



Şekil 1.49: Scale metodu

1.4.2. PictureBox'ta Yazı Yazma

Ekran üzerine yazılan yazının grafikte bir arada olması istenen zamanlar olabilir. Bu grafiğe bir başlık atıldığında yada eksenlere bir isim verildiğinde olabilir. Bu şekilde metnin şekille uyumlu olarak resim kutusunda yer alması grafiğe ayrı bir güzellik verir. Yazı yazmak için Print metodu kullanılır. Metnin konumu da CurrentX ve CurrentY özellikleri ile belirlenir.

PictureBox, varsayılan koordinat ölçüsü olarak twip kullandığından ScaleMode özelliğini vbPixels yapmak gerekir. Daha sonra metnin konumlanacağı mutlak yer belirtilir.

```
Private Sub Form_Load()  
Picture1.ScaleMode = vbPixels  
Picture1.CurrentX = 23  
Picture1.CurrentY = 20  
.....  
.....  
End Sub
```

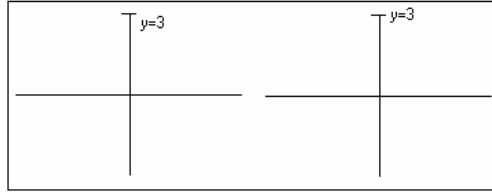
Son olarak Print metodu ile arzu edilen yazı yazılır.

```
Private Sub Form_Load()  
Picture1.ScaleMode = vbPixels  
Picture1.CurrentX = 23  
Picture1.CurrentY = 20  
Picture1.Print ("Bu bir denemedir ")  
End Sub
```

Program yazıp çalıştırıldığında, emeklerinizin boşa gittiği görülür. Yazı ortalıkta görünmemektedir. Bir de PictureBox'ın AutoRedraw özelliği True yapılarak denenir.

TextHeight ve TextWidth özellikleri, bize yazıları grafik boyunca hassas yerleştirmemizi sağlar. Harflerin yüksekliğini ve genişliğini öğrenmek için kullanılır. CurrentX ve CurrentY özellikleri, aynen çizgi çizmede olduğu gibi, yazılan metnin bir sonraki karakterinin basılacağı konumun yatay ve düşey konumlarını kaydeder. Aşağıdaki yordam bunun tipik bir örneğidir.

```
Sub Ciz()  
Picture1.Cls  
Picture1.Scale (-4, 4)-(-4, -4) 'Koordinat tanımlaması  
Picture1.Line (-3, 0)-(-3, 0) 'X eksen  
Picture1.Line (0, -3)-(0, 3) 'y eksen  
Picture1.Line (-0.2, 3)-(0.2, 3) 'İşaret çizgisi  
Picture1.CurrentX = 0.3 'İşaret çizgisinin sağ ucu  
Picture1.CurrentY = 3 'İşaret çizgisinin üst ucu  
Picture1.Print "y=3" 'İşaret çizgisinin yazısı  
End Sub
```



Şekil 1.50 : Harf yerleştirme (a) Yanlış, (b) Doğru

Y ekseninin üstündeki yazının üst kesimi ile işaret çizgisi, aynı hizada. Bu da demek oluyor ki CurrentY değeri karakterin üst kısmına ayarlı. Bu yüzden işaret çizgisi ile aynı hizaya gelmesi için harfin yüksekliğinin yarısı kadar yukarıya taşınması gerekir. Bu yüzden CurrentY değeri harf yüksekliğinin yarısı kadar artırılmalıdır.

Picture1.CurrentY = 3 - Picture1.TextHeight("y=3") / 2

Örneğe bakacak olursanız şaşırmanın. Yukarıda artırılmalıdır denildi. İfadede çıkarma işlemi var. Y ekseninin kartezyen koordinatlara göre ters olduğunu hatırlayınız.

Düzeltilmiş yükseklik Şekil 1.43(b) de görülmektedir.

Aşağıdaki yordamda ise resim kutusunun ortasına altı çift çizgili bir cümle yerleştirilmektedir.

Sub Ciz()

Picture1.Scale (0, 4)-(6, 0)

Picture1.Cls

xMerkez = 3

yMerkez = 2

cumle = "Avazeyi bu aleme Davut gibi sal "

w = Picture1.TextWidth(cumle)

h = Picture1.TextHeight(" ")

Picture1.CurrentX = xMerkez - w / 2

Picture1.CurrentY = yMerkez - h / 2

Picture1.Print cumle

SolKenar = xMerkez - w / 2

SagKenar = xMerkez + w / 2

CizgiKonum1 = yMerkez + h / 2

CizgiKonum2 = CizgiKonum1 + h / 6

Picture1.Line (SolKenar, CizgiKonum1)-(SagKenar, CizgiKonum1)

Picture1.Line (SolKenar, CizgiKonum2)-(SagKenar, CizgiKonum2)

End Sub

Program çalıştırılır.



Şekil 1.51: Programın çalışması

Cümlelerin merkezlenmesi resim kutusunun merkezinin bilinmesini gerektirir. Bu da örnekte (3,2) noktasıdır. Sonra merkezlenecek harflerin yüksekliğinin ve genişliğinin bilinmesi gerekir. İlk harfin yazılacağı yer, merkez noktadan cümle genişliğinin yarısı kadar sol taraf (CurrentX) ve harf yüksekliğinin yarısı kadar merkez noktanın üstüdür. İlk alt çizgi, karakter yüksekliğinin yarısı kadar, ikincisi de buna benzer bir oranlama ile (burada 6) merkeze ilave edilerek çizilir.

Metne bir biçim vermek için FontBold, FontItalic, FontStrikethru ve FontUnderline özellikleri kullanılır. Bunun için, aşağıdaki kodlar ilave edilir.

```
dize = "Baki kalan bu kubbede hoş bir sada imiş"  
w = Picture1.TextWidth(dize)  
h = Picture1.TextHeight(" ")  
Picture1.CurrentX = xMerkez - w / 2  
Picture1.CurrentY = yMerkez + h  
Picture1.FontName = "Arial"  
Picture1.FontUnderline = True  
Picture1.FontBold = True  
Picture1.Print dize
```

1.4.3. Çizgi Çizelgesi (Line Chart)

Çizgi Çizelgesi, veriler arası ilişkiyi çizgilerle gösterir. Aşağıdaki adımlar takip edilir.

- Gösterilecek verilere bakılır. Genelde aylık ya da yıllık zamana göre veriler verilir. Aylık satışlar, kâr-zarar gibi. Zaman birimleri x eksenine yerleştirilir.
- Veri sayısına ve verilerin büyüklüğüne göre bir koordinat sistemi seçilir. X eksenini -1 değerinden veri sayısının bir fazlası olarak seçilir. Y ölçeği ise en büyük veri değerine göre ayarlanır.
- Çizgi parçaları çizilerek, veri sonlarına çemberin çizilmesi gereklidir.

- Eksenlere işaret çizgileri çizilir. X ekseninin her zaman için bir çizgi çizilir. Y ekseninde ise en azından en büyük niceliği gösteren bir çizgi olmalıdır.

ÖRNEK 1.28: Bir firmanın 5 yıllık toplam gömlek satışları Tablo 1.11’de görülmektedir. Toplam satışları gösteren çizelgeyi çizelim.

5 Yıllık Gömlek Satışları					
Yıl	1997	1998	1999	2000	2001
Toplam	853	2320	4526	5240	5143

Tablo 1.11 Veri tablosu

ADIM 1: Tablodaki toplam satışlar ya bir diziyeye aktarılır ya da bir metin dosyasına yazılır. Dosyaya yazıldığında, program bu dosyadan toplam satışları okuyacaktır. C dizisinde “Satis.txt” dosyası içine veriler aşağıdaki şekilde kaydedilir.

```
"1997",853  
"1998",2320  
"1999",4526  
"2000",5240  
"2001",5143
```

ADIM 2: Form üzerine bir picturebox ve “cmdCiz” isimli bir düğme yerleştirilir.

ADIM 3: Kod listesi.

```
Dim Devri_Sene As Integer, ToplamSatis As Single
```

```
Private Sub EksenCiz()  
Picture1.Scale (-1, 1.2 * ToplamSatis)-(Devri_Sene + 1, -0.2 * ToplamSatis)  
Picture1.Line (-1, 0)-(Devri_Sene + 1, 0)  
Picture1.Line (0, -0.1 * ToplamSatis)-(0, 1.1 * ToplamSatis)  
End Sub
```

```
Private Sub GrafikCiz(toplam() As Single)  
Dim i As Integer  
'Veri ile bağlantılı çizgi çizgi ve daire çiz  
For i = 1 To Devri_Sene  
If i < Devri_Sene Then  
Picture1.Line (i, toplam(i))-(i + 1, toplam(i + 1))  
End If  
Picture1.Circle (i, toplam(i)), 0.01 * Devri_Sene, QBColor(12)  
Next i  
End Sub
```

```
Private Sub Konum(x As Single, y As Single)  
Picture1.CurrentX = x  
Picture1.CurrentY = y  
End Sub
```

```
Private Sub VeriOku(etiket() As String, toplam() As Single)
Dim i As Integer
'Veriler Satis.txt dosyasında. İlk satırı "1997",853
'Verileri dizilere aktar, en yüksek rakamı bul
ToplamSatis = 0
```

```
Open "c:\Satis.txt" For Input As #1
For i = 1 To Devri_Sene
Input #1, etiket(i), toplam(i)
If toplam(i) > ToplamSatis Then
ToplamSatis = toplam(i)
End If
Next i
Close #1
End Sub
```

```
Private Sub EtiketGoster(etiket() As String)
Dim i As Integer, Baki As String, EtiketGen As Single
Dim EtiketYuk As Single, İsaretCizgiFak As Single
'İşaret çizgilerini çiz ve etiket yaz
For i = 1 To Devri_Sene
Baki = etiket(i)
EtiketGen = Picture1.TextWidth(Baki)
İsaretCizgiFak = 0.02 * ToplamSatis
Picture1.Line (i, -İsaretCizgiFak)-(i, İsaretCizgiFak)
Call Konum(i - EtiketGen / 2, -İsaretCizgiFak)
Picture1.Print Baki
Next i
```

```
Baki = Str(ToplamSatis)
EtiketGen = Picture1.TextWidth(Baki)
EtiketYuk = Picture1.TextHeight(Baki)
İsaretCizgiFak = 0.02 * Devri_Sene
Picture1.Line (-İsaretCizgiFak, ToplamSatis)-(-İsaretCizgiFak, ToplamSatis)
Call Konum(-İsaretCizgiFak - EtiketGen, ToplamSatis - EtiketYuk / 2)
Picture1.Print Baki
End Sub
```

```
Private Sub ShowTitle()
'Başlık göster
Call Konum(0.5, 1.2 * ToplamSatis)
Picture1.Print "4 Yıllık Gömlek Satışları"
End Sub
```

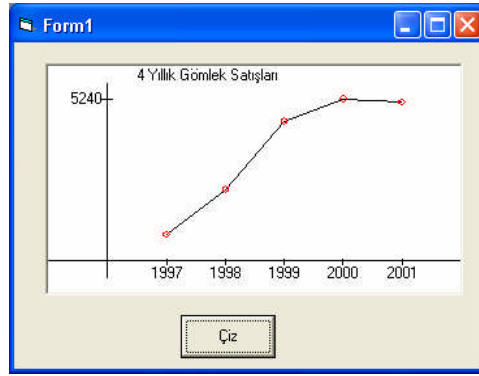
```
Private Sub cmdCiz_Click()
'Toplam gömlek satışlarının çizelgesi
Devri_Sene = 5
```

```

ReDim etiket(1 To Devri_Sene) As String
ReDim toplam(1 To Devri_Sene) As Single
Call VeriOku(etiket(), toplam())
Call EksenCiz
Call GrafikCiz(toplam())
Call ShowTitle
Call EtiketGoster(etiket())
End Sub

```

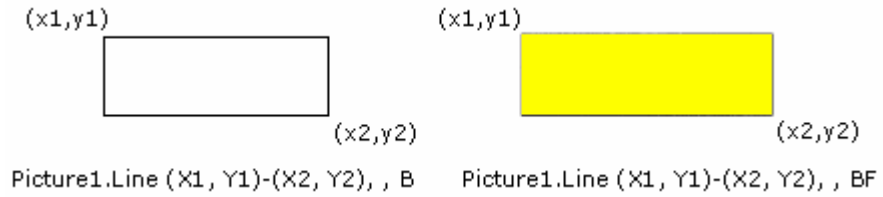
ADIM 4: Program çalıştırılır.



Şekil 1.52: Programın çalışması

1.4.4. Çubuk Çizmelere

Çubuk çizimlerinde içi boş ve dolu kutular Line metodu ile çizilir.



Şekil 1.53: Çubuk çizme

ÖRNEK 1.29: Tablo 1.12' de toplam gömlek satışlarının profili çıkarılmıştır.

5 Yıllık Gömlek Satışları					
Yıl	1997	1998	1999	2000	2001
Uzun Kollu	400	1100	3000	3699	2999
Kısa Kollu	453	1220	1526	1541	2144
Toplam	853	2320	4526	5240	5143

Tablo 1.12: Veri tablosu

Bu satış profili yine c dizinine “satis1.txt” dosyasına kaydedilir.

```
"1997",400,453  
"1998",1100,1220  
"1999",3000,1526  
"2000",3699,1541  
"2001",2999,2144
```

ADIM 2: Form üzerine bir PictureBox ve “cmdCiz” isimli bir düğme yerleştirilir.

ADIM 3: Kod listesi.

```
Private Sub cmdCiz_Click()
```

```
Dim Devri_Sene As Integer, ToplamSatis As Single
```

```
Devri_Sene = 5
```

```
ReDim etiket(1 To Devri_Sene) As String
```

```
ReDim UKollu(1 To Devri_Sene) As Single
```

```
ReDim KisaKollu(1 To Devri_Sene) As Single
```

```
Call VeriOku(etiket(), UKollu(), KisaKollu(), Devri_Sene, ToplamSatis)
```

```
Call EksenCiz(Devri_Sene, ToplamSatis)
```

```
Call VeriCiz(UKollu(), KisaKollu(), Devri_Sene)
```

```
Call BaslikGoster(ToplamSatis)
```

```
Call EtiketGoster(etiket(), Devri_Sene, ToplamSatis)
```

```
Call EvsafGoster(ToplamSatis)
```

```
End Sub
```

```
Private Sub EksenCiz(Devri_Sene As Integer, ToplamSatis As Single)
```

```
Picture1.Scale (-1, 1.2 * ToplamSatis)-(Devri_Sene + 1, -0.2 * ToplamSatis)
```

```
Picture1.Line (-1, 0)-(Devri_Sene + 1, 0)
```

```
Picture1.Line (0, -0.1 * ToplamSatis)-(0, 1.1 * ToplamSatis)
```

```
End Sub
```

```
Private Sub VeriCiz(UKollu() As Single, KisaKollu() As Single, Devri_Sene As  
Integer)
```

```
Dim i As Integer
```

```
'Dikdörtgen Çiz
```

```
For i = 1 To Devri_Sene
```

```
Picture1.Line (i - 0.3, UKollu(i))-(i, 0), , BF
```

```
Picture1.Line (i, KisaKollu(i))-(i + 0.3, 0), , B
```

```
Next i
```

```
End Sub
```

```
Private Sub Locate(x As Single, y As Single)
```

```
Picture1.CurrentX = x
```

```
Picture1.CurrentY = y
```

```
End Sub
```

```
Private Sub VeriOku(etiket() As String, UKollu() As Single, KisaKollu() As Single, _  
Devri_Sene As Integer, ToplamSatis As Single)  
Dim i As Integer
```

```
'Verileri diziye aktar ve en yüksek rakamı bul  
Open "c:\Satis1.txt" For Input As #1
```

```
ToplamSatis = 0
```

```
For i = 1 To Devri_Sene  
Input #1, etiket(i), UKollu(i), KisaKollu(i)
```

```
If UKollu(i) > ToplamSatis Then  
ToplamSatis = UKollu(i)  
End If
```

```
If KisaKollu(i) > ToplamSatis Then  
ToplamSatis = KisaKollu(i)  
End If  
Next i
```

```
Close #1  
End Sub
```

```
Private Sub EtiketGoster(etiket() As String, Devri_Sene As Integer, ToplamSatis As  
Single)
```

```
Dim i As Integer, Fuzuli As String, YazGenislik As Single  
Dim YazYukseklk As Single, CizgiFaktor As Single
```

```
'İşaret çizgilerini çiz ve isim ver
```

```
For i = 1 To Devri_Sene  
Fuzuli = etiket(i)  
YazGenislik = Picture1.TextWidth(Fuzuli)  
CizgiFaktor = 0.02 * ToplamSatis  
Picture1.Line (i, -CizgiFaktor)-(i, CizgiFaktor)  
Call Locate(i - YazGenislik / 2, -CizgiFaktor)  
Picture1.Print Fuzuli  
Next i  
Fuzuli = Str(ToplamSatis)
```

```
YazGenislik = Picture1.TextWidth(Fuzuli)  
YazYukseklk = Picture1.TextHeight(Fuzuli)
```

```
CizgiFaktor = 0.01 * Devri_Sene  
Picture1.Line (-CizgiFaktor, ToplamSatis)-(CizgiFaktor, ToplamSatis)  
Call Locate(-CizgiFaktor - YazGenislik, ToplamSatis - YazYukseklk / 2)
```



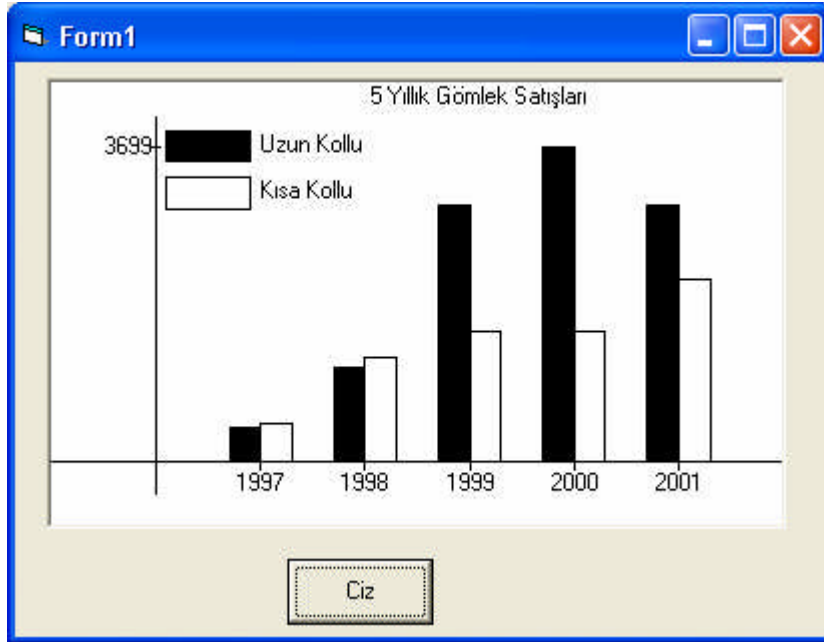
```
Picture1.Print Fuzuli  
End Sub
```

```
Private Sub EvsafGoster(ToplamSatis As Single) 'Evsaf göster  
Picture1.Line (0.1, 1.05 * ToplamSatis)-(0.9, 0.95 * ToplamSatis), , BF  
Call Locate(1, 1.05 * ToplamSatis)  
Picture1.Print "Uzun Kollu"
```

```
Picture1.Line (0.1, 0.9 * ToplamSatis)-(0.9, 0.8 * ToplamSatis), , B  
Call Locate(1, 0.9 * ToplamSatis)  
Picture1.Print "Kısa Kollu"  
End Sub
```

```
Private Sub BaslikGoster(ToplamSatis As Single)  
Call Locate(2, 1.2 * ToplamSatis)  
Picture1.Print " 5 Yıllık Gömlek Satışları"  
End Sub
```

ADIM 4: Program çalıştırılır.

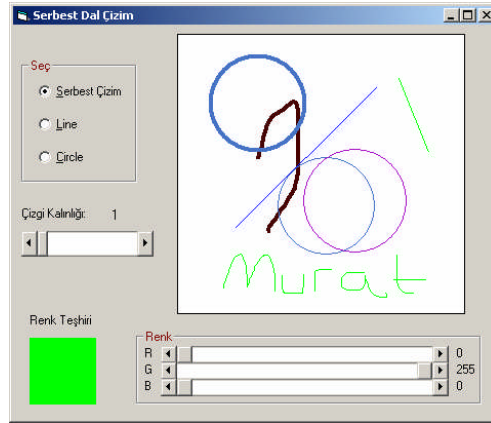


Şekil 1.54: Programın çalışması

UYGULAMA FAALİYETİ

Aşağıdaki sorulara ilişkin uygulama faaliyetini yapınız.

- Örnek 1.6'da Line nesnesi ile yapılan analog saat örneğini Line metodunu kullanarak yapın.
- Aşağıda tasarımı verilen programda fare ile çizim yapılması istenmektedir.



Şekil 1.55: Form tasarımı

Burada programdan istenenler:

- Line, Circle ve serbest tarzda çizim seçeneğini kullanıcıya sunmalıdır.
- Yatay bir kaydırma çubuğu ile çizgi kalınlığı ayarlanmalıdır.
- Çizim rengi RGB'yi temsil eden üç kaydırma çubuğu ile seçilmelidir.
- Kaydırma çubukları ile ayarlanan renk bir PictureBox içinde gösterilmelidir. Bu şartlara uygun olarak programı yazınız.

İşlem Basamakları	Öneriler
<ul style="list-style-type: none">➤ Şekilleri form üzerine yerleştiriniz➤ Hangi olayları kullanacağınıza karar veriniz.➤ Döngü tipine iyi karar veriniz.➤ Yazdığınız programı çalıştırınız.➤ Programda hata var ise bunları gideriniz.	<ul style="list-style-type: none">➤ Nesne özelliklerine uygun değerler atayınız.➤ Hangi değişken tipini kullanacağınıza dikkat ediniz.➤ Değişken artım oranlarını iyi ayarlayınız.➤ Program satırlarının düzenli olmasına özen gösteriniz.➤ Programı çalıştırmadan önce mutlaka kaydediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları cevaplayarak bu faaliyette kazandığınız bilgileri ölçünüz.

OBJEKTİF TEST (ÖLÇME SORULARI)

1. Aşağıdakilerden hangisi Windows'un en küçük grafiksel birimidir?
A) Pixel
B) Twip
C) Character
D) Font
2. Aşağıdakilerden hangisi Shape bileşeninin Shape özelliklerinden değildir?
A) Oval
B) Rounded Rectangle
C) Rectangle
D) Triangle
3. Aşağıdakilerden hangisi grafik metotlarından biri değildir?
A) Pset;
B) Print;
C) Pixel;
D) Cls;
4. Aşağıdakilerden hangisi formun iç çalışma genişliğini temsil eder?
A) ScaleHeight
B) ScaleWidth;
C) Width;
D) Height;
5. Yeni bir koordinat sistemi tanımlayan metot aşağıdakilerden hangisidir ?
A) ScaleMode
B) CurrentX
C) Scale
D) Set

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyete geri dönerek tekrar inceleyiniz.

ÖĞRENME FAALİYETİ-2

AMAÇ

Görsel programlamada API fonksiyonlarını doğru olarak kullanabileceksiniz.

ARAŞTIRMA

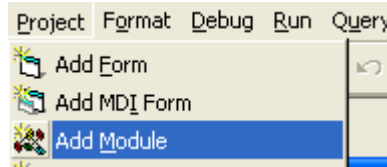
Bu öğrenme faaliyetinden önce aşağıdaki hazırlıkları yapmalısınız.

- Dizi ve dinamik dizi kavramları gözden geçirilmelidir.
- DirectX kütüphanesi hakkında bilgi edininiz.
- Resim dosya formatlarını araştırınız.

2. API İLE GRAFİK

API(Application Programming Interface), Windows fonksiyonlarından oluşan bir kütüphanedir. Microsoft Windows'u yazdığı zaman, belirli işleri yapacak özel kodları programcıların erişebilmesi için kütüphanelerin içine gömmüştür. Hangi dilde olursa olsun Windows üzerinde uygulama yapan tüm programların başvurduğu kullanıma hazır bir kaynaktır. Bu fonksiyonlar Windows'un System32 klasörü içinde bulunan "dll" uzantılı dosyalar içine gömülmüştür.

API fonksiyonları program içinde kullanabilmek için formun "Declarations" bölümünde yada projeye bir modül ekleyip orada tanımlamak gerekir.



Şekil 2.1 : Projeye modül ekleme

Bu tanımlamada API fonksiyonunun nerede olduğu ve fonksiyon ve alt yordam da olduğu gibi hangi parametrelerle çalışacağı belirtilmelidir.

VB' de oluşturulan bir fonksiyonun başlığı şöyledir.

Function Buhara(Byval Semerkant as Long) As Long

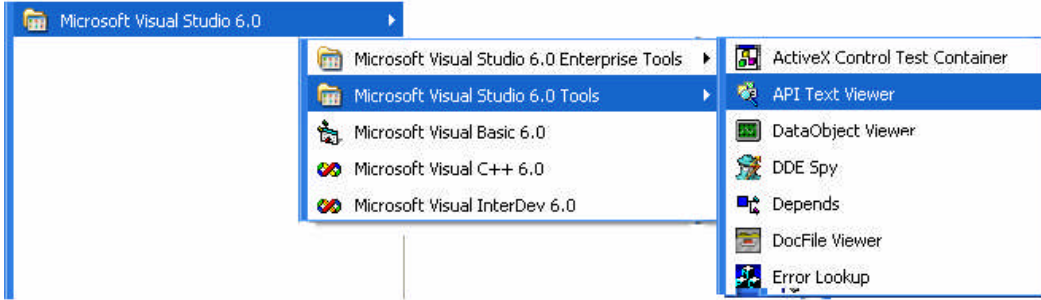
Aynı isimde bir API tanımlaması ise şöyledir.

Declare Function Buhara Lib "Isfehan.dll" (Byval Semerkant As Long) As Long

Aralarındaki temel fark API fonksiyonunun başına Declare ifadesi gelmiştir. Tanımlanması zorunludur. Buhara isimli API'nin hangi kütüphanede bulunacağı Lib ifadesi ile bildirilmektedir(burada Isfehan.dll).

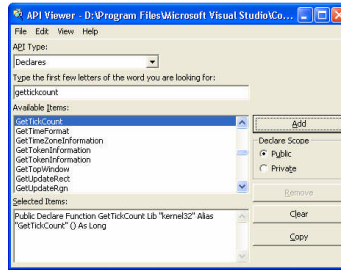
2.1. Api Text Viewer

VB dahilinde API kullanabilmek için API fonksiyonlarının tüm parametrelerinin bilinmesi gerekir. Bunun için VB ile gelen API Text Viewer'dan yararlanır.



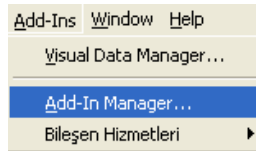
Şekil 2.2 : API Text Viewer

Program açıldığı zaman API'lerin bulunduğu Win32API.txt dosyasının File menüsünden açılması gerekir. API Viewer, programda kullanabilmek için tüm tanımlamaları bize göstermektedir. İlgili API bulunduktan sonra "Add" düğmesi ile "Selected Items" kutusuna alınır ve "Copy" düğmesi ile kopyalanır.



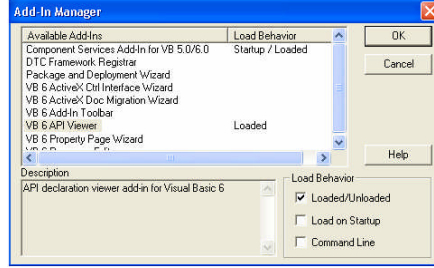
Şekil 2.3 : API Viewer API ekleme

API Text Viwer ile gelen dosyada 1594 API bildirimi vardır.API Text Viewer'a ulaşmanın bir diğer yolu da VB'de Add-Ins menüsünden Add-In Manager tıklanır.



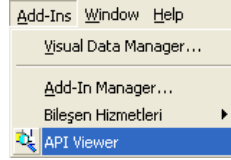
Şekil 2.4: API Viewer ulaşmanın diğer yolu Add-In Manager

Burada VB 6 API Viwer bulunur “Loaded/UnLoaded” onay kutusu işaretlenir.



Şekil 2.5 : Add-In Manager

Pencere kapatıldığında API Viwer menüde yerini alacaktır.



Şekil 2.6 : Menude API Viewer

2.2. Apı Çeşitleri

DLL	Fonksiyonlar
Comdlg32.dll	Dialog pencereleri ilgili API fonksiyonları
Gdi32.dll	Grafik ile ilgili API fonksiyonları
Kernel32.dll	Windows çekirdek fonksiyonları ilgili API fonksiyonları
Netapi32.dll	32-bit Network ilgili API fonksiyonları
User32.dll	Ekranda görünen nesnelerin fonksiyonları
Winmm.dll	Multimedya ilgili API fonksiyonları

Tablo 2.1 :API dosyaları

Bazı işlemler için kullanılan API örnekleri Tablo 2.2’de görülmektedir.

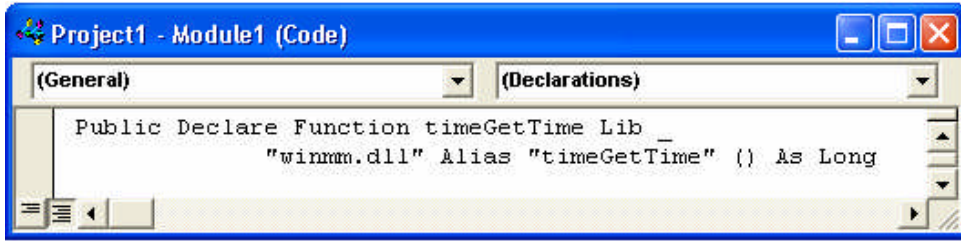
Ses Dosyalarını Çalma	Declare Function <i>sndPlaySound</i> Lib "winmm.dll" Alias "sndPlaySoundA" (ByVal lpszSoundName as string, ByVal uFlags as Long) as Long Result = sndPlaySound (SoundFile, 1)
Zamanlama	Declare Function <i>GetTickCount</i> Lib "kernel32" Alias "GetTickCount" () As Long
Sabitdiskin Seri Numarasını Al	Declare Function <i>GetVolumeInformation</i> 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

Tablo 2.2: API örnekleri

ÖRNEK 2.1: Metin kutusuna 1 saniye aralıkla birden ona kadar rakam yazma.

Bu amaçla timeGetTime isimli bir gecikme API fonksiyonu kullanılacaktır. Bu API Windows açıldığından beri geçen süreyi milisaniye cinsinden tutar.

ADIM 1: Projeye Projects|Add Module seçeneği ile yeni bir modül ekleyelim. Bu modülün içine API Text Viwer ile “timeGetTime” API fonksiyonunu bularak yapıştıralım.



Eğer API fonksiyonunu ayrı modül içinde değil de, formun kod bölümünde tanımlamak istersek Public yerine Private yazmamız gerekir.

```
Private Declare Function timeGetTime Lib "winmm.dll" _  
Alias "timeGetTime" () As Long
```

Burada karşımıza daha önce bahsedilen “Alias” ifadesi çıktı. API doğru olarak tanımlandığı halde VB, ilgili dosyada böyle bir API bulunmadığını söylüyorsa ya da API ile aynı isimde bir VB komutu varsa bu durumda Alias(takma ad) kullanmak gerekir. Genelde API isminin sonuna “A” ya da “W” harfleri getirilerek Alias ismi oluşturulur.

Bizim örneğimizde timeGetTime isimli bir başka VB komutu olmadığından Alias kaldırılarak tanımlama Public Declare Function timeGetTime Lib "winmm.dll" () As Long şeklinde de yapılabilir.

ADIM 2: Form üzerine bir Text kutusu ve bir düğme yerleştirilir. Öncelikle beklemeyi sağlayacak alt yordam formun kod bölümüne yazılır.

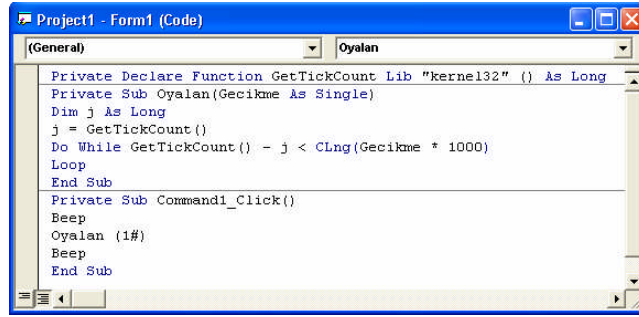
```
Public Sub Sukunet(Gecikme_Miktari As Single)  
Dim Bekleme As Single  
Bekleme = Gecikme_Miktari + timeGetTime()  
Do Until timeGetTime() >= Bekleme  
DoEvents  
Loop  
End Sub
```

Burada Gecikme_Miktari değişkenine gecikme süresi olarak milisaniye cinsinden bir değer yazılacaktır. Kullanımı 1 saniyelik bekleme için “Sukunet 1000” şeklinde olacaktır. DoEvents ise Windows’un bu arada sadece beklemeye takılıp kalmadan kendisinden beklenen başka işleri de yapmasını sağlar.

ADIM 3: Rakam yazdırma.

```
Private Sub Command1_Click()  
Dim sayi As Integer  
For sayi = 1 To 10  
Text1.Text = sayi  
Sukunet 1000  
Next  
Text1.Text = ""  
End Sub
```

Bu programda alt yordama 1000 sayısı gönderilmekte, bu rakam Windows'un açıldığından beri geçen süreye ilave edilmektedir. Zamanın dolmasıyla, yeni süre 1000 milisaniyeden büyük olmakta metin kutusuna diğer sayı yazılmaktadır. Bu API fonksiyonuna eş değer bir diğer API, GetTickCount'tur. Form üzerine bir düğme koyarak aşağıdaki kodlar yazılır. Program, 1 saniyelik gecikmeyle ses çıkaracaktır.

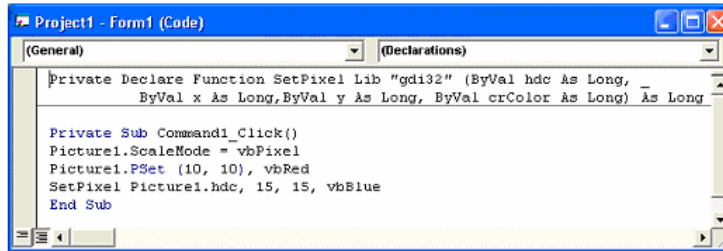


```
Project1 - Form1 (Code)  
(General) Oyalan  
Private Declare Function GetTickCount Lib "kernel32" () As Long  
Private Sub Oyalan(Gecikme As Single)  
Dim j As Long  
j = GetTickCount()  
Do While GetTickCount() - j < CLng(Gecikme * 1000)  
Loop  
End Sub  
Private Sub Command1_Click()  
Beep  
Oyalan (1#)  
Beep  
End Sub
```

2.3. Piksel Çizimi ve Okuması

Resim kutusuna Pset ile bir şeyler çizmeye başladıktan sonra, yine pikselle ilgili API ile devam etmek istenirse öncelikle form üzerine bir düğme ve resim kutusu yerleştirilir.

SetPixel() form yada PictureBox içine belirlenen yere bir piksel koyar. API Viewer'dan yararlanarak SetPixel ile formun başına yapıştırılır.



```
Project1 - Form1 (Code)  
(General) (Declarations)  
Private Declare Function SetPixel Lib "gdi32" (ByVal hdc As Long, _  
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long  
Private Sub Command1_Click()  
Picture1.ScaleMode = vbPixel  
Picture1.PSet (10, 10), vbRed  
SetPixel Picture1.hdc, 15, 15, vbBlue  
End Sub
```

Görüldüğü gibi her iki yöntem de beraber kullandı. SetPixel ile hangi kontrole veri gönderileceği Picture1.hdc parametresi ile bildirildi. Burada hdc, "Handle Device Context"

ifadesinin kısaltılmışıdır. Daha doğrusu Device Context nesnesine bir tutamaktır. Device Context 32 bitlik bir nesnedir ve Windows tabanlı bir uygulama yada bir aygıt sürücüsü ile yazıcı, ekran gibi çıkış birimleri arasında bir bağıdır. Bu nesne grafik çizmek için hafızada oluşturulur ve kendine has Font, Brush, Pen gibi grafiksel nitelikleri vardır.

hDC ise nesnelere tanımlamak için bir isimdir. VB ortamı form, resim kutusu, düğme gibi çok farklı kontrollerden oluşur. VB'de API ile kontrol isimleri üzerinde çalışırken bu kontrollerin isminden ziyade hDC numaraları kullanılır. Bu numaralar Windows tarafından üretilir.

Bir pikselin rengi de GetPixel API'si ile öğrenilir. Tanımı

```
Private Declare Function GetPixel Lib "gdi32" Alias "GetPixel" (ByVal hdc As Long, ByVal x As Long, ByVal y As Long) As Long
```

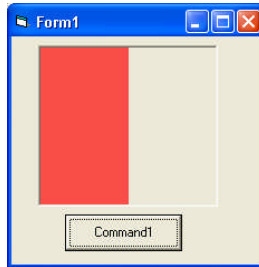
şeklindedir.

Form üzerine bir resim kutusu ve bir düğme yerleştirerek aşağıdaki kodlar yazılır.

```
Private Declare Function GetPixel Lib "gdi32" (ByVal hdc As Long, _  
ByVal x As Long, ByVal y As Long) As Long  
Private Declare Function SetPixel Lib "gdi32" (ByVal hdc As Long, _  
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
```

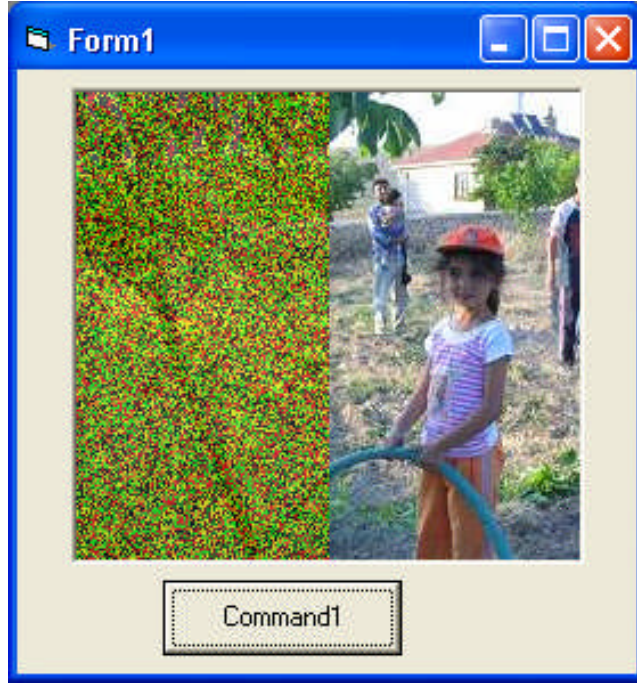
```
Dim Renk, i, j As Integer  
Private Sub Command1_Click()  
Picture1.ScaleMode = VbPixel  
For i = 0 To ((Picture1.ScaleWidth - 1) \ 2)  
For j = 0 To Picture1.ScaleHeight - 1  
Renk = GetPixel(Picture1.hdc, i, j)  
Renk = Abs(Renk) \ 3  
SetPixel Picture1.hdc, i, j, Renk  
DoEvents  
Next j  
Next i  
End Sub
```

Program çalıştırıldığında resim kutusunun yarısına kadar boyandığı görülür.



Şekil 2.7 : Program çıktısı

Resim kutusuna bir resim yerleştirilip program çalıştırılırsa renk cümbüşü meydana gelir.



Şekil 2.8. :Program çıktısı

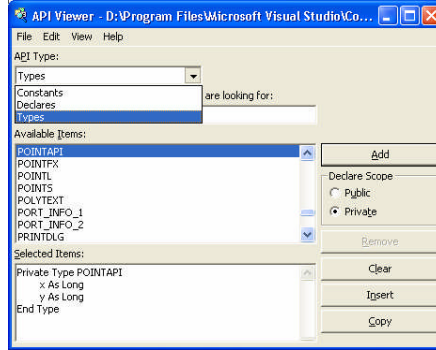
2.4. Çizgi Çizimi

Çizgi için tek bir API yoktur. Bunun yerine iki API fonksiyonundan yararlanarak çizgi çizilir. Birisi bulunan noktayı değiştirmek için diğeri ise bu noktadan diğere noktaya çizgi çizmek için. Bunlar MoveToEx () ve LineTo().

```
Private Declare Function MoveToEx Lib "gdi32" Alias "MoveToEx" _  
    (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, _  
    lpPoint As POINTAPI) As Long
```

```
Private Declare Function LineTo Lib "gdi32" Alias "LineTo" _  
    (ByVal hdc As Long, ByVal x As Long, ByVal y As Long) As Long
```

MoveToEx()'in içinde POINTAPI isminde bir parametre vardır. Bu tekrar API Viewer'dan görülebilir. API Viewer sadece fonksiyonları göstermez aynı zamanda sabitleri(constants) ve tipleri(types) de gösterir.



Şekil 2.9. : API Viewer

POINTAPI parametresi, şekilde görüldüğü gibi bulunarak kopyalanır.

```
Private Type POINTAPI
    x As Long
    y As Long
End Type
```

Bu noktayı program içerisinde temsil edecek bir değişken tanımlanır.

```
Dim Cerbe As POINTAPI
```

Mevcut nokta, çizginin başlayacağı ilk nokta olan 10,15 noktasına taşınır.

```
MoveToEx Picture1.hdc, 10, 15, Cerbe
```

Buradan son noktaya bir çizgi çizilir.

```
LineTo Picture1.hdc, 237, 156
```

Çizginin rengi siyah. API tanımlaması içinde, renkle alakalı bir parametre yok. Renk ilave edilmek istenirse kontrollerin özelliklerinden yararlanılır. LineTo()'dan önce

```
Picture1.ForeColor=vbYellow
```

yazılabilir.

2.5. Dikdörtgen Çizimi

Dikdörtgen için Rectangle() API'si kullanılır. Tanımlaması:

```
Private Declare Function Rectangle Lib "gdi32" Alias "Rectangle" (ByVal hdc As Long, _
    ByVal X1 As Long, ByVal Y1 As Long, ByVal X2 As Long, ByVal Y2 As Long) As Long
```

Örneğin form üzerine mavi zemin üzerine kırmızı bir dikdörtgen çizilmek istenirse;

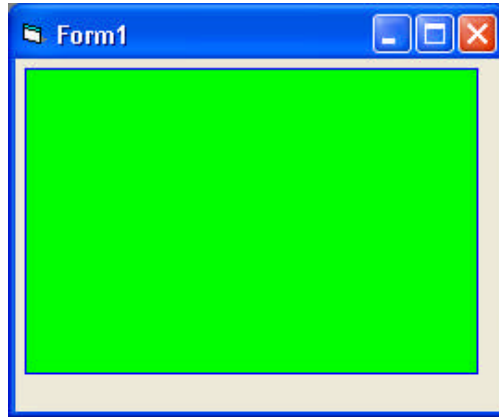
```
Private Sub Form_Click()  
Form1.ScaleMode = vbPixel  
Form1.ForeColor = vbRed  
Form1.BackColor = vbBlue  
Rectangle Form1.hdc, 25, 50, 250, 250  
End Sub
```

kodları kullanılabilir.

Eğer dikdörtgen doldurulmak istenirse;

```
Form1.ForeColor = vbBlue  
Form1.FillColor = vbGreen  
Form1.FillStyle = vbFSSolid
```

satırları Rectangle'dan önce yazılır.



Şekil 2.10. : API ile dikdörtgen çizimi çıktısı

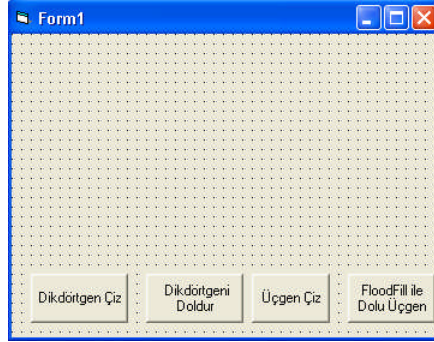
Aslında şekillerin içini doldurmak için FloodFill isiminde bir API daha vardır.

```
Private Declare Function FloodFill Lib "gdi32" (ByVal hdc As Long, _  
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
```

x ve y, doldurulacak alanın içinde kalan bir koordinattır. crColor ile dolum rengi.

ÖRNEK 2.2: FloodFill ile üçgen doldurma.

ADIM 1: Form tasarımı şekilde görüldüğü gibi yapılır.



Şekil 1.11: Form tasarımı

ADIM 2: API bildirimleri

```
Private Declare Function Rectangle Lib "gdi32" (ByVal hdc As Long, _  
ByVal X1 As Long, ByVal Y1 As Long, ByVal X2 As Long, _  
ByVal Y2 As Long) As Long
```

```
Private Declare Function FloodFill Lib "gdi32" (ByVal hdc As Long, _  
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
```

ADIM 3: Formun Load olayı

```
Private Sub Form_Load()  
Form1.ScaleMode = vbPixels  
End Sub
```

ADIM 4: Dikdörtgen çizme

```
Private Sub Command1_Click()  
Form1.ForeColor = vbBlue  
Form1.FillColor = vbGreen  
Rectangle Form1.hdc, 5, 5, 150, 70  
End Sub
```

ADIM 5: Dikdörtgeni doldurma

```
Private Sub Command2_Click()  
Form1.FillStyle = vbFSSolid  
Rectangle Form1.hdc, 5, 5, 150, 70  
End Sub
```

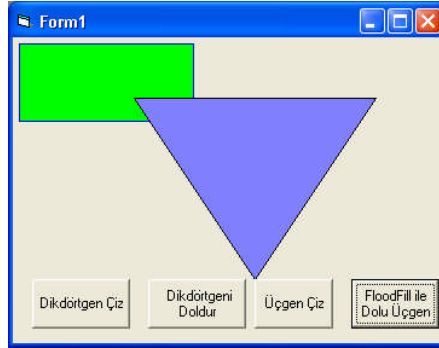
ADIM 6: Üçgen çizme

```
Private Sub Command3_Click()  
ForeColor = vbBlack  
Line (100, 50)-(300, 50)  
Line -(200, 200)  
Line -(100, 50)  
End Sub
```

ADIM 7: FloodFill ile üçgeni doldurma

```
Private Sub Command4_Click()  
FillStyle = vbFSSolid  
FillColor = RGB(128, 128, 255)  
FloodFill hdc, 200, 100, ForeColor  
End Sub
```

ADIM 8: Programı çalıştırma



Şekil 2.12. : Program çıktısı

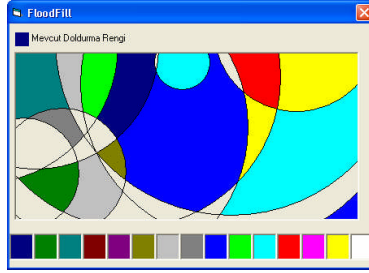
Görüldüğü gibi dikdörtgeni doldurmak için FillStyle özelliğine vbFSSolid atayıp dikdörtgeni tekrar çizmek gerekiyor. Floodfill’de ise FillStyle özelliği belirlendikten sonra sadece doldurulacak alandan bir noktanın koordinatını vermek gerekiyor.

FloodFill hdc, 200, 100, ForeColor
satırı aşağıdaki satırla değiştirilir. Dikdörtgeni çizdikten sonra “FloodFill ile Dolu Üçgen” başlıklı düğmeye tıklanır.
FloodFill hdc, 20, 20, ForeColor

ÖRNEK 2.3: Fare ile seçilen yerleri doldurma.

Fare ile kapalı alanlardan işaretlenen yerler doldurulacaktır. Bunun için rastgele çemberler çizilerek renk paletinden alınan seçilen renklere göre içi doldurulacaktır.

Programın son çalışma şekli şekilde görülmektedir.



Şekil 2.13 : Program çalışması

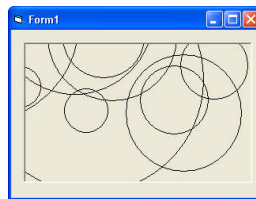
Bu örneği adım adım gerçekleştirmek için;

ADIM 1: Form üzerine bir PictureBox yerleştirilir. Ve aşağıdaki kod eklenir.

```
Private Sub Form_Load()  
Dim i As Integer  
Randomize  
Picture1.AutoRedraw = True  
For i = 1 To 10  
RasgeleDaireCiz  
Next i  
With Picture1  
FillStyle = vbFSSolid  
ScaleMode = vbPixels  
End With  
End Sub
```

```
Private Sub RasgeleDaireCiz()  
Dim Cerbe As Long  
Dim x As Long, y As Long  
Cerbe = Rnd * Picture1.ScaleHeight + 200  
x = Rnd * Picture1.ScaleWidth - Cerbe / 2  
y = Rnd * Picture1.ScaleHeight - Cerbe / 2  
Picture1.Circle (x, y), Cerbe  
End Sub
```

Resim kutusunun ScaleWidth ve ScaleHeight özelliklerine göre Rnd fonksiyonu ile rastgele üretilen sayılar çizilerek, 10 tane çemberin merkezini belirlemektedir. Her program çalıştırıldığında rastgele üretilen sayının değişmesini Randomize ifadesi sağlamaktadır.



Şekil 2.14 : İlk çıktı

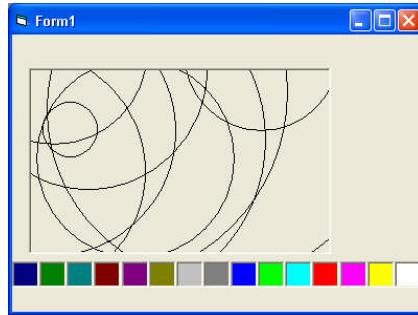
ADIM 2: Resim kutusunun altına bir tane etiket(Label) yerleştirilir. Name özelliğine "lblRenk" ve Index özelliğine 0 atanır. Bu, renklerin seçilmesine yarayacak kutulara temel oluşturacaktır.

Declarations bölümüne iki değişken tanımlanır.
Private Renk As Long
Private indeks As Integer

Formun Load olayına italik yazı ile yazılanlar eklenir.

```
Private Sub Form_Load()  
Dim i As Integer  
Randomize  
Picture1.AutoRedraw = True  
For i = 1 To 10  
RasgeleDaireCiz  
Next i  
With Picture1  
FillStyle = vbFSSolid  
ScaleMode = vbPixels  
End With  
lblRenk(0).Caption = ""  
lblRenk(0).BorderStyle = 1  
lblRenk(0).Left = 0  
lblRenk(0).Width = (Form1.ScaleWidth) / 15  
lblRenk(0).BackColor = QBColor(1)  
For i = 1 To 14  
Load lblRenk(i)  
lblRenk(i).Visible = True  
lblRenk(i).Left = lblRenk(i - 1).Left + lblRenk(i - 1).Width  
lblRenk(i).BackColor = QBColor(i + 1)  
Next  
End Sub
```

Program çalıştırıldığında aşağıdaki ekran görüntüsü elde edilir.



Şekil 2.15 : Program ikinci aşama

ADIM 3: Seçilen rengi gösterecek bir etiket resim kutusu üzerine ilave edilir. İsmi "lblSimdikiRenk" olarak belirlenir ve BackColor özelliği, mavi renk(&H00FF0000&) seçilir.

ADIM 4: Renk kutularına tıkladığında seçilen rengi alacak yordam

```
Private Sub lblRenk_Click(Index As Integer)
Renk = lblRenk(Index).BackColor
lblSimdikiRenk.BackColor = Renk
End Sub
```

ADIM 5: FloodFill fonksiyonu formun başına yazılır.

```
Private Declare Function FloodFill Lib "gdi32" (ByVal hdc As Long, _
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
```

ADIM 5: Fare ile tıkladığında çemberlerin içini doldurmak için

```
Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, _
x As Single, y As Single)
```

```
Dim DonusDegeri As Long
```

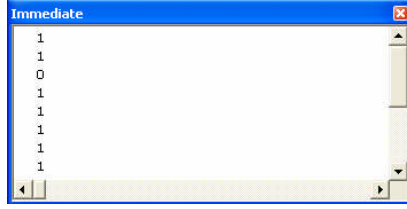
```
'Foodfill işleminin rengini belirler
Picture1.FillColor = Renk
```

```
'Çizgiler siyah çizildiğinden,FloodFill'e sınır çizgisi siyah geçirilir.
DonusDegeri = FloodFill(Picture1.hdc, x, y, vbBlack)
```

```
'Tazeleme yapılmazsa değişiklik gözlenemez
Picture1.Refresh
Debug.Print DonusDegeri
```

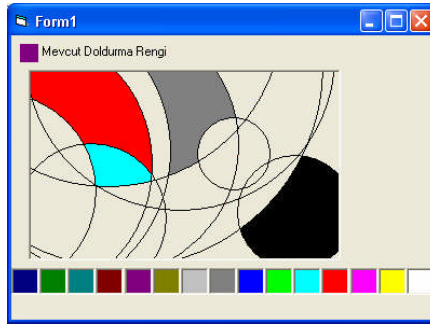
```
'Renk indeksini bir arttır
indeks = indeks + 1
If indeks > lblRenk.UBound Then indeks = 0
Renk = lblRenk(indeks).BackColor
lblSimdikiRenk.BackColor = Renk
End Sub
```

FloodFill bir fonksiyon olduğu için geriye bir değer döndürür. İşlem başarılı olduğunda 1, başarısız olduğunda 0 döndürür. Biz bunu Debug.Print DonusDegeri komutu ile Immediate penceresinde gösterdik.



Şekil 2.16 : Immediate penceresi

Program çalıştırılır.



Şekil 2.17 : Program üçüncü aşama

ADIM 6: Programın düzenlenmiş hali aşağıda görülmektedir.

```
Option Explicit
Private Declare Function FloodFill Lib "gdi32" (ByVal hdc As Long, _
ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
Private Renk As Long
Private indeks As Integer
Private Sub Form_Load()
Dim i As Integer
Randomize
Picture1.AutoRedraw = True
For i = 1 To 10
RasgeleDaireCiz
Next i
With Picture1
FillStyle = vbFSSolid
ScaleMode = vbPixels
End With

lblRenk(0).Caption = ""
lblRenk(0).BorderStyle = 1
lblRenk(0).Left = 0
lblRenk(0).Width = (Form1.ScaleWidth) / 15
lblRenk(0).BackColor = QBColor(1)
```

```

For i = 1 To 14
Load lblRenk(i)
lblRenk(i).Visible = True
lblRenk(i).Left = lblRenk(i - 1).Left + lblRenk(i - 1).Width
lblRenk(i).BackColor = QBColor(i + 1)
Next

indeks = 0
Renk = lblRenk(indeks).BackColor
lblSimdikiRenk.BackColor = Renk
End Sub

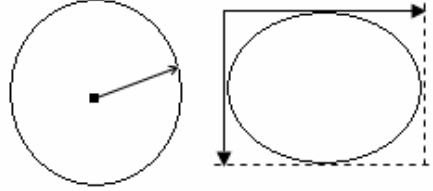
Private Sub RasgeleDaireCiz()
Dim Cerbe As Long
Dim x As Long, y As Long
Cerbe = Rnd * Picture1.ScaleHeight + 200
x = Rnd * Picture1.ScaleWidth - Cerbe / 2
y = Rnd * Picture1.ScaleHeight - Cerbe / 2
Picture1.Circle (x, y), Cerbe
End Sub
Private Sub lblRenk_Click(Index As Integer)
Renk = lblRenk(Index).BackColor
lblSimdikiRenk.BackColor = Renk
End Sub

Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, x As Single, y
As Single)
Dim DonusDegeri As Long
Foodfill işleminin rengini belirler
Picture1.FillColor = Renk
Çizgiler siyah çizildiğinden,FloodFill'e sınır çizgisi siyah geçirilir.
DonusDegeri = FloodFill(Picture1.hdc, x, y, vbBlack)
Tazeleme yapılmazsa değişiklik gözlenemez
Picture1.Refresh
Debug.Print DonusDegeri
Renk indeksini bir arttır
indeks = indeks + 1
If indeks > lblRenk.UBound Then indeks = 0
Renk = lblRenk(indeks).BackColor
lblSimdikiRenk.BackColor = Renk
End Sub

```

2.6. Daire ve Elips Çizimi

VB ile API fonksiyonlarının daire çizimine yaklaşımı farklıdır. VB'de çemberin merkezi ve yarıçapı parametre olarak verilir. API ise köşeden köşeye olan koordinatı kullanır.



Şekil 2.18. : VB ile API'deki çember çizim kuralı farkı

Bu metod hem elipsi hem de daireyi çizme imkanı verir. Elipse API'si:

```
Private Declare Function Ellipse Lib "gdi32" Alias "Ellipse" _  
(ByVal hdc As Long, ByVal X1 As Long, ByVal Y1 As Long, _  
ByVal X2 As Long, ByVal Y2 As Long) As Long
```

Çemberin merkezini ve yarıçapını dış köşe koordinatına çevirirsek;

```
X1=Merkez.X-Yarıçap  
Y1=Merkez.Y-Yarıçap  
X2=Merkez.X+Yarıçap  
Y2=Merkez.Y+Yarıçap
```

Ellipse Picture1.hdc, 0, 5, 120, 135 komutu bir daire çizerken

Ellipse Picture1.hdc, 90, 15, 270, 150 komutu bir elips çizmektedir.

2.7. Polyline ile Çoklu Çizgi

Ardışık olarak birden fazla çizgi çizmek için PolyLine API fonksiyonu kullanılır. Bu fonksiyon Line metoduna benzemekle beraber CurrentX ve CurrentY ile tutulan son nokta, bu DLL fonksiyonunda tutulmaz. Bildirimi:

```
Private Declare Function Polyline Lib "gdi32" Alias "Polyline" _  
(ByVal hdc As Long, lpPoint As POINTAPI, ByVal nCount As Long) As Long  
Burada lpPoint, çizgilerin koordinatlarını tutan bir dizideki ilk noktadır ve POINTAPI  
olarak tanımlanan kullanıcı tanımlı bir değişken tipidir.  
Private Type POINTAPI  
x As Long  
y As Long  
End Type
```

Bu yapıyla ilişkili olan değişken, X ve Y olmak üzere iki koordinat değerine sahip olacaktır.

```
Dim Kemal_Reis(10) As POINTAPI
```

ÖRNEK 2.4: Fare ile rastgele çizimler.

ADIM 1: Form üzerine bir PictureBox ve bir düğme eklenir. ScaleMode özelliğini vbPixels değerine ayarlanır.

ADIM 2: General Declarations bölümüne yazılacak tanımlamalar.

```
Private Declare Function Polyline Lib "gdi32" (ByVal hdc As Long, _  
lpPoint As POINTAPI, ByVal nCount As Long) As Long  
Private Type POINTAPI  
X As Long  
Y As Long  
End Type
```

```
Dim indeks As Integer  
Dim Kemal_Reis(1000) As POINTAPI
```

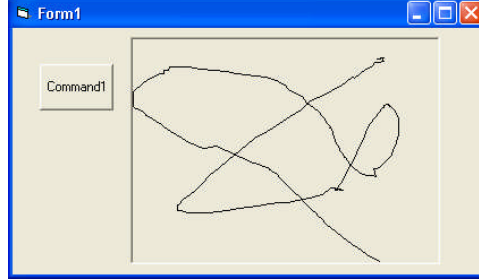
ADIM 3: Fare gezdirildiğinde çizgiler çizilsin.

```
Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
If Index = 0 Then Picture1.Cls  
indeks = indeks + 1  
Kemal_Reis(indeks).X = X  
Kemal_Reis(indeks).Y = Y  
n = Polyline(Picture1.hdc, Kemal_Reis(1), indeks)  
End Sub
```

ADIM 4: Düğme tıklandığında çizgiler silinsin.

```
Private Sub Command1_Click()  
Dim n As Integer  
Picture1.Cls  
indeks = 0  
End Sub
```

ADIM 5: Program çalıştırılır.



Şekil 2.19 : Örnek program çıktısı

Fareyi gezdirdikçe koordinat Kemal_Reis dizisine kopyalanmakta, daha sonra PolyLine, diziden ilk noktaları okuyarak çizgileri çizmektedir.

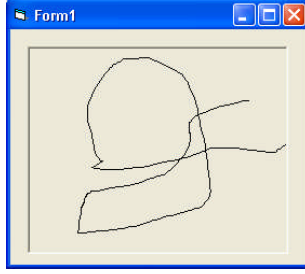
Aynı işi Line komutu ile yapmak istenirse aşağıdaki kodların yazılması gerekir.

```
Private Type POINTAPI
X As Single
Y As Single
End Type
Dim indeks As Integer
Dim Kemal_Reis(1000) As POINTAPI

Sub ÇizgileriCiz()
Picture1.Cls
Dim n As Integer
Picture1.PSet (Kemal_Reis(1).X, Kemal_Reis(1).Y)
For n = 2 To indeks
Picture1.Line -(Kemal_Reis(n).X, Kemal_Reis(n).Y)
Next
End Sub

Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, _
X As Single, Y As Single)
If indeks = 0 Then Picture1.PSet (X, Y) Else Picture1.Line -(X, Y)
indeks = indeks + 1
Kemal_Reis(indeks).X = X
Kemal_Reis(indeks).Y = Y
ÇizgileriCiz
End Sub
```

Program çalıştırılır.



Şekil 2.20 : Program çıktısı

Aynı iş yapıldı. Ya çizilenlerin bütün olarak değil de tek tek silinmesi istenseydi! Aşağıdaki yordam ilave edilir.

```
Private Sub Picture1_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = 46 And indeks > 0 Then '46 Delete tuşu
indeks = indeks - 1
ÇizgileriCiz
End If
End Sub
```

Artık Del tuşuna bastıkça sondan başa doğru çizgilerin silindiği görülebilir.

Kullanıcı tanımlı POINTAPI değişkenine, değerleri doğrudan verilebilir. Aşağıdaki kod resim kutusunda bir üçgen çizmektedir.

```
Private Type POINTAPI
x As Single
y As Single
End Type

Dim Piyale_Reis(3) As POINTAPI
Private Sub Command1_Click()
Picture1.Scale (-120, 230)-(120, -10)
Piyale_Reis(1).x = 0: Piyale_Reis(1).y = Sqr(3) * 100
Piyale_Reis(2).x = -100: Piyale_Reis(2).y = 0
Piyale_Reis(3).x = 100: Piyale_Reis(3).y = 0
UcgenCiz 5, Piyale_Reis(1), Piyale_Reis(2), Piyale_Reis(3)
End Sub

Private Sub UcgenCiz(n As Integer, p1 As POINTAPI, _
p2 As POINTAPI, p3 As POINTAPI)
Dim pa As POINTAPI, pb As POINTAPI, pc As POINTAPI
Picture1.Line (p1.x, p1.y)-(p2.x, p2.y)
Picture1.Line (p2.x, p2.y)-(p3.x, p3.y)
Picture1.Line (p3.x, p3.y)-(p1.x, p1.y)
End Sub
```

2.7. Metin Yazmak

Metin yazmak için TextOut API'si bir cümleyi resim kutusuna yazmak için kullanılan Print metodunun eş değeridir.

```
Private Declare Function TextOut Lib "gdi32" Alias "TextOutA" _  
(ByVal hdc As Long, ByVal x As Long, ByVal y As Long, _  
ByVal lpString As String, ByVal nCount As Long) As Long
```

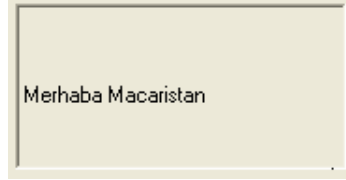
Burada x ve y yazılması istenen koordinatı, lpString metnin kendisini, nCount ise metnin uzunluğunu ifade etmektedir.

Öncelikle cümle bir sabite atanır.

```
Const metin As String = "Merhaba Macaristan"
```

Daha sonra resim kutusuna gönderilir.

```
TextOut Picture1.hdc, 2, 40, metin, Len(metin)
```



Şekil 2.21 : Program çıktısı

Metin resim kutusunun varsaydığı font ile yazılır. Rengi ise ForeColor ile belirtilen renktir. Font rengini belirleyen SetTextColor isminde bir API vardır.

```
Private Declare Function SetTextColor Lib "gdi32" _  
(ByVal hdc As Long, ByVal crColor As Long) As Long
```

Yukarıdaki satıra renk eklenir.

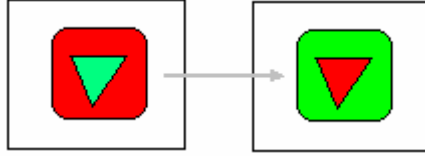
```
SetTextColor Picture1.hdc, vbRed  
TextOut Picture1.hdc, 2, 40, metin, Len(metin)
```

Kırmızı renkte "Merhaba Macaristan" yazıldı.

Grafik, kullanıcıya görsel bir ortam sunar. API fonksiyonlarının en sık kullanım alanlarından başında grafik gelmektedir. Arka planda resimlerin kayması, sprite adı verilen küçük resimlerin hareket etmesi, görsel efektler gibi pek çok grafiksel olaylar API fonksiyonlarıyla çok daha kolay yapılmaktadır. Şimdi bu API fonksiyonları ile görelim.

2.8. BitBlt API Fonksiyonu

BitBlt (Bit Block Transfer), resim alanının bir bölümünü bir yerden(kaynak) başka bir yere (hedef) kopyalamak için kullanılır.



Şekil 2.22 : Resim alanının bir bölümünü başka yere kopyalama

Bildirimi aşağıda görüldüğü gibidir.

```
Public Declare Function BitBlt Lib "gdi32" Alias "BitBlt" _ (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
```

Burada;

hDestDC: Hedef resmin hDC numarası

X,Y: Kopyalama yapılacak hedef resmin sol üst koordinatı

nWidth: Kopyalanan resmin(kaynak) genişliği

nHeight: Kopyalanan resmin yüksekliği

hSrcDC: Kaynak resmin hDC numarası

xSrc, ySrc: Kaynak resmin sol üst köşe koordinatı

dwRop: Resmin nasıl kopyalanacağı belirten bir sabit. Doğrudan bir kopya yapılacaksa dwRop, SRCCOPY sabitine eşitlenir.

BitBlt, tüm geometrik birimlerin piksel olmasını ister. Fonksiyonun işleyişi sırasında geriye Long türünde bir değer döner. Programlarda genelde kullanılmamasına rağmen bu değere bakarak fonksiyonun görevini başarıyla yapıp yapmadığına bakılır.

Form üzerine bir düğme ve bir picturebox yerleştirdikten sonra, Picture özelliğinden yararlanarak bir resim yüklenir ve aşağıdaki kod yazılır.

```
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 Command1_Click()
```

```
Me.Cls
```

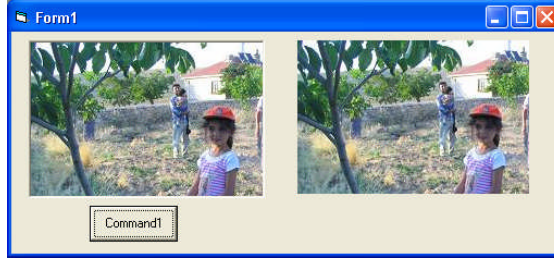
```
BitBlt Me.hDC, Picture1.ScaleWidth + 50, Picture1.Top, Picture1.ScaleWidth, _
```

```
Picture1.ScaleHeight, Picture1.hDC, 0, 0, vbSrcCopy
```

```
End Sub
```

```
Private Sub Form_Load()  
Form1.ScaleMode = vbPixels  
Picture1.ScaleMode = vbPixels  
End Sub
```

Programın çalışma görüntüsü:



Şekil 2.23 : Program çıktısı

Düğme tıklandığında BitBlt fonksiyonu Picture1.hdc'nin işaret ettiği yerden resmi alarak form üzerinde koordinatları gösterilen yere kopyalamaktadır. BitBlt'nin hDestDC parametresi olarak formun (Burada Me) hDCsi belirtilmiştir.

Resmin kopyalanacağı koordinat olarak resim kutusu genişliğinin 50 piksel sağ (X) ve yine resim kutusunun üst koordinatıdır (Y). Kaynak hDCsi olarak da resim kutusunun hDC numarası ayarlanmıştır.

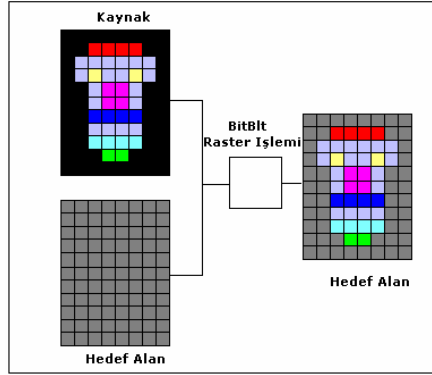
Bu yöntem daha önce anlatılan PaintPicture yöntemi ile aynı işi yapmasına rağmen daha hızlıdır. Aslında PaintPicture, dahili kodunda bu fonksiyonu kullanmaktadır.

Bu örnekte dwRop parametresini vbSrcCopy yaparak resim kutusunda olan her şey olduğu gibi hedef alana kopyalandı. Yani hedef piksellerinin hepsi kaynak pikselleriyle aynı oldu. Eğer Bitblt'nin düşündüğünü duyabilseydik herhalde kendi kendine şöyle dediğini duyardık:

“Şimdi kaynaktan şu siyah pikseli kopyaladım ve zemini kırmızı olan bir yere kopyalayacağım. İki rengi karıştırsam mı yoksa diğerinin yerine mi koysam?” BitBlt'ye burada nasıl hareket etmesini söylediğimiz yer, işte dwRop parametresidir. Bu işlemlere Raster işlemleri denilmektedir.













Tablo 2.3'te örnek resim eşliğinde bu işlemlerin açıklamaları görülmektedir.

Raster işlemleri için hem hedef yerin mevcut piksel değerleri hem de kopyalanan yerin piksel bilgisine ihtiyaç vardır. Raster işlemleri NOT, AND, OR ve XOR gibi mantıksal işlemler yaparak pikselin bu renk bilgisini hedef alanda yeni bir piksel oluşturmak için kullanılmaktadır.



Şekil 2.24 : Kopyalama işlemi esnasında olanlar

Sabit ve Değerleri	Açıklama Başlangıç Resmi →		
vbDstInvert &H00550009	Hedef alanı tersine çevirir.		
vbMergeCopy &H00C000CA	Hedef ile kaynak desenini birleştirir.		
vbMergePaint &H00BB0226	Hedef ile terslenmiş kaynak alanı OR işlemine tabii tutar.		

<p>vbNotSrcCopy &H00330008</p>	<p>Kaynağı tersler ve hedefe kopyalar.</p>		
<p>vbNotSrcErase &H001100A6</p>	<p>Kaynak ile hedefi OR işlemine tabii tuttuktan sonra tersler.</p>		
<p>vbSrcAnd &H008800C6</p>	<p>Hedef ile kaynak alanı AND operatörü ile işleme tabii tutar.</p>		
<p>vbSrcCopy &H00CC0020</p>	<p>Kaynağı hedefe doğrudan kopyalar.</p>		
<p>vbSrcErase &H00440328</p>	<p>Terslenmiş hedefi kaynakla AND işlemine tabii tutar.</p>		
<p>vbSrcInvert &H00660046</p>	<p>Terslenmiş hedefi kaynakla XOR işlemine tabii tutar.</p>		

vbSrcPaint &H00EE0086	Hedef ile kaynak alanı OR işlemine tabii tutar.	
--------------------------------------	--	--

Tablo 2.3 : Raster işlemleri

2.9. Sprite ve Maskeleme

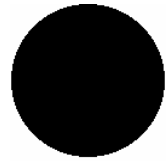
Sprite(peri), grafiksel alanda hareket eden küçük resim parçalarına denir. Genellikle oyun ve demo programlarında kullanılır. Ekran üzerinde yeniden konumlandırabilir ve boyutlandırılabilir.

Bir resmin ya da nesnenin arka planla bütünleşmesi istenen yerlerde bazı işlemlerin uygulanması gerekir ki bu bölümde bunun üzerinde durulacaktır.



Şekil 2.25 : Hareket ettirilecek resim alanı

Sadece ortadaki mavi dairenin belli alanda hareket etmesi istenirse maskelenmesi gerekir. Maske sadece iki renkten (siyah ve beyaz) oluşan özel bir bitmap resimdir.



Şekil 2.26 : Maske

Maskedeki beyaz alanlar şeffaf alanlar, siyah alanlar da görmek istediğimiz alanlar olacaktır. BitBlt fonksiyonu ile önce maske, arkasından sprite bir yüzeye kopyalanır. Sonuç mavi bir daire olacaktır.

Maske yüzeye yapıştırılırken BitBlt fonksiyonunun dwRop parametresi, vbSrcAnd'dir. Bu işlem hedef alanın rengini kaynak alanın rengi ile AND'ler.

Hedef alanda arka alanının renginin 11001001(201) olduğu kabul edilir. Kaynak alanın beyaz kısımları 11111111(255) ile AND'lenirse sonuç rengi:

Kaynak:1111 1111
Hedef:1100 1001
AND _____
Sonuç:11001001

Hedef alanın rengi ile aynı oldu.

Arka alanın siyah kısımları için yapılırsa;

Kaynak:0000 0000
Hedef:1100 1001
AND _____
Sonuç:0000 0000

Bu bize maskenin tüm siyah alanının siyah, beyaz alanının ise hedef alandaki renk olacağını göstermektedir. Buna göre hedef alanda siyah bir daire görünecekti. Sanki maske resminin istenmeyen kısımları tırpanlanmış gibi.Bu işlemi görsel olarak yapmak için PaintBrush'ta içi siyah bir daire çizilir. Bunu kaydederken mümkün olduğu kadar daireyi kapsayacak şekilde beyaz bir alanda kaydedilir.

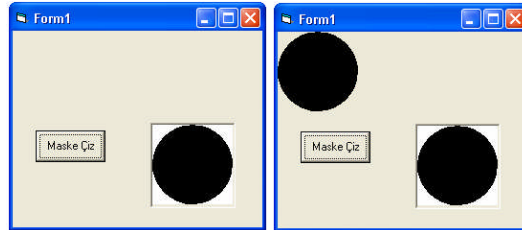


Şekil 2.27 : Maske hazırlığı

Bu resim kaydedildikten sonra bir form açılır, üzerine bir düğme ve bir resim kutusu yerleştirilir. ScaleMode özelliği vbPixels olarak ayarlanır. Formun başında BitBlt API fonksiyonunun bildirimi yapılır. Ve aşağıdaki kod yazılır.

```
Private Sub cmdMaske_Click()  
vbSrcAnd  
BitBlt Me.hDC, 0, 0, Picture1.ScaleWidth, Picture1.ScaleHeight, Picture1.hDC, 0, 0,  
vbSrcAnd  
End Sub
```

Program çalıştırılır.



Şekil 2.28 : Program çıktısı

Bir sonraki iş renkli sprite resmini, hedef alanda aynı koordinata kopyalamaktır. Burada vbSrcPaint raster işlemi dwRod olarak kullanılacaktır. Bu hedef alanı, kaynak alanı ile OR işlemine sokar. Sprite resminin siyah alanı için

Kaynak:0000 0000
Hedef:1100 1001
OR—————
Sonuç:1100 1001

Asıl hedef alanı ile aynı renkte. Sprite'ın siyah alanı işleme tabii tutulduğu renkle aynı olur.

Mavi(0011 1111) alan için;
Kaynak:0011 1111
Hedef:0000 0000 Maskenin siyah kısmı
OR
Sonuç:0011 1111

Bu da mavi renk.

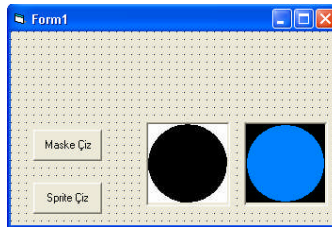


Şekil 2.29 : Mavi maske

Özetlenecek olursa;

- Sprite resminin şeffaf yani görünmez olması istenen tüm alanlar siyah yapılır.
- Nihai sprite resminde şeffaf olması gereken yerler maskede beyaz yapılır. Hedef alanda değişmesi istenilmeyen yerler siyah yapılır.
- Maske, hedef alana vbSrcAnd raster parametresini kullanarak BitBlt fonksiyonu ile kopyalanır.
- Sprite resmi, hedef alana vbSrcPaint raster parametresini kullanarak BitBlt fonksiyonu ile kopyalanır.

Yukarıdaki örnek form üzerine cmdSprite isimli, Sptite Çiz başlıklı bir düğme ve picSprite isiminde bir PictureBox yerleştirerek aşağıdaki kod ilave edilir.



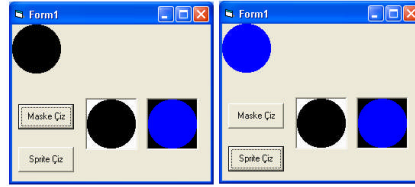
Şekil 2.30: Form tasarımı

```

Private Sub cmdSprite_Click()
'vbSrcPaint
BitBlt Me.hDC, 0, 0, picSprite.ScaleWidth, picSprite.ScaleHeight, _
picSprite.hDC, 0, 0, vbSrcPaint
End Sub

```

Program tekrar çalıştırılır.



Şekil 2.31 : Program çıktısı

Görüldüğü gibi sadece mavi bir daire elde edildi. İlerde animasyon kısmında bu daire form üzerinde hareket ettirilecektir.

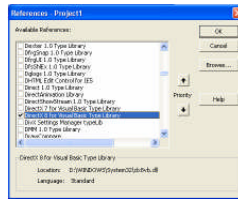
2.10. DirectX

Bu kitabın ana konusu DirectX olmadığı için kısaca hatırlatmakla yetinilecektir. Grafik ortamda bazı işler, API fonksiyonları aracılığıyla gerçekleştirilir. Fakat API'ler genel amaçlar için tasarlanmıştır. Microsoft bu yüzden uzmanlar için DirectX adı verilen grafik DLL yordamları hazırlamıştır. DirectX, çoğunlukla oyun programcılığı için faydalı olan fonksiyonları barındıran DLL kütüphanesidir. DirectX, donanım farklılığından dolayı ortaya çıkan bazı sıkıntıları da önlemektedir. Bir başka grafik kütüphanesi olan OpenGL'in eşdeğeridir.

DirectX aşağıdaki alt bileşenlerden oluşur:

- DirectDraw: 2D grafikler için.
- DirectInput: Klavye, fare, joystick gibi donanımsal girişleri denetler.
- DirectPlay: Ağ haberleşmesini kontrol eder.
- DirectSound: Ses işlerinden sorumludur.
- Direct3D: 3D grafik çizimlerinde kullanılır.

DirectX kütüphanesine Project/References menüsünden ulaşılır.

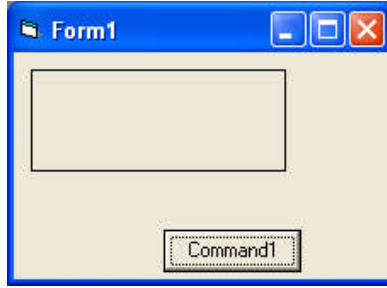


Şekil 2.32: DirectX kütüphanesine erişim

UYGULAMA FAALİYETİ

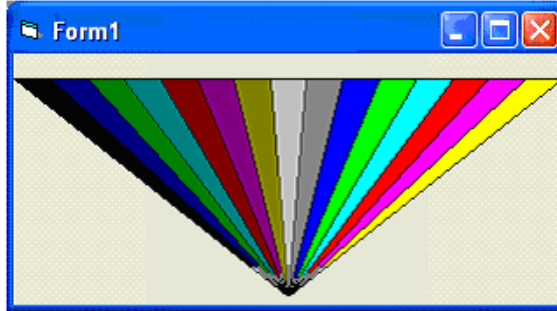
Aşağıdaki sorulara ilişkin uygulama faaliyetini yapınız.

Ekran görüntüsünde olduğu gibi Rectangle API'sini kullanarak formun (10,160) (10,70) noktaları arasına bir dikdörtgen çizersiniz.



Şekil 2.33: Form tasarımı

- Ekran görüntüsünde görünen şekli çizerek FloodFill fonksiyonu ile içini boyayın.



Şekil 2.34: Form tasarımı

İşlem Basamakları	Öneriler
<ul style="list-style-type: none">➤ Şekilleri form üzerine yerleştiriniz.➤ Hangi olayları kullanacağınıza karar veriniz.➤ Döngü tipine iyi karar veriniz.➤ Yazdığınız programı çalıştırınız.➤ Programda hata var ise bunları gideriniz.	<ul style="list-style-type: none">➤ Nesne özelliklerine uygun değerler atayınız.➤ Hangi değişken tipini kullanacağınıza dikkat ediniz.➤ Değişken artım oranlarını iyi ayarlayınız.➤ Program satırlarının düzenli olmasına özen gösteriniz. Programı çalıştırmadan önce muhakkak kaydediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları cevaplayarak bu faaliyette kazandığınız bilgileri ölçünüz.

OBJEKTİF TEST (ÖLÇME SORULARI)

1. Aşağıdakilerden hangisi grafik fonksiyonlarının bulunduğu kütüphanedir?

- A) User32.dll
- B) Kernel32.dll
- C) Gdi32.dll
- D) Winmm.dll

2. Aşağıdakilerden hangisi form ya da resim kutusu içinde belirlenen yere piksel koyan API'dir?

- A) SetPixel
- B) GetPixel
- C) Pset
- D) Point

3. Kapalı alanların içini doldurmak için kullanılan API, aşağıdakilerden hangisidir?

- A) TextOut
- B) GetBitmapBits
- C) SetPixel
- D) FloodFill

4. Bir resim kutusunun içeriğini tek seferde hafızaya kopyalayan API aşağıdakilerden hangisidir?

- A) CopyMemory
- B) GetBitmapBits
- C) VarPtr
- D) GetPixel

5. Aşağıdakilerden hangisi CommonDialog bileşenin metodlarından değildir?

- A) ShowOpen
- B) ShowSave
- C) ShowColor
- D) ShowMode

6.Aşağıdakilerden hangisi 255 Xor 65 işleminin sonucudur?

- A) 190
- B) 68
- C) 185
- D) 168

7.RGB renk değerinden yeşil rengi elde etmek istediğimizde hangi kodla rengi diğerlerinden ayırırız?

- A) Green= (Renk \ (2^8)) And &H00FF
- B) Green= (Renk \ (2^8)) And &HFF
- C) Green= (Renk \ (2^8)) And &HFF00
- D) Green= (Renk \ (2^8)) And &HFFFF

8.Device Context'in değişken tipi aşağıdakilerden hangisidir?

- A) Byte
- B) Double
- C) Single
- D) Long

9.Pset ve Line metodlarına renk ataması hangi fonksiyonla yapılmaz ?

- A) RBG
- B) Renk kodlarını doğrudan yazarak
- C) QBColor
- D) GetSysColor API fonksiyonu ile

10.Aşağıdakilerden hangisi resim alanının bir bölümünü bir yerden(kaynak) başka bir yere (hedef) kopyalamak için kullanılır?

- A) BitBlt
- B) StretchBlt
- C) CopyMemory
- D) VarPtr

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyete geri dönerek tekrar inceleyiniz.

ÖĞRENME FAALİYETİ-3

AMAÇ

Görsel programlamada animasyon oluşturma işlemini yapabileceksiniz.

ARAŞTIRMA

Bu öğrenme faaliyetinden önce aşağıdaki hazırlıkları yapmalısınız.

- Resim maskeleyme yöntemlerini hatırlayınız.
- Dinamik olarak bileşen yüklemeyi öğreniniz.
- Ekran koruyucuların yazım tekniklerini araştırınız.

3. CANLANDIRMA (ANİMASYON)

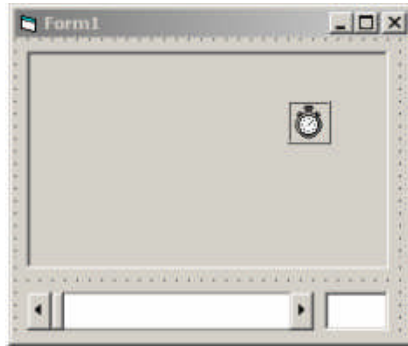
Animasyon, görüntü ve resimleri bir biri ardına hızlı bir şekilde getirerek canlıymış izlenimini vermektir. Günlük hayatta çok sık karşılaştığımız canlandırma, gerçek hayata daha yakın olduğu için insanı etkilemektedir. Hayat, zaten daimî harekettir.

Bu bölümde basit canlandırma işlemlerinin nasıl yapıldığını göreceğiz.

3.1. Çizgilerle Canlandırma

ÖRNEK 3.1: Timer ile çizgi döndürme.

ADIM 1: Form tasarımını şekilde görüldüğü gibi yapılıır. Bileşenlerin özellikleri tablodan yararlanılarak ayarlanır.

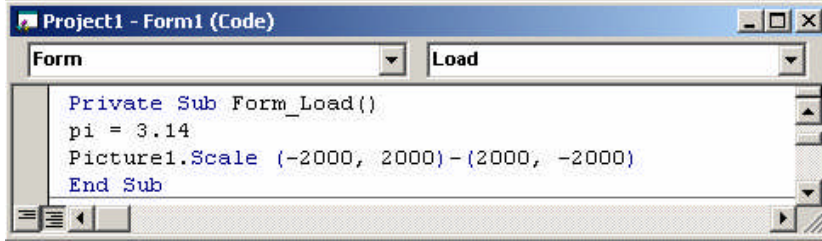


Şekil 3.1: Form tasarımı

Kontrol İsmi	Özellik	Değer
Picture1	Width	3500
	Heigth	3500
	AutoRedraw	True
HScroll1	Min	0
	Max	360
Timer1	Interval	100

Tablo 3.1: Kontroller ve özellikleri

ADIM 2: Formun Load olayı.



```

Project1 - Form1 (Code)
Form Load
Private Sub Form_Load()
    pi = 3.14
    Picture1.Scale (-2000, 2000)-(2000, -2000)
End Sub

```

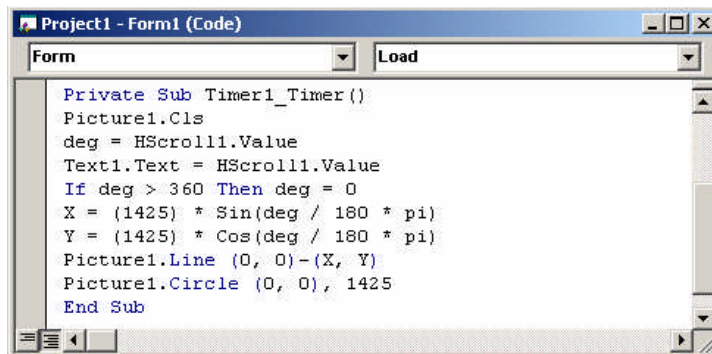
Burada Scale metodu ile daha önce de görüldüğü gibi yeni bir koordinat sistemi tanımlanmaktadır. Hatırlarsak kullanımı:

Scale(x1,y1,x2,y2) şeklindedir. Burada sol üst köşe ve sağ alt köşe tanımlanmaktadır. Örneğin form üzerinde sol üst köşeyi (123,234) ve sağ alt köşeyi (1123,1234) olarak kabul edecek yeni bir koordinat sistemi tanımlanabilir.

Form1.Scale (123,234, 1123,1234)

Bu işlemden sonra uygulanacak metotlar bunu esas alır.

ADIM 3: Kod listesi

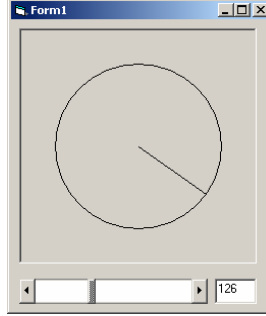


```

Project1 - Form1 (Code)
Form Load
Private Sub Timer1_Timer()
    Picture1.Cls
    deg = HScroll11.Value
    Text1.Text = HScroll11.Value
    If deg > 360 Then deg = 0
    X = (1425) * Sin(deg / 180 * pi)
    Y = (1425) * Cos(deg / 180 * pi)
    Picture1.Line (0, 0)-(X, Y)
    Picture1.Circle (0, 0), 1425
End Sub

```

ADIM 4: Program çalıştırılır.



Şekil 3.2: Programın çalışması

Daire üzerindeki noktaları bulmak için kutupsal koordinatlar kullanıldı.

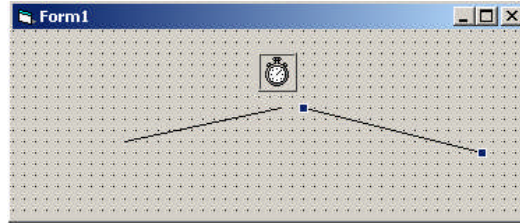
$$X = R * \text{Cos}(\text{theta})$$

$$Y = R * \text{Sin}(\text{theta})$$

Buradaki theta, raydan cinsinden açıdır.

ÖRNEK 3.2: Line bileşeni ile dönen iki çizgi.

ADIM 1: Form tasarımı tablodan yararlanılarak yapılır.



Şekil 3.3: Form tasarımı

Kontrol	Özellik	Değer
Form1	Width	5640
	Heigth	3855
Line1	X1	1200
	Y1	2880
	X2	1680
	Y2	1320
Line2	X1	3120
	Y1	5040
	X2	1320
	Y2	1800
Timer1	Interval	100

Tablo 3.2: Kontroller ve özellikleri

ADIM 2: Değişken bloğu

```
Private deg, pi
Private Const D_THETA = 3.14159265 / 12
Private Theta As Single
Private R1 As Single
Private R2 As Single
Private Cx1 As Single
Private Cy1 As Single
Private Cx2 As Single
Private Cy2 As Single
```

ADIM 3: Formun Load olayına başlangıç değerleri verilir.

```
Private Sub Form_Load()
Dim dx As Single
Dim dy As Single
dx = Line1.X2 - Line1.X1
dy = Line1.Y2 - Line1.Y1
R1 = Sqr(dx * dx + dy * dy)
Cx1 = Line1.X1
Cy1 = Line1.Y1
dx = Line2.X2 - Line2.X1
dy = Line2.Y2 - Line2.Y1
R2 = Sqr(dx * dx + dy * dy) / 2
Cx2 = (Line2.X1 + Line2.X2) / 2
Cy2 = (Line2.Y1 + Line2.Y2) / 2
End Sub
```

Burada çizginin uzunluğu bulunmakta ve Pisagor teoreminden hipotenüsü hesaplanarak ilgili değişkene aktarılmaktadır. Cx1, Cy1 değişkenlerine birinci çizginin başlangıç noktaları, Cx2, Cy2 değişkenlerine ise de ikinci çizginin orta noktası yüklenmektedir.

ADIM 4: Çizginin dönmesi Timer bileşeninde sağlanmaktadır.

```
Private Sub Timer1_Timer()
Theta = Theta + D_THETA
Line1.X2 = Cx1 + Cos(Theta) * R1
Line1.Y2 = Cy1 + Sin(Theta) * R1
Line2.X1 = Cx2 + Cos(Theta) * R2
Line2.Y1 = Cy2 + Sin(Theta) * R2
Line2.X2 = Cx2 - Cos(Theta) * R2
Line2.Y2 = Cy2 - Sin(Theta) * R2
End Sub
```

Burada çizginin uzunluğu trigonometrik fonksiyonlardan yararlanarak hesaplanmaktadır.

$$\text{Line1.X2} = \text{Cx1} + \text{Cos}(\text{Theta}) * \text{R1}$$

$$\text{Line1.Y2} = \text{Cy1} + \text{Sin}(\text{Theta}) * \text{R1}$$

satırları ile birinci çizginin yatay ve dikey bileşenleri hesaplanarak çizginin diğer ucu bulunmaktadır.

$$\text{Line2.X1} = \text{Cx2} - \text{Cos}(\text{Theta}) * \text{R2}$$

$$\text{Line2.Y1} = \text{Cy2} - \text{Sin}(\text{Theta}) * \text{R2}$$

$$\text{Line2.X2} = \text{Cx2} + \text{Cos}(\text{Theta}) * \text{R2}$$

$$\text{Line2.Y2} = \text{Cy2} + \text{Sin}(\text{Theta}) * \text{R2}$$

satırlarında ikinci çizginin yatay ve dikey bileşenleri Cx2, Cy2 orta noktasına eklenerek ve çıkarılarak çizginin başlangıç ve bitiş noktası bulunmaktadır. İkinci çizgi orta noktasından geçen eksen etrafında dönmektedir.

ADIM 4: Program çalıştırılır.

3.2. Image Kontrol ile Basit Animasyon

Image kontrolü, Bitmap, icon ve wmf türü dosyaları göstermek için kullanılır. PictureBox kadar gelişmiş özelliklere sahip olmasa da sadece resimlerin gösterilmesi istenen yerlerde tercih edilir. Resim yüklemek için Picture, resme göre uzayıp esnemesini sağlayan Stretch, görünür-görünmez olmasını sağlayan Visible özellikleri sık kullanılır.

Bu ikon alet kutusunun alt satırlarında yer almaktadır.



Şekil 3.4: Image kontrol

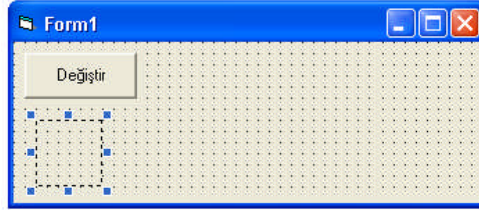
ÖRNEK 3.3: Trafik ışıklarına göre bir arabanın ilerlemesini canlandırma.

Bunun için aşağıdaki adımlar takip edilir.

ADIM 1: Form üzerine bir düğme ve bir image kontrol bileşeni koyulur ve tablodan yararlanarak özellik ataması yapılır.

Kontrol	Özellik	Değer
CommandButton	Name	cmdDegistir
	Caption	Değiştir
Image Control	Name	ImgLamba
	Height	720
	Width	720

Tablo 3.3: Kontroller ve özellikleri

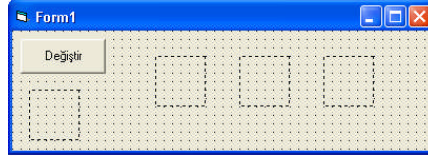


Şekil 3.5: Form tasarımı

Image bileşenin Stretch özelliği True yapılır. Böylelikle içindeki resmi orantılı olarak kapsamaya sağlanır. Bu bileşen arabanın daha doğrusu sürücünün yanmasını beklediği trafik lambası olacaktır.

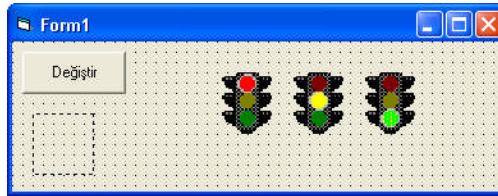
ADIM 2: Forma üç tane daha ImgLamba ile aynı ölçülerde image kontrolü koyun. Bunun için hepsinin tek tek ya da topluca Height ve Width özellikleri düzenlenebilir. Her üçünün de Stretch özellikleri True, Visible özellikleri False yapılır. Çalışma anında görünmeleri engellenir.

İsimlerini sırayla
imgYesil,
imgSari,
imgKirmizi olarak belirlenir.



Şekil 3.6: Form tasarımı

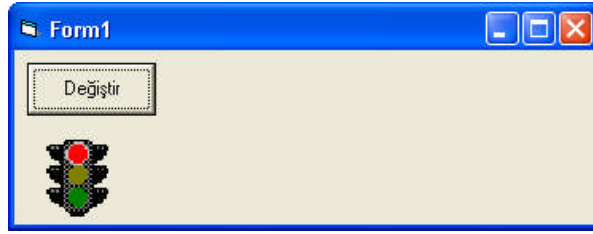
ADIM 4:Trafik lamba şekilleri sırayla image bileşenlerine atanacaktır. Bu amaçla Picture özelliklerine, “C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Traffic “ klasöründe bulunan TRFFC10C, TRFFC10B, TRFFC10A ikonları yüklenir.



Şekil 3.7: Form tasarımı

ADIM 5: Bu üç ışıktan kırmızı olanı program çalıştığında görünmesi istenirse formun Load olayına aşağıdaki kod yazılır.

```
Private Sub Form_Load()  
imgLamba.Picture = imgKirmizi.Picture  
End Sub
```



Şekil 3.8: Program çıktısı

ADIM 6: Düğmeye basıldıkça ışığın sırayla kırmızıdan yeşile dönmesi sağlanır.

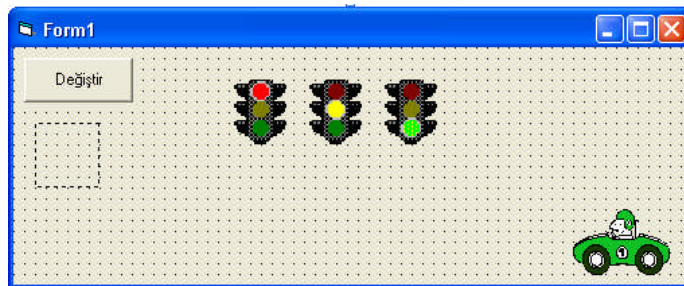
```
Private Sub cmdDegistir_Click()  
Select Case imgLamba.Picture  
Case imgKirmizi.Picture  
imgLamba.Picture = imgSari.Picture  
Case imgSari.Picture  
imgLamba.Picture = imgYesil.Picture  
Case imgYesil.Picture  
imgLamba.Picture = imgKirmizi.Picture  
End Select  
End Sub
```

Burada Select Case kullanılan yapısında imgLamba'nın Picture özelliği, diğerlerinin Picture özellikleri ile karşılaştırılıyor. Aynıysa bir sonraki ile değiştiriyor.

ADIM 7: Program çalıştırılarak denenir.

ADIM 8: Bu aşamada form üzerine temsili bir araba konularak yeşil ışık yandığında geçmesi sağlanacaktır.

Araba, başlangıç olarak formun sağında görünecek, sol tarafa doğru gözden kaybolana kadar ilerleyecek. Bu amaçla formun sağ kenarından tutarak genişletin ve bir image kontrolü yerleştirilir. Stretch özelliği True yapılır. Picture özelliğine C:\Program Files\Microsoft Office\media\cagcat10 klasöründe bulunan J0212957.wmf resim dosyası yüklenir. Bunu beğenmezseniz yada bulamadıysanız yine C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Industry klasöründe bulunan "Cars.Ico" dosyası yüklenebilir.



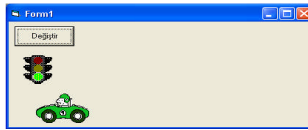
Şekil 3.9: Form tasarımı

ADIM 9: İsmi imgAraba olarak verilir. Formun sağ ucundan arabayı gizleyecek şekilde sürüklenir. Artık sıra arabayı sola yürütecek koda geldi. Bunun için Do While Loop döngüsü içinde imgAraba'nın Left özelliği azaltılır. Lamba yeşil yanar yanmaz arabanın hareketi gerektiği için kod ilgili lambanın altına yazılır.

```
Private Sub cmdDegistir_Click()  
Select Case imgLamba.Picture  
Case imgKirmizi.Picture  
imgLamba.Picture = imgSari.Picture  
Case imgSari.Picture  
imgLamba.Picture = imgYesil.Picture  
Do While ImgAraba.Left > 0  
ImgAraba.Left = ImgAraba.Left - 2  
DoEvents  
Loop  
Case imgYesil.Picture  
imgLamba.Picture = imgKirmizi.Picture  
End Select  
End Sub
```

ADIM 10: Program çalıştırılır.

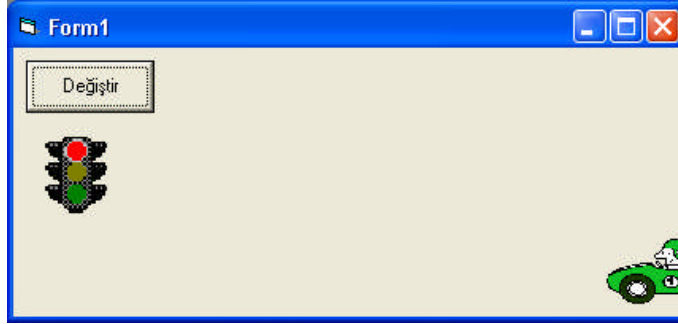
```
Private Sub cmdDegistir_Click()  
Dim Sakla As Single  
Select Case imgLamba.Picture  
Case imgKirmizi.Picture  
imgLamba.Picture = imgSari.Picture  
Case imgSari.Picture  
imgLamba.Picture = imgYesil.Picture  
Sakla = ImgAraba.Left  
Do While ImgAraba.Left > 0  
ImgAraba.Left = ImgAraba.Left - 2  
DoEvents  
Loop  
ImgAraba.Left = Sakla  
Case imgYesil.Picture  
imgLamba.Picture = imgKirmizi.Picture  
End Select  
End Sub
```



Şekil 3.10: Programın çalışması

ADIM 11: Bir şey dikkatimizi çekti. Araba sol kenara yanaştığında lambanın ışığını dikkate almadı. Lamba kırmızıya geçtiğinde tekrar arabanın sol tarafta yerini alması istenirse kodun biraz değiştirilmesi gerekir.

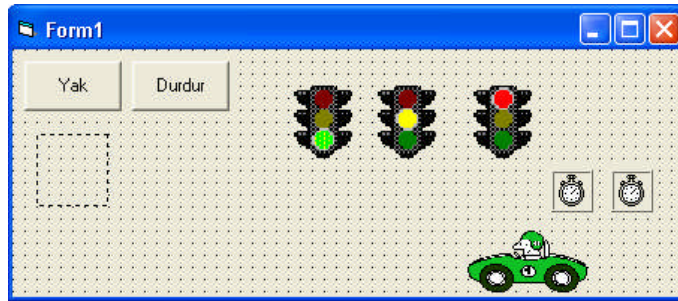
```
Private Sub cmdDegistir_Click()  
Select Case imgLamba.Picture  
Case imgKirmizi.Picture  
imgLamba.Picture = imgSari.Picture  
Case imgSari.Picture  
imgLamba.Picture = imgYesil.Picture  
Do While ImgAraba.Left > 0  
ImgAraba.Left = ImgAraba.Left - 2  
DoEvents  
Loop  
Case imgYesil.Picture  
imgLamba.Picture = imgKirmizi.Picture  
ImgAraba.Left = Form1.ScaleWidth - imgLamba.Width  
End Select  
End Sub
```



Şekil 3.11: Programın çalışması

ÖRNEK 3.4: Örnek 3.3'ün Timer ile yapılması.

ADIM 1: Bu amaçla form aşağıdaki gibi değiştirilir.



Şekil 3.12: Form tasarımı

Image kontrol bileşenleri aynı kalacak, yeni bileşenlere tabloda görülen atamalar yapılacaktır.

Kontrol	Özellik	Değer
CommandButton	Caption	Yak
	Name	cmdYak
CommandButton	Caption	Durdur
	Name	cmdDurdur
Timer	Name	tmrLamba
	Interval	0
	Enabled	False
Timer	Name	tmrAraba
	Interval	0
	Enabled	False

Tablo 3.4: Kontroller ve özellikleri

ADIM 2: Aşağıdaki kod yazılır.

```
Const LambaSure = 1000  
Const ArabaSure = 50  
Const Mesafe = 10
```

```
Private Sub cmdDurdur_Click()  
tmrLamba.Enabled = False  
tmrAraba.Enabled = False  
End Sub
```

```
Private Sub tmrAraba_Timer()  
ImgAraba.Left = ImgAraba.Left - Mesafe  
End Sub
```

```
Private Sub cmdYak_Click()  
tmrLamba.Enabled = True  
End Sub
```

```
Private Sub Form_Load()  
imgLamba.Picture = imgKirmizi.Picture  
tmrLamba.Interval = LambaSure  
tmrAraba.Interval = ArabaSure  
End Sub
```

```
Private Sub tmrLamba_Timer()  
Select Case imgLamba.Picture  
Case imgKirmizi.Picture  
imgLamba.Picture = imgSari.Picture  
Case imgSari.Picture
```

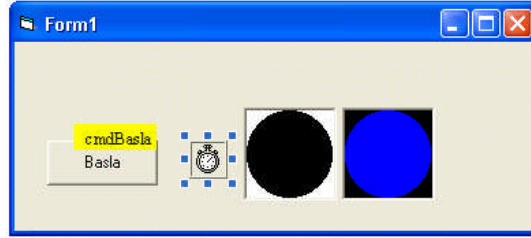
```

imgLamba.Picture = imgYesil.Picture
tmrAraba.Enabled = True
Case imgYesil.Picture
imgLamba.Picture = imgKirmizi.Picture
tmrAraba.Enabled = False
End Select
End Sub

```

ADIM 3: Program çalıştırılır.

ÖRNEK 3.5: Daha önce Öğrenme Faaliyeti 2, bölüm 2.9’da maskeleyen sprite resmi hareket ettirilmek istenirse form tasarımı aşağıdaki gibi değiştirilir. Sprite resmini hareket ettirecek kod, Timer nesnesinin içine yerleştirilmiştir.



Şekil 3.13: Form tasarımı

Option Explicit

```

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
Dim X As Long, Y As Long
Dim SpriteGenislik As Long
Dim SpriteYukseklik As Long
Private Sub cmdBasla_Click()
Timer1.Enabled = True
End Sub

Private Sub Form_Load()
'Her iki resim kutusunun boyutları aynı yap
SpriteGenislik = picSprite.ScaleWidth
SpriteYukseklik = picSprite.ScaleHeight
Timer1.Enabled = False
Timer1.Interval = 35
End Sub

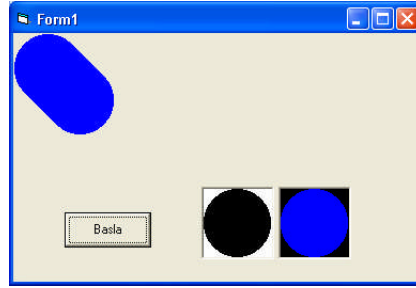
```

```

Private Sub Timer1_Timer()
Static X As Long, Y As Long
X = X + 1
Y = Y + 1
'Topu çalışma alanında tut
If X > Me.ScaleWidth Then
X = 0
End If
If Y > Me.ScaleHeight Then
Y = 0
End If
BitBlt Me.hDC, X, Y, SpriteGenislik, SpriteYukseklk, Picture1.hDC, 0, 0, vbSrcAnd
BitBlt Me.hDC, X, Y, SpriteGenislik, SpriteYukseklk, picSprite.hDC, 0, 0,
vbSrcPaint
End Sub

```

Program çalıştırıldığında mavi bir kuyruk uzar. Çünkü resim birer piksel kaydırılmakta fakat bir önceki sildirilmemektedir.



Şekil 3.14: Programın çalışması

Bunu önlemenin yolu, bir öncekinin sildirilmesidir. Bu da Form1.Cls yada Me.Cls komutu ile yapabilir. Bunun için aşağıdaki kod eklenir.

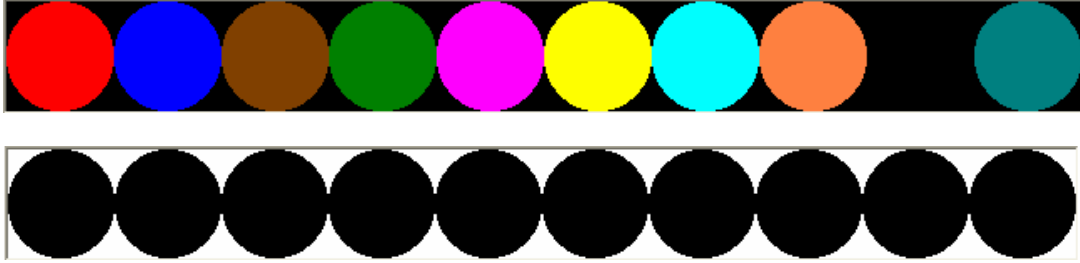
```

.....
If Y > Me.ScaleHeight Then
Y = 0
End If
'Formu Temizle
Me.Cls
BitBlt Me.hDC, X, Y, SpriteGenislik, SpriteYukseklk, Picture1.hDC, 0, 0, vbSrcAnd.
.....

```

Artık tek top hareket etmesine rağmen hâlâ bir sıkıntı var. Top hareket ederken titremektedir. Bilgisayar hızına bağlı olarak da siyah maske görünebilir. Bu, formun Cls metodundan sonra, yeniden çizilme zamanı ile sprite resminin çizilme anı eş zamanlı olmamaktadır. Sanki bir filmin bazı kısımları kesilmiş ve görüntü oraya geldiğinde titreme yapıyor. Bunu önlemenin bir yolu çizim yaptıktan sonra formu Refresh metodu ile tazelemektir. Tabii AutoRedraw özelliğini True yapmak kaydıyla.İlerde bu tür bir sorun başka yoldan çözülecektir.

Bu aşamada sorulacak soru şudur; Acaba dairenin rengi hareket ettikçe değiştirebilir mi? Neden olmasın. Kaç renk olması isteniyorsa o kadar maskesiyle beraber resim kutusu çizilir. Sırayla BitBlt fonksiyonu ile form üzerine gönderebilir. Fakat bu çok zaman alıcı ve hafıza tüketici bir yöntem olur. Bunun yerine farklı renkte daireler tek bir resim kutusuna çizilir. Aynı şekilde buna uygun maske de hazırlanır.

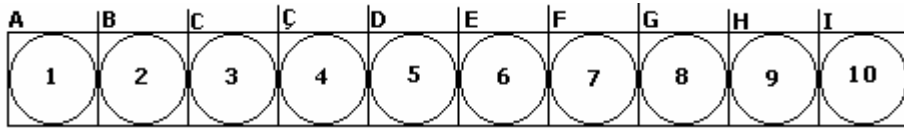


Şekil 3.15: Renkler ve maskeleri

Her bir daire çerçevesi belli bir boyuta sahiptir. 64x64 piksel gibi. BitBlt fonksiyonu, sadece kaynağın sol üst köşesine ve boyutuna ihtiyaç duyar. Tüm mesele belli zaman diliminde bu sol üst köşeyi sağa bir çerçeve kadar kaydırmaktır. Bu da çerçeve sayısına eş değer bir değişkenin değerini eşzamanlı arttırarak sağlanabilir.

X kaydırma miktarı şu formülle bulunur.

$$X=(\text{ÇerçeveSayısı}-1)*\text{Genişlik}$$

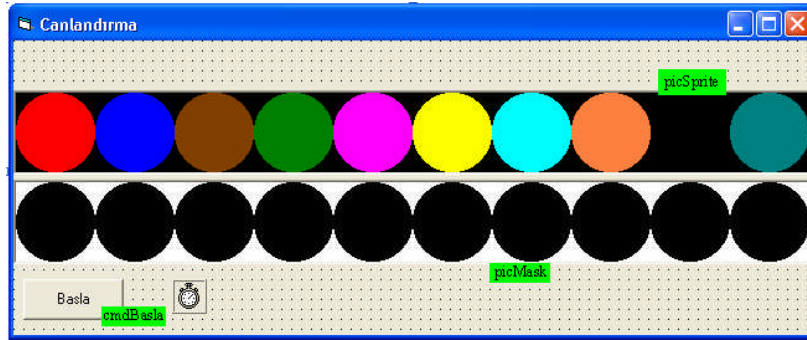


Şekil 3.16: Kaydırma genişliği

Her bir çerçevenin genişliğinin 64 bit olduğu kabul edilerek Şekil 43'te görülen sol üst köşe noktaları hesaplanır.

$$\begin{aligned} A &= (1-1)*64 = 0 \\ B &= (2-1)*64 = 64 \\ C &= (3-1)*64 = 128 \\ \text{Ç} &= (4-1)*64 = 192 \\ D &= (5-1)*64 = 256 \\ E &= (6-1)*64 = 320 \\ F &= (7-1)*64 = 384 \\ G &= (8-1)*64 = 448 \\ H &= (9-1)*64 = 512 \\ I &= (10-1)*64 = 576 \end{aligned}$$

Daha önce tek bir sprite ve maske çizilmişti. Şimdi şekilde görüldüğü gibi 10 tane sprite ve maske çizilecektir. Her ikisinin de aynı ölçüde olmasına dikkat edilmelidir.



Şekil 3.17: Form tasarımı

Bu temel bilgisiyi kullanarak Timer bileşeni altına aşağıdaki kod yazılır.

```
Private Sub Timer1_Timer()
Static X As Long, Y As Long

'Form yüzeyi temizlenir
Form1.Cls

'Maske çiz
BitBlt Form1.hDC, X, Y, SpriteGenisligi, SpriteYuksekligi, picMask.hDC, _
(CerceveNo - 1) * SpriteGenisligi, 0, vbSrcAnd

'Sprite çiz
BitBlt Form1.hDC, X, Y, SpriteGenisligi, SpriteYuksekligi, picSprite.hDC, _
(CerceveNo - 1) * SpriteGenisligi, 0, vbSrcPaint

'Çerçeve numarasını güncelle
CerceveNo = (CerceveNo Mod SonCerceveNo) + 1

'Çizim alanlarını güncelle
X = (X Mod Form1.ScaleWidth) + 1
Y = (Y Mod Form1.ScaleHeight) + 1

'Formu güncelle
Form1.Refresh
End Sub
```

Burada $CerceveNo = (CerceveNo \text{ Mod } SonCerceveNo) + 1$ satırı ile çerçeve sayısına sınırlama getirilmektedir. Bu durum bir örnekle açıklanırsa;

İlk çerçeve için:
 $CerceveNo: (1 \text{ Mod } 10) + 1 = 2$

Son çerçeve için:
 $CerceveNo: (10 \text{ Mod } 10) + 1 = 1$

Böylece tüm çerçeve, Timer bileşenin tik olayında taranmaktadır.

$$X = (X \text{ Mod Form1.ScaleWidth}) + 1$$
$$Y = (Y \text{ Mod Form1.ScaleHeight}) + 1$$

satırlarında ise daire, formun alanı içinde tutulmaktadır.

Fakat bu örnekte dairelerin renk değiştirme süresi Timer'ın Interval özelliğine atanan süre kadar olacaktır. Bu süre 1 saniyeye uzatılmak istenirse ne yapılacaktır? Bunun için daha önce de kullanılan GetTickCount API fonksiyonunun kullanılması gerekir. Kodun tamamı aşağıda görüldüğü gibidir.

```
Option Explicit
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 GetTickCount Lib "kernel32" () As Long
```

```
Const SpriteGenisligi As Long = 64
Const SpriteYuksekligi As Long = 64
Const SonCerceveNo As Long = 10
Const GecisSuresi As Long = 1000
```

```
Dim CerceveNo As Long
Dim SimdikiAn As Long
Dim IlkAn As Long
```

```
Private Sub cmdBasla_Click()
Timer1.Enabled = Not (Timer1.Enabled)
IlkAn = GetTickCount()
CerceveNo = 1
End Sub
```

```
Private Sub Form_Load()
Form1.AutoRedraw = True
Timer1.Enabled = False
Timer1.Interval = 60
picSprite.Visible = False
picMask.Visible = False
Form1.ScaleMode = vbPixels
Form1.Width = 5120
End Sub
```

```
Private Sub Timer1_Timer()
Static X As Long, Y As Long
```

```

'Form yüzeyi temizlenir
Form1.Cls
'Maske çiz
BitBlt Form1.hDC, X, Y, SpriteGenisligi, SpriteYuksekligi, picMask.hDC, _
(CerceveNo - 1) * SpriteGenisligi, 0, vbSrcAnd

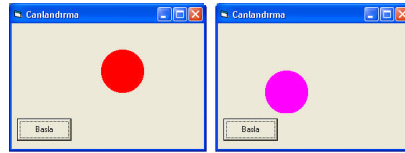
'Sprite çiz
BitBlt Form1.hDC, X, Y, SpriteGenisligi, SpriteYuksekligi, picSprite.hDC, _
(CerceveNo - 1) * SpriteGenisligi, 0, vbSrcPaint

SimdikiAn = GetTickCount()

'Diğer çerçeveye geçip geçmeyeceğimizi murakabe et
If SimdikiAn - IlkAn > GecisSuresi Then
'Çerçeve numarasını güncelle
CerceveNo = (CerceveNo Mod SonCerceveNo) + 1
IlkAn = GetTickCount()
End If
'Çizim alanlarını güncelle
X = (X Mod Form1.ScaleWidth) + 1
Y = (Y Mod Form1.ScaleHeight) + 1
'Formu güncelle
Form1.Refresh
End Sub

```

Program çalıştırılır.



Şekil 3.18: Programın çalışması

GetTickCount() fonksiyonu düğmenin tıklanması ile o anı (daha doğrusu Window'un çalışmaya başladığı andan itibaren süre milisaniye cinsinden geçen süreyi) IlkAn değişkenine atamaktadır.

```
IlkAn = GetTickCount()
```

Timer1_Tick yordamı içinde de yine belli bir anı almaktadır.

```
SimdikiAn = GetTickCount()
```

Arada geçen süre, program başlangıcında “Const GecisSuresi As Long = 1000” satırı ile sabit olarak verilen 1000 milisaniyelik (1 san) süreden fazla ise çerçeveyi güncelleyecektir.

```
If SimdikiAn - IlkAn > GecisSuresi Then
'Çerçeve numarasını güncelle
CerçeveNo = (CerçeveNo Mod SonCerçeveNo) + 1
IlkAn = GetTickCount()
End If
```

SimdikiAn değeri IlkAn değerine atanıyor ki bundan sonra bundan sonra çerçevenin güncellenme süresi artık IlkAn değişkeninde tutulsun.

3.3. StretchBlt API Fonksiyonu

Yukarıdaki örneklerde daire hem hareket ettirildi, hem de rengi değiştirildi. Peki bu dairenin büyüüp küçülmesi istenirse ne yapmak gerekir? Bu durumda StretchBlt fonksiyonunu öğrenmek, sorunun çözümüne katkıda bulunacaktır.

StretchBlt, sprite resimlerini küçültür veya büyültür. BitBlt fonksiyonuna benzerliği vardır. Her ikisi de kaynak ve hedef alanın Device Context(DC) numarasını ve raster işlemlerini kullanır. Farklı olarak StretchBlt, kaynak alanı, hedef alan ölçüsüne uydurmak için daraltır ya da genişletir. Bildirimi :

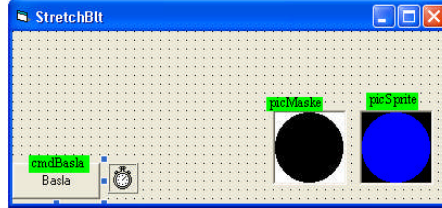
```
Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, _
ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, _
ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, _
ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As
Long
```

Burada:

hdc: Hedef DC
X: Hedef alanın sol üst köşesinin X koordinatı
Y: Hedef alanın sol üst köşesinin Y koordinatı
nWidth: Hedef alanın genişliği
nHeight: Hedef alanın yüksekliği
hSrcDC: Kaynak DC
xSrc: Kaynak alanın sol üst köşesinin X koordinatı
ySrc: Kaynak alanın sol üst köşesinin Y koordinatı
nSrcWidth: Kaynak alanın genişliği
nSrcHeight: Kaynak alanın yüksekliği
dwRop: Raster işlemleri

ÖRNEK 3.6: Bir önceki örnekteki dairenin büyültülüp küçültülmesi.

Bu amaçla form aşağıdaki gibi tasarlanır.



Şekil 3.19: Form tasarımı

Programın kod bölümü:

```
Private 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
```

```
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
```

```
Dim Buzulme As Boolean  
Dim Genlesme As Long
```

```
Const SpriteGenislik As Long = 64  
Const SpriteYukseklk As Long = 64  
Const AzamiGenlesme As Long = 96
```

```
Private Sub cmdBasla_Click()  
Genlesme = 64  
Buzulme = False  
Timer1.Enabled = Not (Timer1.Enabled)  
End Sub
```

```
Private Sub Form_Load()  
Form1.AutoRedraw = True  
Timer1.Enabled = False  
Timer1.Interval = 60  
picSprite.Visible = False  
picMaske.Visible = False  
Form1.ScaleMode = vbPixels  
Form1.Width = 5120
```

```

Form1.Height = 4200
End Sub

Private Sub Timer1_Timer()
Static X As Long, Y As Long
Me.Cls

If Buzulme Then
Genlesme = Genlesme - 2
Else
Genlesme = Genlesme + 2
End If

If Genlesme < 32 Then Buzulme = False
If Genlesme > AzamiGenlesme Then Buzulme = True

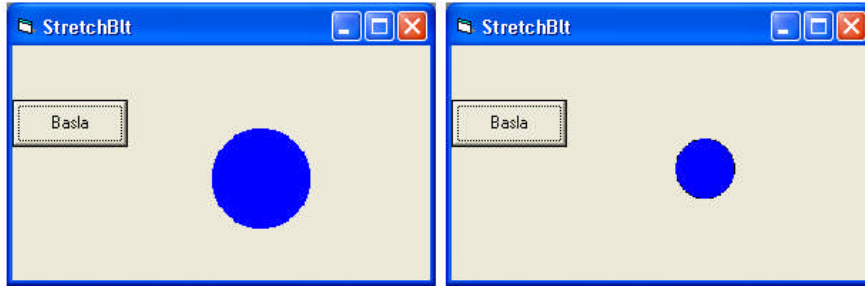
StretchBlt Me.hdc, X, Y, Genlesme, Genlesme, picMaske.hdc, 0, 0, _
SpriteGenislik, SpriteYukseklk, vbSrcAnd

StretchBlt Me.hdc, X, Y, Genlesme, Genlesme, picSprite.hdc, 0, 0, _
SpriteGenislik, SpriteYukseklk, vbSrcPaint

X = (X Mod Me.ScaleWidth) + 2
Y = (Y Mod Me.ScaleHeight) + 2

Me.Refresh
End SubProgram çalıştırılır.

```



Şekil 3.20: Programın çalışması

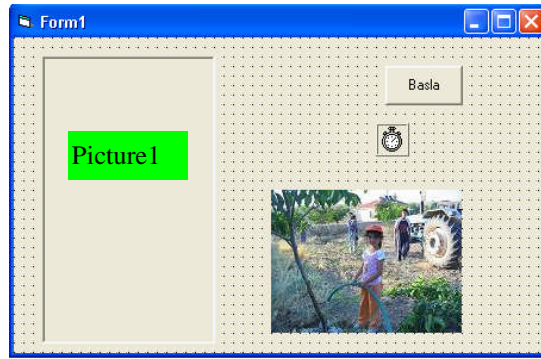
İlk yapılan, genleşme değerinin test edilmesidir. Bu değer daralma ve genleşme durumlarında olabilir. Farklı durumları teşhis için Buzulme isminde boolean değişkeni tanımlanmıştır. Genleşme katsayısı 32'den az, 64'ten büyük olamaz. Bu değerleri aşarsa değişken durum değiştirerek zıddı kullanılır.

Sprite, daha önce anlatıldığı gibi çizilir. Önce maskelenir ardından sprite çizilir. Tek fark BitBlt yerine Stretch kullanılır.

ÖRNEK 3.7: Bir resmi aşağı yukarı hareket ettirme.

ADIM 1: Form üzerine iki resim kutusu, bir Timer ve bir düğme yerleştirilir.

Resim kutusunun birisi büyük diğeri küçük yapılır. Her ikisinin ScaleMode özelliği piksele ayarlanır. Picture özelliğinden yararlanarak küçük olana bir resim yüklenir. Yine bunun Border özelliği "None" yapılır. Timer bileşenin Enabled özelliği False, Interval özelliği 113 olarak belirlenir.



Şekil 3.21: Form tasarımı

ADIM 2: Aşağıdaki kodu yazılır.

```
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 Const SRCCOPY = &HCC0020  
Dim ResimY, ResimYonu As Integer
```

```
Private Sub Command1_Click()  
Timer1.Enabled = Not (Timer1.Enabled)  
If Timer1.Enabled = True Then Picture2.Visible = False  
End Sub
```

```
Private Sub Form_Load()  
ResimY = 0  
ResimYonu = 1  
End Sub
```

```
Private Sub Timer1_Timer()  
Static ResimY As Long  
Picture1.Cls  
ResimY = ResimY + ResimYonu * Picture1.ScaleWidth / 50  
If ResimY < 0 Then
```

```

ResimY = 0
ResimYonu = 1
ElseIf ResimY + Picture2.ScaleHeight > Picture1.ScaleHeight Then
ResimY = Picture1.ScaleHeight - Picture2.ScaleHeight
ResimYonu = -1
End If

```

```

BitBlt Picture1.hDC, CLng(0.5 * (Picture1.ScaleWidth - Picture2.ScaleWidth)), _
ResimY, CLng(Picture2.ScaleWidth), _
CLng(Picture2.ScaleHeight), Picture2.hDC, CLng(0), _
CLng(0), SRCCOPY
End Sub

```

ADIM 3: Program çalıştırılır. Görüldüğü gibi resim bir aşağı bir yukarı gezinmeye başladı.



Şekil 3.22: Programın çalışması

Resmin aşağı yukarı hareket etmesi “ResimY” değişkeni ile sağlanmaktadır. Resim X yönünde hareket etmediği için sadece bu değişkenin artırılıp azaltılması yeterlidir. Başlangıçta Picture1.ScaleHeight=330 ve Picture2.ScaleHeight=120 olduğu kabul edilir.

Program başlangıcında ResimY=0’dır.
 $ResimY = ResimY + ResimYonu * Picture1.ScaleWidth / 50$
 $ResimY = 0 + (1 * 330) / 50 \rightarrow 6.6$

Sıfırdan büyük olduğu için

If ResimY < 0 Then

ResimY = 0

ResimYonu = 1

satırlarını atlayacak.

Yine

ElseIf ResimY + Picture2.ScaleHeight > Picture1.ScaleHeight Then

şartı sağlanmadığından (çünkü $6.6 + 120 < 330$) bu satırlarda işletilmeyecektir. BitBlt fonksiyonunda resmin çizileceği Y koordinat 6.6 olacaktır. Bir sonraki döngüde de aynı şey olacak bu sefer ResimY=13. olacak ve resim sürekli olarak aşağı inecektir. Tâ ki $(ResimY + Picture2.ScaleHeight > Picture1.ScaleHeight)$

ifadesi doğru olana kadar. Bu sefer ResimYonu = -1 olacağından resim yukarı çıkmaya başlayacaktır.

Şimdi resmin aşağı yukarı çarpma anında bir ses çıkarmasını sağlayalım. Bunun için

```
Declare Function sndPlaySound Lib "winmm.dll" Alias "sndPlaySoundA" _  
(ByVal lpszSoundName As String, ByVal uFlags As Long) As Long
```

API fonksiyonu kullanılır. İlk parametresi (lpszSoundName) çalınacak wav uzantılı ses dosyasını, ikinci parametre ise sesi nasıl çalacağını gösterir. Bunun alabileceği sabitler:

```
SND_SYNC = &H0  
SND_ASYNC = &H1  
SND_NODEFAULT= &H2  
SND_MEMORY= &H4  
SND_LOOP= &H8  
SND_NOSTOP= &H10
```

Burada sadece ilk iki parametre açıklanacaktır.

SND_SYNC: Ses dosyasının çalınması bitinceye kadar fonksiyon, geriye bir değer döndürmez.

SND_ASYNC : Ses dosyası çalınmaya başladığı anda geriye bir değer döndürülür.

Aşağıdaki satırlar çalışma dizininde bulunan "engine.wav" dosyasını çalmaktadır.

```
Private Declare Function sndPlaySound Lib "winmm.dll" Alias _  
"sndPlaySoundA" (ByVal lpszSoundName As String, ByVal uFlags As Long) As  
Long
```

```
Const SND_SYNC = &H0
```

```
Private Sub Command1_Click()  
x = sndPlaySound("engine.wav", SND_ASYNC)  
End Sub
```

Artık yukarıdaki örneğe ses dosyası eklenebilir.

ÖRNEK 3.8: Ses Ekleme

Bu amaçla kod yeniden düzenlenir.

```
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 sndPlaySound Lib "winmm.dll" Alias _
"sndPlaySoundA" (ByVal lpszSoundName As String, ByVal uFlags As Long) As
Long
```

```
Private Const SND_ASYNC = &H1
Dim BongSound As String
Dim ResimY, ResimYonu As Integer
Private Sub Command1_Click()
Timer1.Enabled = Not (Timer1.Enabled)
End Sub
```

```
Private Sub Form_Load()
ResimY = 0
ResimYonu = 1
BongSound = ("C:\WINDOWS\Media\Windows XP Ding.wav")
End Sub
```

```
Private Sub Timer1_Timer()
Static ResimY As Long
Dim GeriDonDeger As Long
Picture1.Cls
ResimY = ResimY + ResimYonu * Picture1.ScaleWidth / 50
If ResimY < 0 Then
ResimY = 0
ResimYonu = 1
sndPlaySound BongSound, SND_ASYNC
ElseIf ResimY + Picture2.ScaleHeight > Picture1.ScaleHeight Then
ResimY = Picture1.ScaleHeight - Picture2.ScaleHeight
ResimYonu = -1
sndPlaySound BongSound, SND_ASYNC
End If
```

```
BitBlt Picture1.hDC, CLng(0.5 * (Picture1.ScaleWidth - Picture2.ScaleWidth)), _
ResimY, CLng(Picture2.ScaleWidth), _
CLng(Picture2.ScaleHeight), Picture2.hDC, CLng(0), _
CLng(0), SRCCOPY
End Sub
```

3.4. Titremeyi Önleme

Resim kayarken oluşan titremeyi giderilmek için AutoRedraw ve Refresh metotları kullanılmıştı. Bu yöntem, daima iki resim kutusu ile beraber çalışmaya dayanır. Bunların her ikisi de aynı özellikte olmalarına karşın tek fark birisi görünür, birisi görünmez olur. Görünmeyen resim kutusu esas çalışma yeridir. Canlandırma ile alakalı tüm işler bunun üzerinde yapılır. Her şey yerli yerinde olduğunda BitBlt ile görünen resim kutusuna kopyalanır.

Önceki örnekte aşağıda görüldüğü gibi bir değişiklik yapılır.

- Picture1'in Index özelliği 0 yapılır. Bu onu, istenildiği kadar kopyası alınabilen bir dizi haline getirecektir. Bizim için sadece iki resim kutusu olacağından Picture1(0), görünen resim kutusu, Picture1(1) ise çalışma alanı olan görünmeyen resim kutusu olacaktır.
- Form_Load() olayına

```
Load Picture1(1)
Picture1(1).AutoRedraw = True
```

Satırlarını ilave ederek Picture1(1) oluşturulur.

- Timer1_Timer olayında gerekli değişiklikler yapılır. İlk BitBlt ile resmi barındıran Picture2'den görünmeyen resim kutusuna kopyalanır; buradan ikinci BitBlt ile görünen resim kutusuna gönderilir.

```
Private Sub Timer1_Timer()
Static ResimY As Long
Picture1(1).Cls
ResimY = ResimY + ResimYonu * Picture1(1).ScaleWidth / 50
If ResimY < 0 Then
ResimY = 0
ResimYonu = 1
sndPlaySound BongSound, SND_ASYNC
ElseIf ResimY + Picture2.ScaleHeight > Picture1(1).ScaleHeight Then
ResimY = Picture1(1).ScaleHeight - Picture2.ScaleHeight
ResimYonu = -1
sndPlaySound BongSound, SND_ASYNC
End If

BitBlt Picture1(1).hDC, CLng(0.5 * (Picture1(1).ScaleWidth - Picture2.ScaleWidth)),
_
ResimY, CLng(Picture2.ScaleWidth), CLng(Picture2.ScaleHeight), _
Picture2.hDC, CLng(0), CLng(0), SRCCOPY

BitBlt Picture1(0).hDC, CLng(0), CLng(0), CLng(Picture1(1).ScaleWidth), _
CLng(Picture1(1).ScaleHeight), Picture1(1).hDC, CLng(0), CLng(0), SRCCOPY
End Sub
```

ÖRNEK 3.9: Zıplayan top.

Fizik dersinde görülen eğik atış formüllerini kullanarak bir topun zıplaması olayının canlandırılması. Hatırlanılacağı gibi:

Yatay bileşen: $x = Vx0 * t + x0$

Dikey Bileşen: $y = Vy0 * t - g * t^2 / 2$

Form üzerine bir resim kutusu, bir Timer ve bir de düğme yerleştirilir. Aşağıdaki kod yazılır.

```
Option Explicit
Dim t As Single, x0 As Single, y0 As Single, vx0 As Single
Dim vy0 As Single, Azalt As Single
Const g = 9.8
```

```
Private Sub Command1_Click()
If Timer1.Enabled = True Then
Timer1.Enabled = False
Else
t = 0
vx0 = 10
vy0 = 40 '(m/s)
Timer1.Enabled = True
End If
End Sub
```

```
Private Sub Form_Load()
Timer1.Enabled = False
Timer1.Interval = 100
```

```
With Picture1
.AutoRedraw = True
.BackColor = &HFFFFFF
.ScaleMode = 0
.Width = 6000
.Height = 3000
.ScaleWidth = 200
.ScaleHeight = -100
.ScaleTop = 98
.ScaleLeft = -50
End With
```

```
x0 = -50
y0 = 0
Azalt = 0.7
End Sub
```

```
Private Sub Timer1_Timer()
Dim x As Single, y As Single
```

```
x = vx0 * t + x0
y = vy0 * t - g * t ^ 2 / 2
Picture1.Cls
```

```

Picture1.FillStyle = 0
Picture1.FillColor = &HFF
Picture1.Circle (x, y), 5, &HFF
If t > Timer1.Interval / 1000 And y <= 0 Then
t = 0
x0 = x
y0 = y
vy0 = vy0 * Azalt
vx0 = vx0 * Azalt
End If

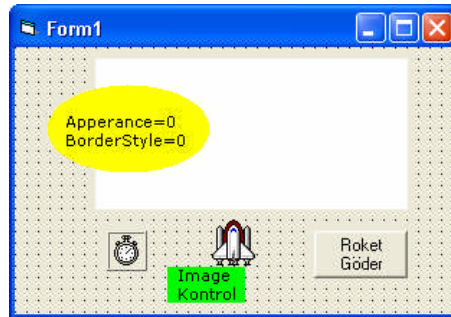
t = t + Timer1.Interval / 1000
End Sub

```

Program çalıştırılır ve zıplayan bir top görülür.

ÖRNEK 3.9: Roket gönderilmesi.

Form üzerine bir image kontrol, bir düğme, bir timer ve bir picturebox koyulur.



Şekil 3.23: Form tasarımı

Image kontrole resim olarak \Microsoft Visual Studio\Common\Graphics \Icons \Industry klasöründe bulunan rocket.ico resmi yüklenebilir.

Kod listesi aşağıda verilmiştir.

```
Dim cx, cy As Integer
```

```

Private Sub Command1_Click()
Image1.Top = Form1.ScaleHeight - 10
Image1.Left = cx - 16
Image1.Visible = True
Timer1.Interval = 100
Timer1.Enabled = True
End Sub

```

```
Private Sub Form_Load()
```

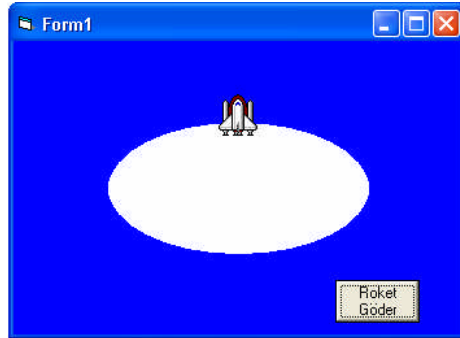
```

Picture1.Visible = False
Image1.Visible = False
With Form1
Show
AutoRedraw = True
ScaleMode = 3
cx = .ScaleWidth / 2
cy = .ScaleHeight / 2
Line (0, 0)-(cx * 2, cy * 2), QBColor(9), BF
FillColor = QBColor(15)
FillStyle = 0
Circle (cx, cy), 100, QBColor(15), , , 0.5
End With
End Sub

Private Sub Timer1_Timer()
Image1.Top = Image1.Top - 5
End Sub

```

Programı çalıştırılır.



Şekil 3.24: Programın çalışması

3.5. Shape Kontrolü ile Canlandırma

Bazen form üzerine kontroller çalışma anında konular. Kontrollerin hepsi programı derleme anında hafızaya yüklenmez, ihtiyaç duyuldukça sahneye davet edilir. Bu örnekte bununla ilgili olarak timer ve shape kontrolleri çalışma anında dinamik olarak programa dahil edilecektir. Kendine ait olayları olan bir kontrol, WithEvents deyimini ile tanıtılır. Sadece özellikleri ve metotları olan kontroller ise değişkenmiş gibi tanıtılır.

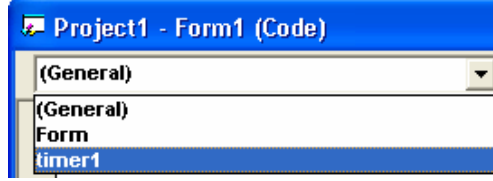
Programda bir timer ve shape bileşeni olduğundan;

```

Dim WithEvents timer1 As Timer
Dim shape1 As Shape

```

Burada timer1 ve shape1 adında bileşenler tanıtıldı. Kodu yazar yazmaz ismi, nesnelere gösteren kutuda görüldü.



Şekil 3.25: Bileşen kutusu

Dinamik kontroller ya Form_Activate olayında ya da daha sonra herhangi bir olaya bağlı olarak oluşturulurlar. Bunun için ilgili olduğu yerde

```
Set timer1 = Controls.Add("VB.Timer", "timer1")
```

satırını yazmak gerekir. Form_Activate olayı Form_Load olayından önce geldiğinden önce kontrol oluşturulur arkasından buna ait özellikler belirlenir. Fakat Form_Load olayı altında da kullanılmadan önce oluşturulursa bir sakınca olmaz. Ama ilki, daha iyi bir yoldur. Burada ikincisi tercih edilecektir. Form_Load kod listesi:

```
Private Sub Form_Load()  
Randomize  
Form1.ScaleMode = 3  
Form1.AutoRedraw = False  
Form1.Caption = "Shapes"  
Form1.Show  
DoEvents  
"Timer kontrolü oluştur"  
Set timer1 = Controls.Add("VB.Timer", "timer1")  
timer1.Interval = 1  
timer1.Enabled = True  
'Shape kontrolü oluştur  
Set shape1 = Controls.Add("VB.Shape", "shape1")  
ShapeDegistir  
shape1.Visible = True  
End Sub
```

Form_Load olayında ShapeDegistir adında bir yordam vardır. Bu yordamda shape nesnesine ait özellikler rastgele değerlere göre ayarlanacaktır. Form_Load() kodunun başındaki Randomize, her program başlangıcında elde edilen rasgele sayının değişik olmasını sağlar.

```
Private Sub ShapeDegistir()  
shape1.BackStyle = Rastgele(1)  
shape1.BorderStyle = Rastgele(6)
```

```
shape1.BorderColor = RGB(Rastgele(256), Rastgele(256), Rastgele(256))
shape1.BorderWidth = 2
shape1.DrawMode = Rastgele(15)
shape1.FillColor = shape1.BorderColor
shape1.FillStyle = Rastgele(7)
shape1.Shape = Rastgele(5)
End Sub
```

Burada Rastgele rumuzlu fonksiyon rastgele sayı üretimi için kullanılmıştır.

```
Private Function Rastgele(ByVal AzamiDeger As Long) As Long
Rasgele = CLng(Rnd * AzamiDeger)
End Function
```

Harekete renk katacak işler ShapeDegistir yordamında yapılmasına rağmen onu sahneye çağırarak olan, daha önce oluşturulan timer1 bileşenidir.

```
Private Sub timer1_Timer()
Static XHareket As Long
Static YHareket As Long
```

```
'Başlangıç hareket değerleri
If XHareket = 0 Then XHareket = 5
If YHareket = 0 Then YHareket = 5
```

```
'Shape yatay olarak hareket etsin
shape1.Left = shape1.Left + XHareket
If shape1.Left < 1 Then
shape1.Left = 1
XHareket = Rasgele(5) + 1
ShapeDegistir
ElseIf shape1.Left + shape1.Width > Form1.ScaleWidth Then
shape1.Left = Form1.ScaleWidth - shape1.Width - 1
XHareket = Rasgele(5) - 5
ShapeDegistir
End If
```

```
'Shape dikey olarak hareket etsin
shape1.Top = shape1.Top + YHareket
If shape1.Top < 1 Then
shape1.Top = 1
YHareket = Rasgele(5) + 1
ShapeDegistir
ElseIf shape1.Top + shape1.Height > Form1.ScaleHeight Then
shape1.Top = Form1.ScaleHeight - shape1.Height - 1
YHareket = Rasgele(5) - 5
ShapeDegistir
```

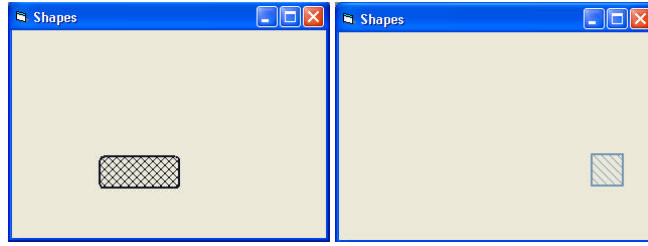


```
End If  
End Sub
```

Program kodunun sonuna geldik. Farenin bir tuşuna basıldığı zaman çıkalım.

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)  
End  
End Sub
```

Artık program çalıştırabilir. Gezindikçe şekil değiştiren, şekil değiştirdikçe renk değiştiren bir animasyon ortaya çıktı.

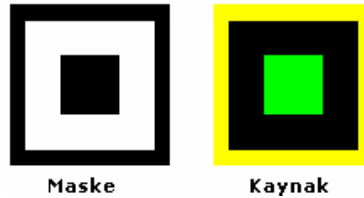


Şekil 3.26: Programın çalışması

ÖRNEK 3.10: Sprite resminin gezinmesi.

Şeffaf resimlerin nasıl çizileceği daha önce BitBlt fonksiyonunu kullanarak görülmüştü. Hatta onlara hareket verildi. Bu sefer de alışılan yolun tersine giderek bunu PaintPicture metodu ile yaparak arkasına bir manzara resmi eklenmesi istenirse yapılacaklar aşağıda anlatılmıştır. Bilindiği gibi, önce eskisi anlatılıp yenisi ve iyisi daha sonra anlatılıyordu.

Bir kere daha hatırlatmak gerekirse şeffaf resimler çizmek için öncelikle kaynak resimdeki göz alıcı renklerin negatifi alan maske resimlerinin çizilmesi gerekir.



Şekil 3.27: Kaynak ve maske resmi

Şekilde sarı ve yeşil olan renklerin kalıcı olması ve aralarından arkadaki fon resminin görünmesi isteniyor. Nerede renkli kalacak bir bölge varsa onun karşılığı maske resimlerinde orası siyaha boyanır. Şeffaflaşması yani kaybolması istenilen alan ise beyaz yapılıdır. Önce maske resmi vbSrcAnd işlemi ile arka plan üzerine kopyalanır. Arkasından kaynak resim vbSrcPaint ile aynı yere yapıştırılır.

Öncelikle aynı ölçüde olan maske ve kaynak isimli iki resim çizilerek çalışma klasörüne kaydedilir. Arka plan olarak Çanakkale Boğazı'nın fotoğrafı seçildi. Buna benzer bir resim de aynı klasöre kopyalanır

Kod listesi aşağıdadır.

```
Option Explicit
Dim WithEvents timer1 As Timer
Dim ImageK As Image
Dim Maske As Image
Private Sub Form_Load()
Randomize
Form1.ScaleMode = 3
Form1.AutoRedraw = True
Form1.Caption = "Gezinen Şeffaf Resim"
Form1.Picture = LoadPicture(App.Path & "\arkaplan.bmp")
Form1.Show
DoEvents

'Zamanlayıcıyı oluştur
Set timer1 = Controls.Add("VB.Timer", "timer1")
timer1.Interval = 10
timer1.Enabled = True

'Image Kontrolü oluştur
Set ImageK = Controls.Add("VB.Image", "ImageK")
ImageK.Picture = LoadPicture(App.Path & "\kaynak.bmp")
ImageK.Left = Rasgele(Form1.ScaleWidth - ImageK.Width - 1)
ImageK.Top = Rasgele(Form1.ScaleHeight - ImageK.Height - 1)
ImageK.Visible = False

'Maske için image kontrolü oluştur.
Set Maske = Controls.Add("VB.Image", "Maske")
Maske.Picture = LoadPicture(App.Path & "\maske.bmp")
Maske.Visible = False
End Sub

Private Sub timer1_Timer()
Static XHareketi As Long
Static YHareketi As Long

'Hareket başlangıç değerleri
If XHareketi = 0 Then XHareketi = 5
If YHareketi = 0 Then YHareketi = 5
```

```

'Yatay hareket
ImageK.Left = ImageK.Left + XHareketi
If ImageK.Left < 1 Then
ImageK.Left = 1
XHareketi = Rasgele(5) + 1
ElseIf ImageK.Left + ImageK.Width > Form1.ScaleWidth Then
ImageK.Left = Form1.ScaleWidth - ImageK.Width - 1
XHareketi = Rasgele(5) - 5
End If

'Dikey olarak gezinsin
ImageK.Top = ImageK.Top + YHareketi

If ImageK.Top < 1 Then
ImageK.Top = 1
YHareketi = Rasgele(5) + 1
ElseIf ImageK.Top + ImageK.Height > Form1.ScaleHeight Then
ImageK.Top = Form1.ScaleHeight - ImageK.Height - 1
YHareketi = Rasgele(5) - 5
End If

'Arkaplanı bir daha çiz
Form1.Picture = LoadPicture(App.Path & "\arkaplan.bmp")

'Maskeleme resmini çiz
Form1.PaintPicture Maske.Picture, ImageK.Left, ImageK.Top, _
ImageK.Width, ImageK.Height, 0, 0, _
ImageK.Width, ImageK.Height, vbSrcAnd

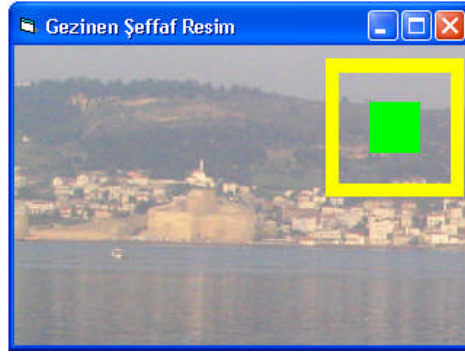
'Kaynak resmini çiz
Form1.PaintPicture ImageK.Picture, ImageK.Left, ImageK.Top, _
ImageK.Width, ImageK.Height, 0, 0, _
ImageK.Width, ImageK.Height, vbSrcPaint
End Sub

Private Function Rasgele(ByVal AzamiDeger As Long) As Long
Rasgele = CLng(Rnd * AzamiDeger)
End Function

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
End
End Sub

Program çalıştırılır.

```



Şekil 3.28: Programın çalışması

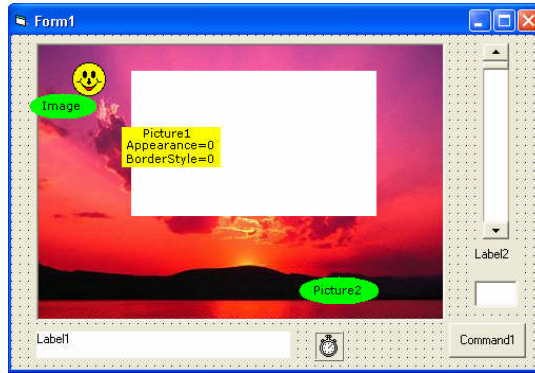
3.8. Oyun

Çalışmaları biraz daha etkili hale getirmek için basit iki oyun üzerinde durulacaktır.

ÖRNEK 3.11: Gülen adamı yakalama

Form tasarımı şekildeki gibi yapalır. Image kontrol içine yerleştirilen gülen adam,

Microsoft Visual Studio\Common\Graphics\Icons\Misc klasöründe FACE05 ikonudur. İstenirse başka bir resim seçebilir.



Şekil 3.29: Form tasarımı

Kod listesi

```
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 hWnd As Long, _  
ByVal hRgn As Long, ByVal bRedraw As Boolean) As Long
```

```
Dim cx, cy As Integer  
Dim XArt, YArt As Single  
Dim Katsayi As Integer
```

```

Dim Hasilat As Long
Dim deneme As Long
Private Sub Form_Load()
Picture1.ScaleMode = 3
Picture2.ScaleMode = 3
Form1.Show
hRgn = CreateEllipticRgn(0, 0, 400, 280)
Hurda = SetWindowRgn(Picture1.hWnd, hRgn, True)
hRgn = CreateEllipticRgn(20, 20, 230, 120)
Hurda = SetWindowRgn(Picture2.hWnd, hRgn, True)
cx = 400 / 2: cy = 280 / 2
Picture2.Top = cy - 75: Picture2.Left = cx - 125
Picture2.PaintPicture Picture1.Picture, 0, 0, 240, 130, _
    cx - 125, cy - 75, 240, 130
Picture2.Visible = True
Katsayi = 15
Label1.Caption = "BAŞLA"
Command1.Caption = "BAŞLA"
Picture1.MousePointer = 5
Label2.Caption = "Yavaşlık Değeri"
Text1.Text = VScroll1.Value
End Sub

```

```

Private Sub Command1_Click()
Hasilat = 0
deneme = 0
Ava_Basla
Form1.BackColor = QBColor(14)
Label1.BackColor = QBColor(3)
Timer1.Interval = VScroll1.Value
Timer1.Enabled = True
End Sub

```

```

Private Sub Image1_Click()
Beep
Hasilat = Hasilat + 1
Label1.Caption = " Sayı: " & Hasilat
Ava_Basla
End Sub

```

```

Sub Ava_Basla()
If deneme >= 10 Then
Timer1.Enabled = False
Form1.BackColor = QBColor(15)
Label1.BackColor = QBColor(15)
Label1.Caption = " Sayı: " & Hasilat

```

```
Beep  
End If
```

```
Image1.Top = cy: Image1.Left = cx  
XArt = (Rnd - 0.5) * Katsayi  
YArt = (Rnd - 0.5) * Katsayi  
deneme = deneme + 1  
End Sub
```

```
Private Sub Timer1_Timer()  
Image1.Top = Image1.Top + YArt  
Image1.Left = Image1.Left + XArt  
If Image1.Top > 280 Or Image1.Top < -32 Or Image1.Left > 400 Or Image1.Left < -  
32 Then  
Ava_Basla  
End If  
End Sub
```

```
Private Sub VScroll1_Change()  
Timer1.Interval = VScroll1.Value  
Text1.Text = VScroll1.Value  
End Sub
```

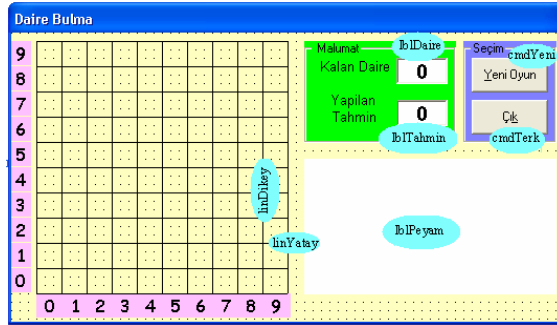
Program çalıştırılır. Başla düğmesi tıklandığında gülen adam şekli ekranda gezinecektir. Fareyle üzerine tıklanırsa sayı alınır. 10 seferlik deneme hakkı var. Yavaşlık ayarı yapılabilir. Görüldüğü gibi gülen adam hızını değiştirmektedir.



Şekil 3.30: Programın çalışması

ÖRNEK 3.12: Izgara Oyunu

Örnek 1.1’de çizgi bileşeni ile bir ızgara oluşturulduğu hatırlanacaktır. Şimdi bu ızgaranın mayın tarlasına benzer bir tarzda nasıl kullanılabileceği gösterilecektir. O projeyi tekrar açarak form Şekil 3.19’da görüldüğü gibi değiştirilir.



Şekil 3.31: Form tasarımı

Bu oyunun amacı, 10x10 karelik bir ızgara rastgele gizlenen dört daireyi bulmaktır. Herhangi bir kareye tıklayıp da bulamazsanız size kaç kare uzakta olduğun bildirilecektir. Bu sorunu çözmek için kullanılacak yöntem pisagor teoremidir.

Program bulunmadık kaç daire kaldığını ve o ana kadar kaç tahminde bulunduğunu da bildirmektedir.

Şu ana kadar yapılanlardan farklı olarak program, yordam yordam açıklanmak istenirse önce genel değişkenler toplu olarak verilir.

```
Option Explicit
Dim OyunBitti As Boolean
Dim DaireYer(4, 2) As Integer
Dim DenemeSay As Integer
Dim Kalan As Integer
```

Formun Load olayında program hakkında kısa bir bilgi verilmektedir. Ayrıca pencere ekranı ortalayacak şekilde ayarlanmaktadır.

```
Private Sub Form_Load()
Randomize
lblPeyam.Caption = "10 x 10 ızgaralık alanda saklanmış dört tane daire var." +
vbNewLine + "Her daireyi en az tahminle bulmaya çalış"
```

```
lblPeyam.Caption = lblPeyam.Caption + vbCrLf + vbCrLf + ">>Yeni oyun için
düğmeye tıkla <<"
Left = 0.5 * (Screen.Width - Width)
Top = 0.5 * (Screen.Height - Height)
End Sub
```

Form üzerinde “Yeni Oyun” başlıklı düğmeye tıklayınca yeni bir oyun başlatılmaktadır.

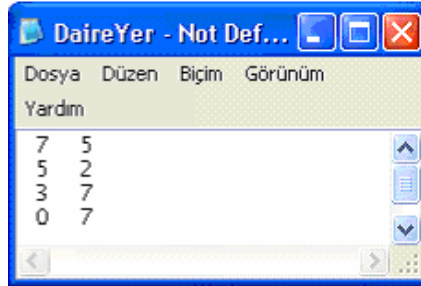
```
Private Sub cmdYeni_Click()  
Dim Dongu, IcDongu As Integer
```

```
'Yeni bir oyuna başla  
OyunBitti = False  
lblPeyam.Caption = ""  
cmdYeni.Enabled = False  
cmdTerk.Caption = "&Dur"  
Kalan = 4  
DenemeSay = 0  
lblDaire.Caption = "4"  
lblTahmin.Caption = "10"  
Form1.Cls
```

```
For Dongu = 1 To 4  
    For IcDongu = 1 To 2  
        DaireYer(Dongu, IcDongu) = Int(10 * Rnd)  
    Next IcDongu  
Next Dongu
```

```
lblPeyam.Caption = vbCrLf + "Dört Daire Saklı." + vbCrLf + "Gizli olduğunu  
düşündüğünüz yer için tıklayın."  
lblPeyam.Refresh  
End Sub
```

Öncelikle bir döngü ile tüm ızgarayı temsil eden iki boyutlu “Izgara” dizisi, sıfırla doldurulur. Sonra başka bir döngü ile dairelerin saklı olduğu yerler rasgele sayılarla belirlenerek “DaireYer” isimli diziye atılır. Bunlar bir dosyaya yazdırılıp bakılırsa başlangıç değerleri görülebilir.



Şekil 3.32: Dizi değerleri

Demek ki daireler (7,5)-(5,2)-(3,7)-(0,7) karelerine yerleşecek. Ama bu sayılar her seferinde farklı olur ve bunu Randomize ifadesi ve Rnd fonksiyonu sağlar.

Duruma göre hemen oyundan çıkmak istenebilir. İşte çıkış düğmesine basıldığında olacaklar:

```
Private Sub cmdTerk_Click()  
  
'Oyundan ya çık yada oyunu durdur  
If cmdTerk.Caption = "Çı&k" Then  
Unload Form1  
Else  
cmdTerk.Caption = "Çı&k"  
cmdYeni.Enabled = True  
If Not (OyunBitti) Then lblPeyam.Caption = vbCrLf + "Oyun durdu"  
cmdYeni.SetFocus  
End If  
End Sub
```

Form etkin olduğunda program, düğme üzerinde odaklansın.

```
Private Sub Form_Activate()  
cmdYeni.SetFocus  
End Sub
```

En önemli bölüm fare ile tıklayarak yer bulmadır..

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, _  
X As Single, Y As Single)
```

```
Dim I As Integer  
Dim M As Integer, N As Integer  
Dim D As Single  
Dim Bulunan As Boolean
```

```
If cmdTerk.Caption = "E&xit" Then Exit Sub  
If X < linDikey(0).X1 Or X > linDikey(10).X1 Then Exit Sub  
If Y < linYatay(10).Y1 Or Y > linYatay(0).Y1 Then Exit Sub
```

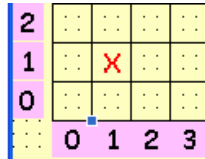
```
'sütun Bul  
For M = 0 To 9  
If X >= linDikey(M).X1 And X <= linDikey(M + 1).X1 Then Exit For  
Next M
```

```
'Satır Bul  
For N = 0 To 9  
If Y <= linYatay(N).Y1 And Y >= linYatay(N + 1).Y1 Then Exit For  
Next N
```

```

DenemeSay = DenemeSay + 1
lblTahmin.Caption = Format(DenemeSay, "0")
lblTahmin.Refresh
lblPeyam.Caption = vbCrLf
Bulunan = False
For I = 1 To 4
If DaireYer(I, 1) <> -1 Then
If DaireYer(I, 1) <> M Or DaireYer(I, 2) <> N Then
D = Sqr((DaireYer(I, 1) - M) ^ 2 + (DaireYer(I, 2) - N) ^ 2)
lblPeyam.Caption = lblPeyam.Caption + Str(I) + "nci daireye " + Format(D, "0.0") + "
kare uzaktasınız" + vbCrLf
Else
Bulunan = True
DaireYer(I, 1) = -1
lblPeyam.Caption = lblPeyam.Caption + Str(I) + "nci daireyi buldunuz" + vbCrLf
Kalan = Kalan - 1
lblDaire.Caption = Format(Kalan, "0")
lblDaire.Refresh
End If
End If
Next I
If Bulunan Then
Form1.FillStyle = 0
Form1.FillColor = vbRed
Form1.Circle (0.5 * (linDikey(M).X2 + linDikey(M + 1).X2), 0.5 * (linYatay(N).Y2 +
linYatay(N + 1).Y1)), 0.4 * (linDikey(M + 1).X2 - linDikey(M).X2), QBColor(12)
Else
Form1.FillStyle = 7
Form1.FillColor = vbBlue
Form1.Line (linDikey(M).X2, linYatay(N).Y2)-(linDikey(M + 1).X2, linYatay(N +
1).Y1), QBColor(9), B
End If
If Kalan = 0 Then OyunBitti = True
lblPeyam = lblPeyam + vbCrLf + "Dört daire için" + vbCrLf + Str(DenemeSay) + "
defa tahmin yaptın."
Call cmdTerk_Click
End If
End Sub
Fare ile tıklanılan yer ızgaranın içinde olup olmadığı döngü içinde yoklanıyor. Şekilde
görülen kareye tıklандığı düşünülürse;

```



Şekil 3.33: Tıklama yeri

Bu alanı, (linYatay(1), linYatay(2), linDikey(1), linDikey(2) çizgileri sınırlar.

```
For M = 0 To 9
If X >= linDikey(M).X1 And X <= linDikey(M + 1).X1 Then Exit For
Next M
```

Döngüsünün sonucu bunun birinci ve ikinci çizgi arasında olduğu bulunur (örneğin 720-1080 pikselleri arasında) ve M=1 olur.

```
For N = 0 To 9
If Y <= linYatay(N).Y1 And Y >= linYatay(N + 1).Y1 Then Exit For
Next N
```

döngüsünde yine N=1 bulunur. Bu sayılar

If DaireYer(I, 1) <> M Or DaireYer(I, 2) <> N Then.....

Şart cümlesinde DaireYer dizisini elemanları ile karşılaştırılıyor. Örneğin DaireYer(2,1) ya da DaireYer(2,2) de yer alan sayılar M ve N sayılarını tutuyorsa daire bulunmuştur ve oraya kırmızı bir daire çizilecektir.

Bulamadığında kaç kare uzakta olduğunu gösteren değer nedir?

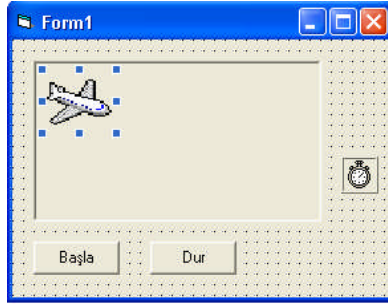


Şekil 3.34: Programın çalışması

UYGULAMA FAALİYETİ

Aşağıdaki sorulara ilişkin uygulama faaliyetini yapınız.

- Örnek 3.3'te yapılan uygulamaya, araba penceresinin solundan gözden kaybolduğunda tekrar başa gelmesini sağlayan kodu ekleyiniz.
- Örnek 3.3'te trafik lambalarının yanma zamanını yatay bir kaydırma çubuğu ile belirleyin.
- PictureBox içinde bir uçağın uçmasını canlandırın. Uçak resim kutusunun sınırlarına çarptığında geri dönsün.



Şekil 3.35: Form tasarımı

Uçak sembolü için "C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Industry" klasöründe plane.ico dosyasının kullanabilirsiniz.

- Örnek 3.14'te görülen oyunda oyun bittiğinde kullanıcıların adlarını ve skorları, ikinci bir form üzerindeki liste kutusuna yazdırın. Buradan da rasgele erişimli yada kullanıcı tanımlı bir dosyaya kaydedin. Program tekrar açıldığında puan sırasına göre kullanıcılar sıralansın.

İşlem Basamakları	Öneriler
<ul style="list-style-type: none">➤ Şekilleri form üzerine yerleştiriniz.➤ Hangi olayları kullanacağınıza karar veriniz.➤ Döngü tipine iyi karar verin➤ Yazdığınız programı çalıştırın.➤ Programda hata var ise bunları gideriniz.	<ul style="list-style-type: none">➤ Nesne özelliklerine uygun değerler atayınız.➤ Hangi değişken tipini kullanacağınıza dikkat ediniz.➤ Değişken artım oranlarını iyi ayarlayınız.➤ Program satırlarının düzenli olmasına özen gösteriniz.➤ Programı çalıştırmadan önce muhakkak kaydediniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları cevaplayarak bu faaliyette kazandığınız bilgileri ölçünüz.

OBJEKTİF TEST (ÖLÇME SORULARI)

1.Kendine ait olayları olan bileşenler formun declarations bölümüne nasıl hangi deyimle tanıtır?

- A) Dim
- B) WithEvents
- C) Set
- D) Load

2.Şeffaf resimler oluşturmak istediğimizde BitBlt fonksiyonunun yada PaintPicture metodunun hangi parametreleri beraber kullanılır?

- A) vbScrAnd-vbScrPaint
- B) vbNotSrcErase-vbMergePaint
- C) vbNotSrcErase- vbSrcAnd
- D) vbSrcCopy- vbSrcErase

3.Zaman ayarlamak için hangi fonksiyon kullanılır?

- A) GettickNumber
- B) GetTickcount
- C) GetTickTime
- D) GetTickDate

4.Resimlerin büyütülmesi ya da küçültülmesi ile ilgilenen fonksiyon aşağıdakilerden hangisidir?

- A) BitBlt
- B) GetBitmapBits
- C) VarPtr
- D) StretchBlt

5.Aşağıdakilerden resimlerin canlandırma anında titremesini önleme metotlarından birisidir?

- A) DrawMode-Refresh
- B) ScaleMode-Refresh
- C) AutoRedraw-Refresh
- D) AutoRedraw-ScaleMode

DEĞERLENDİRME

Cevaplarınızı cevap anahtarı ile karşılaştırınız. Doğru cevap sayınızı belirleyerek kendinizi değerlendiriniz. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt yaşadığınız sorularla ilgili konuları faaliyete geri dönerek tekrar inceleyiniz.

MODÜL DEĞERLENDİRME

UYGULAMALI TEST (YETERLİK ÖLÇME)

Modülde yaptığınız uygulamaları tekrar yapınız. Yaptığınız bu uygulamaları aşağıdaki tabloya göre değerlendiriniz.

AÇIKLAMA: Aşağıda listelenen kriterleri uyguladıysanız Evet sütununa, uygulamadıysanız Hayır sütununa X işareti yazınız.		
DEĞERLENDİRME ÖLÇÜTLERİ	Evet	Hayır
Temel çizim elemanlarını doğru şekilde kullandınız mı?		
Grafik metotlarını (Line, Pset) doğru olarak kullandınız mı?		
PictureBox olaylarını doğru olarak kullandınız mı?		
Form üzerine fare ile çizim yaptınız mı?		
Farklı çizelgelerle çalıştınız mı?		
Projenize farklı amaçlar için API eklediniz mi?		
Projenizde maskeleyme işlemi yaptınız mı?		
API tanımlamalarını yaptınız mı?		
Projenizde herhangi bir animasyon kullandınız mı?		

DEĞERLENDİRME

Hayır cevaplarınız var ise ilgili uygulama faaliyetini tekrar ediniz. Cevaplarınızın tümü evet ise bir sonraki modüle geçebilirsiniz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ - 1 (UYGULAMA FAALİYETİ)

1:

```
Const pi = 3.14
```

```
Private Sub Form_Load()  
    Form1.ScaleMode = 3  
    Form1.AutoRedraw = True  
    Timer1.Interval = 100  
End Sub
```

```
Private Sub Timer1_Timer()  
    Celebi_Mehmet = Hour(Time)  
    Dakika = Minute(Time)  
    Saniye = Second(Time)  
    Celebi_Mehmet = (Celebi_Mehmet / 12) * 2 * pi  
    Dakika = (Dakika / 60) * 2 * pi  
    Saniye = (Saniye / 60) * 2 * pi  
    Cls  
    Form1.DrawWidth = 4  
    Line (200, 200)-(200 + (100 * Cos(Celebi_Mehmet - 1.57)), _  
200 + (100 * Sin(Celebi_Mehmet - 1.57))), &H0  
    Form1.DrawWidth = 2  
    Line (200, 200)-(200 + (120 * Cos(Dakika - 1.57)), _  
200 + (120 * Sin(Dakika - 1.57))), &H0  
    Form1.DrawWidth = 1  
    Line (200, 200)-(200 + (150 * Cos(Saniye - 1.57)), _  
200 + (150 * Sin(Saniye - 1.57))), &HFF  
End Sub
```

2:

```
Option Explicit
```

```
Dim CentreX As Integer, CentreY As Integer  
Dim StartX As Integer, StartY As Integer  
Dim Gul As Boolean
```

```
Private Sub hsbColour_Change(Index As Integer)  
    lblInkPot.BackColor = RGB(CInt(lblRenk(0).Caption), CInt(lblRenk(1).Caption),  
CInt(lblRenk(2).Caption))  
    lblRenk(Index).Caption = hsbColour(Index).Value
```

```

    picCanvas.ForeColor = lblInkPot.BackColor
End Sub

Private Sub hsbColour_Scroll(Index As Integer)
    lblInkPot.BackColor = RGB(CInt(lblRenk(0).Caption), CInt(lblRenk(1).Caption),
CInt(lblRenk(2).Caption))
    lblRenk(Index).Caption = hsbColour(Index).Value
    picCanvas.ForeColor = lblInkPot.BackColor
End Sub

Private Sub hsbKalinlik_Change()
    lblThickness.Caption = hsbKalinlik.Value
    picCanvas.DrawWidth = hsbKalinlik.Value
End Sub

Private Sub picCanvas_MouseDown(Button As Integer, Shift As Integer, X As Single,
Y As Single)
    Dim DrawingStyle As Integer

    DrawingStyle = GetStyle()

    If Button = vbLeftButton Then
        Select Case DrawingStyle
            Case 0 ' Free Hand
                picCanvas.PSet (X, Y)
            Case 1 ' Line

                StartX = X
                StartY = Y

            Case 2 ' Circle
                CentreX = X
                CentreY = Y
            End Select
        Else
            picCanvas.Cls
        End If

    End Sub

Private Sub picCanvas_MouseMove(Button As Integer, Shift As Integer, X As Single,
Y As Single)
    Static OldX As Integer, OldY As Integer

    Dim Radius As Double
    Dim DrawingStyle As Integer
    DrawingStyle = GetStyle()

```



```

If Button = vbLeftButton Then
  Select Case DrawingStyle
    Case 0 ' Serbest El
      picCanvas.Line -(X, Y)

    Case 1 ' Line

      picCanvas.DrawMode = vbInvert

      If Gul = True Then
        picCanvas.Line (StartX, StartY)-(OldX, OldY)
      End If
      picCanvas.Line (StartX, StartY)-(X, Y)

    Gul = True

      OldX = X
      OldY = Y

    Case 2 ' Circle

      picCanvas.DrawMode = vbInvert

      If Gul = True Then

        Radius = Sqr((OldX - CentreX) ^ 2 + (OldY - CentreY) ^ 2)
        picCanvas.Circle (CentreX, CentreY), Radius
      End If

      Radius = Sqr((X - CentreX) ^ 2 + (Y - CentreY) ^ 2)
      picCanvas.Circle (CentreX, CentreY), Radius
      Gul = True

      OldX = X
      OldY = Y
    End Select
  End If
End Sub

Private Sub picCanvas_MouseUp(Button As Integer, Shift As Integer, X As Single, Y
As Single)

  Dim Radius As Double
  Dim DrawingStyle As Integer
  DrawingStyle = GetStyle()

  If Button = vbLeftButton Then

```

```

Select Case DrawingStyle

Case 1 ' line
    picCanvas.DrawMode = vbCopyPen
    picCanvas.Line (StartX, StartY)-(X, Y)

Case 2 ' Circle
    picCanvas.DrawMode = vbCopyPen

    Radius = Sqr((X - CentreX) ^ 2 + (Y - CentreY) ^ 2)
    picCanvas.Circle (CentreX, CentreY), Radius
End Select
End If
Gul = False
End Sub

Private Function GetStyle() As Integer
    Dim Counter As Integer

    For Counter = 0 To 2
        If optStyle(Counter).Value = True Then
            GetStyle = Counter
        End If
    Next Counter
End Function

```

OBJEKTİF TEST

1	A
2	D
3	C
4	B
5	A

ÖĞRENME FAALİYETİ - 2 (UYGULAMA FAALİYETİ)

1.

```

Private Declare Function Rectangle Lib "gdi32" (ByVal _
hdc As Long, ByVal X1 As Long, ByVal Y1 As Long, ByVal X2 As Long, _
ByVal Y2 As Long) As Long
Private Sub Command1_Click()
    retval = Rectangle(Form1.hdc, 10, 10, 160, 70)
End Sub

```

2.

```
Private Declare Function FloodFill Lib "gdi32" (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, ByVal crColor As Long) As Long
Private Sub Form_Load()
Dim cs As Long
Dim j As Long
cs = QBColor(0)
Form1.Show
Form1.FillStyle = 0
Scale (0, 0)-(15, 100)
Line (0, 20)-(15, 20), cs
Line (0, 20)-(7.5, 80), cs
For i = 1 To 15
Form1.Line (i, 20)-(7.5, 80), cs
Form1.FillColor = QBColor(i)
xp = ScaleX(i, 0, 3) - 5
yp = ScaleY(20, 0, 3) + 1
FloodFill hdc, xp, yp, cs
Next i
For j = 1 To 45000
Next j
For i = 15 To 1 Step -1
Form1.Line (i, 20)-(7.5, 80), cs
Form1.FillColor = QBColor(i - 1)
xp = ScaleX(i, 0, 3) - 5
yp = ScaleY(20, 0, 3) + 1
FloodFill hdc, xp, yp, cs
Next i

End Sub
```

OBJEKTIF TEST

1	C
2	A
3	C
4	B
5	D
6	A
7	B
8	D
9	D
10	A

ÖĞRENME FAALİYETİ-3 CEVAP ANAHTARI

1	B
2	A
3	B
4	D
5	C

KAYNAKÇA

- BALENA Francesco, **Programming Microsoft Visual Basic 6.0**, Microsoft Pres, 1999.
- GREG Perry, **Sams Teach Yourself Visual Basic 6 in 21 Days**, Macmillan Computer, 1998.
- HARBOUR Jonathan S., **VB Game Programming with DirectX**, by Premier Press, Inc., 2002.
- HOLZNER Steven ,The Coriolis Group, **Visual Basic 6 Black Book**, 1998.
- NORTON Peter, **Peter Norton's Guide to Visual Basic 6**, Macmillan Computer Publishing, 1998.
- ÖĞÜTMEN Nigar, **Grafik Formatları ve 3. Boyut**, Beta Yayınevi, İstanbul 2000.
- PALA Zeydin , KARAGÜLLE İhsan, **Visual Basic 6.0 Pro**, Türkmen Yayınevi, İstanbul, 2002.
- SCHNEIDER David I., **Computer Programming Concepts and Visual Basic**, Pearson Custom Publishing, 1999.
- **Visual Basic**, CQ Yayınevi, 1998.
- <http://www.activevb.de>
- <http://docvb.free.fr>
- <http://www.ex-designz.net>
- <http://www.freevbcode.com/>
- <http://www.garybeene.com>
- <http://goforit.unk.edu>
- <http://www.kidwaresoftware.com>
- <http://pages.cpsc.ucalgary.ca>
- <http://www.tutorialized.com>
- <http://www.vbarchiv.net>
- <http://www.vbexplorer.com>
- <http://www.vb-fun.de>
- <http://vb-helper.com/>