

**T.C.
MİLLÎ EĞİTİM BAKANLIĞI**

BİLİŞİM TEKNOLOJİLERİ

GELİŞMİŞ İNTERNET UYGULAMALARINDA TASARIM ARAÇLARI

Ankara, 2013

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- **PARA İLE SATILMAZ.**

İÇİNDEKİLER

AÇIKLAMALAR	iii
GİRİŞ	2
ÖĞRENME FAALİYETİ-1	4
1. YERLEŞİM.....	4
1.1. StackPanel.....	4
1.2. Canvas.....	6
1.3. Grid	6
1.4. Border	7
UYGULAMA FAALİYETİ	9
ÖLÇME VE DEĞERLENDİRME	12
ÖĞRENME FAALİYETİ-2.....	13
2. FIRÇALAR	13
2.1. SolidColorBrush.....	13
2.2. LinearGradientBrush.....	14
2.3. RadialGradientBrush.....	16
2.4. ImageBrush	17
2.5. VideoBrush	18
UYGULAMA FAALİYETİ	20
ÖLÇME VE DEĞERLENDİRME	23
3. GÖRSEL ÖZELLİKLER.....	24
3.1. Ölçü ve Yerleşim Özellikleri	24
3.2. Opacity	25
3.3. Cursor.....	25
3.4. Stroke	26
UYGULAMA FAALİYETİ	27
ÖLÇME VE DEĞERLENDİRME	29
ÖĞRENME FAALİYETİ-4.....	30
4. ŞEKİLLER.....	30
4.1. Elips (Ellipse).....	30
4.2. Dikdörtgen (Rectangle).....	31
4.3. Çizgi (Line).....	31
4.4. Yol (Path).....	33
4.5. Çoklu Geometrik Nesnelere (GeometryGroup).....	34
UYGULAMA FAALİYETİ	36
ÖLÇME DEĞERLENDİRME.....	39
ÖĞRENME FAALİYETİ-5	40
5. KONTROLLER	40
5.1. Resim (Image).....	40
5.2. Kabartma (Glyphs).....	41
5.3. Metin Bloğu (TextBlock).....	42
UYGULAMA FAALİYETİ	43
ÖLÇME VE DEĞERLENDİRME	45
ÖĞRENME FAALİYETİ-6	46
6. DÖNÜŞÜM VE ANİMASYON	46
6.1. Döndürme (RotateTransform).....	46
6.2. Ölçeklendirme (ScaleTransform).....	47

6.3. Dönüşüm (TranslateTransform).....	48
6.4. Çarpıtma (SkewTransform)	49
6.5. Matris (MatrixTransform).....	49
6.6. Senaryo (Storyboard)	50
6.7. Animasyon Çeşitleri.....	51
6.7.1. Double Animation	51
6.7.2. ColorAnimation	53
6.7.3. PointAnimation.....	54
6.8. Anahtar Çerçevesel (Key Frame)	55
6.8.1. <DoubleAnimationUsingKeyFrames> Etiketi	56
6.8.2. <ColorAnimationUsingKeyFrames> Etiketi	57
6.8.3. <PointAnimationUsingKeyFrames> Etiketi.....	58
6.9. Animasyonların Kontrol Olayları İçinden Tetiklenmesi.....	58
UYGULAMA FAALİYETİ	60
ÖLÇME VE DEĞERLENDİRME	62
MODÜL DEĞERLENDİRME	64
CEVAP ANAHTARLARI	65
KAYNAKÇA	68

AÇIKLAMALAR

ALAN	Bilişim Teknolojileri
DAL / MESLEK	Veri Tabanı Programcısı
MODÜLÜN ADI	Gelişmiş İnternet Uygulamalarında Tasarım Araçları
MODÜLÜN TANIMI	Tasarım özelliklerini düzenlemeyi, fırçaları kullanmayı, görsel özellikleri kullanarak uygulama geliştirmeyi, şekil işlemlerini gerçekleştirmeyi, kontrolleri kullanmayı, dönüşüm ve animasyon işlemlerini yapmak için gerekli bilgilerin kazandırıldığı bir öğrenme materyalidir.
SÜRE	40/32
ÖN KOŞUL	
YETERLİK	XAML tasarım araçlarını kullanmak
MODÜLÜN AMACI	Genel Amaç Bu modül ile gerekli ortam sağlandığında; tasarım araçlarını kullanabileceksiniz. Amaçlar <ol style="list-style-type: none">1. Tasarım özelliklerini düzenleyebileceksiniz.2. Fırça işlemlerini gerçekleştirebileceksiniz.3. Görsel özellikleri kullanarak uygulama geliştirebileceksiniz.4. Şekil işlemlerini gerçekleştirebileceksiniz.5. Kontrolleri kullanabileceksiniz.6. Dönüşüm ve animasyon işlemlerini gerçekleştirebileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Ortam: Atölye, bilişim teknolojisi sınıfı, kütüphane, <i>İnternet</i> , bireysel öğrenme ortamları vb. Donanım: Bilgisayar (PC), projeksiyon, gerekli olan yazılımlar vb.
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

XAML tabanlı *İnternet* uygulamaları geliřtirmede tasarımın rolü çok büyüktür. Bu modülden arayüz tasarımında kullanılan kontroller ve yöntemler hakkında yeterli derecede bilgi verilmiştir.

Bu modülden kontrollerin sayfaya yerleşirilmesi için kullanılan panelleri öğrenebileceksiniz. Ardından şekil ve nesnelerin renklendirilmesinde kullanılan çeşitli fırça nesnelerini ve özellikleri hakkında yeterli bilgiye sahip olacaksınız. Daha sonra nesnelerin ölçülendirilmesi ve sayfa üzerinde konumlandırılması ile ilgili olarak gerekli bilgiler verilmiştir. Ayrıca sayfa üzerine düzgün ya da düzgün olmayan cisimlerin çizimlerini yapma yöntemlerini öğrenebilirsiniz. Çeşitli XAML kontrolleri ile ilgili bazı önemli bilgiler de verilmiştir. Nesnelerin döndürülme yöntemleri hakkında da yeterli derecede bilgi mevcuttur. En önemlisi de uygulamalarda kullanılan nesnelerin çeşitli yöntemlerle hareketlendirilmesi için gerekli bazı konulara değinilmiştir.

Tasarım ve animasyon işlemleri için fevkalade önemli bilgiler içeren bu modülü başarıyla bitirmeniz dileğiyle.

ÖĞRENME FAALİYETİ-1

AMAÇ

Tasarım özelliklerini düzenleyebileceksiniz.

ARAŞTIRMA

- XAML sayfalarında kullanılan panelleri araştırınız.

1. YERLEŞİM

XAML, gelişmiş *Internet* uygulamalarında tasarım kısımlarını geliştireceğimiz, statik ve dinamik kullanıcı arayüzleri oluşturmamızı sağlayan işaretleme dilidir. XML dilinin genişletilmiş hâli gibi düşünülebilir. XAML bir programlama dili olmamakla beraber program kodları xaml uzantılı tasarım dosyalarına bağlı olan cs uzantılı codebehind dosyalarında bulunmaktadır.

XAML dilinde tanımlanan ilk eleman, root eleman olarak adlandırılır. UserControl, bir root elemandır. Root elemanlarının kapsadığı panel elemanları ise Stackpanel, Grid, Canvas ve Border gibi yerleşim panelleridir.

1.1. StackPanel

StackPanel, en basit tanımıyla içine aldığı nesnelere, yatay ve dikey olarak hizalamakla görevlidir. Oldukça kullanışlı olan Stack Panel yardımı ile menu, liste, sayfalama uygulamalarını rahatlıkla yapabiliriz. Stack Panel kontrolünün en önemli özelliği Orientation özelliğidir. Orientation özelliği nesnelere yatay ya da dikey olarak dizilmelerini sağlar.

Orientation özelliği Vertical olarak ayarlanmış bir StackPanel örneği:

```
<StackPanel Name="stackPanel1" Orientation="Vertical">
  <Button Content="Ana Sayfa" Name="button1" />
  <Button Content="Duyurular" Name="button2" />
  <Button Content="İletişim" Name="button3" />
  <Button Content="Hakkımızda" Name="button4" />
  <Button Content="Yardım" Name="button5" />
</StackPanel>
```

Ekran ıktısı:

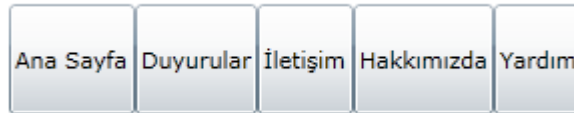


Resim 1.1: Butonların üst üste dizilmesi

Orientation özelliđi Horizontal olarak ayarlandığında ise aşıđıdaki ekran ıktısını göreceđiz.

```
<StackPanel Name="stackPanel1" Orientation="Horizontal" Height="55" Width="357">  
  <Button Content="Ana Sayfa" Name="button1" />  
  <Button Content="Duyurular" Name="button2" />  
  <Button Content="İletişim" Name="button3" />  
  <Button Content="Hakkımızda" Name="button4" />  
  <Button Content="Yardım" Name="button5" />  
</StackPanel>
```

Ekran ıktısı:



Resim 1.2: Butonların yan yana dizilmesi

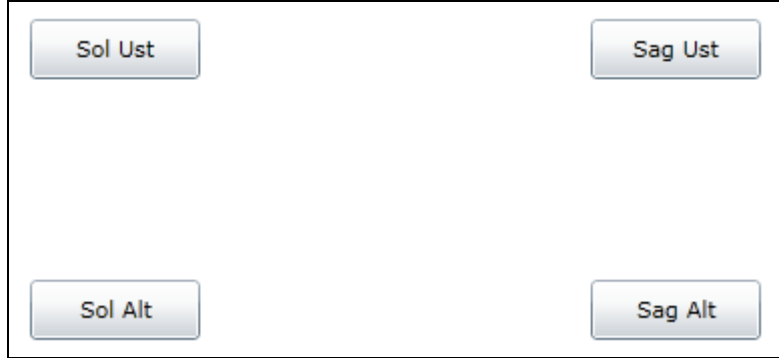
1.2. Canvas

Canvas kontrolü, içinde yer alan nesnelerin sol ve üst konumlarını belirlemek için kullanılır. Canvas paneli içerisine yerleştirdiğimiz kontrolün sol ve üst pozisyonları, tuvalin sol üst köşesi referans alınarak pixel cinsinden belirlenebilir.

Örnek:

```
<Canvas>  
  <Button Content="Sol Ust" Width="85" Height="30" Canvas.Top="20"  
Canvas.Left="20"/>  
  <Button Content="Sag Ust" Width="85" Height="30" Canvas.Top="20"  
Canvas.Left="300"/>  
  <Button Content="Sol Alt" Width="85" Height="30" Canvas.Top="150"  
Canvas.Left="20"/>  
  <Button Content="Sag Alt" Width="85" Height="30" Canvas.Top="150"  
Canvas.Left="300"/>  
</Canvas>
```

Ekran çıktısı:



Resim 1.3: Canvas kontrolü

1.3. Grid

HTML tabloları gibi satır ve sütun tanımlamaları yapılarak hücelere bölünebilen grid kontrolünde her bir satır ve sütun için genişlik ve yükseklik değerleri belirlenebilir.

Grid kontrolünde kullanılan özelliklerden;

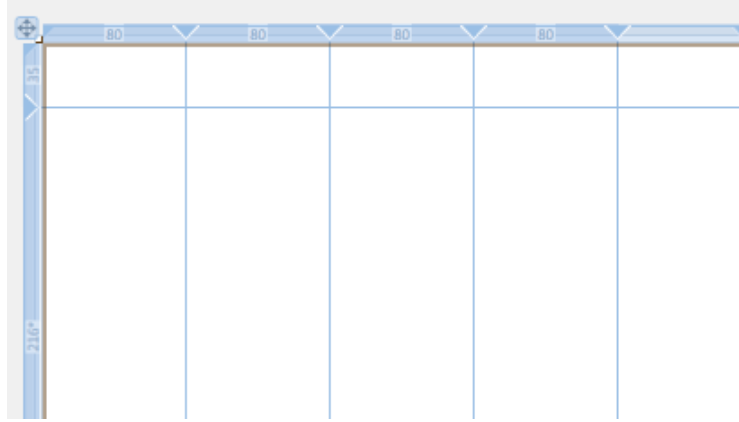
- RowDefinitions özelliği satır oluşturmamızı sağlar.
- ColumnDefinitions özelliği sütun oluşturmamızı sağlar.
- ShowGridLines özelliği tanımlanan satır ve sütunların çizgilerinin görünmesini sağlar.

Burada ölçüler genelde pixel(px) cinsinden verilir. Ama farklı ölçü birimleri kullanarak da bu değerleri atayabiliriz.

Örnek:

```
<Grid Margin="5">  
  <Grid.ColumnDefinitions>  
    <ColumnDefinition Width="80"/>  
    <ColumnDefinition Width="80"/>  
    <ColumnDefinition Width="80"/>  
    <ColumnDefinition Width="80"/>  
  </Grid.ColumnDefinitions>  
  <Grid.RowDefinitions>  
    <RowDefinition Height="35"/>  
    <RowDefinition Height="216*"/>  
  </Grid.RowDefinitions>  
</Grid>
```

Ekran çıktısı: Form alanının hücelere bölündüğünü görebiliriz.



Resim 1.4: Grid kontrolü

1.4. Border

Border kontrolü içine yer alan nesnelere bir çerçeve eklemek için kullanılır. Border kontrolünün en önemli özellikleri Borderbrush, BorderThickness ve CornerRadius özellikleridir.

BorderBrush özelliği, kenar çizgilerinin rengini değiştiren özelliktir. BorderThickness özelliği ise kenar çizgilerinin kalınlık değerlerini değiştirir. CornerRadius özelliği, çerçeve köşelerinin yuvarlaklık değerini değiştirir. Ayrıca Opacity (şeffaflık) ve Visible (görünürlük) özellikleri de mevcuttur.

Örnek: Border kullanımı (Bu örnekteki kodlar yazılmadan önce Penguins.jpg dosyası proje klasörünün ClientBin klasörüne kopyalanmalıdır.)



```
<Canvas>  
  <Border HorizontalAlignment="Center" VerticalAlignment="Center"  
  CornerRadius="10" BorderBrush="Red" BorderThickness="5" Canvas.Left="71"  
  Canvas.Top="58">  
    <Image Height="150" Name="image1" Stretch="Fill" Width="200"  
    Source="Penguins.jpg" />  
  </Border>  
</Canvas>
```

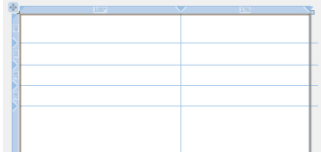

Ekran çıktısı:



Resim 1.5: Border kontrolü

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler
Tasarım özelliklerini düzenlemek için;	
<p>➤ StackPanel kontrolünü kullanarak aşağıdaki form görüntüsünü oluşturunuz.</p> 	<p>➤ Wildlife.wmv video dosyasını ClientBin klasörüne kopyalayınız.</p> <p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Grid> <StackPanel Height="238" Margin="53, 36, 0,0" Name="stackpanel1" Orientation="Vertical" HorizontalAlignment="Left" Width="371" MaxHeight="750" MaxWidth="750"> <MediaElement Height="120" Name="mediaelement1" Width="160" Source="Wildlife.wmv" /> <Button Content="Oynat" Height="23" Name="button2" Width="75" HorizontalAlignment="Center" /> <Button Content="Duraklat" Height="23" Name="button3" Width="75" HorizontalAlignment="Center" /> </StackPanel> </Grid></pre>
<p>➤ Canvas kullanarak aşağıdaki form görüntüsünü oluşturunuz.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Rectangle Canvas.Left="35" Canvas.Top="80" Height="100" Width="100" Stroke="Yellow" StrokeThickness="5" Fill="Red" /> <Rectangle Canvas.Left="60" Canvas.Top="62" Height="100" Width="100" Stroke="Yellow" StrokeThickness="5" Fill="Blue" /> <Rectangle Canvas.Left="95" Canvas.Top="36" Height="100" Width="100" Stroke="Yellow" StrokeThickness="5" Fill="Green" /> </Canvas></pre>
<p>➤ Grid kontrolünü kullanarak aşağıdaki görüntüyü oluşturunuz.</p>	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Grid> <Grid.ColumnDefinitions> <ColumnDefinition Width="100*" /> <ColumnDefinition Width="176" /> <ColumnDefinition Width="*" /> </Grid></pre>

	<pre> </Grid.ColumnDefinitions> <Grid.RowDefinitions> <RowDefinition Height="40" /> <RowDefinition Height="30" /> <RowDefinition Height="28" /> <RowDefinition Height="28" /> <RowDefinition Height="*" /> </Grid.RowDefinitions> </Grid> </pre>
<p>➤ Border kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Tulips.jpg dosyasını Belgelerim klasöründen projenin ClientBin klasörüne kopyalayınız.</p> <p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre> <Border BorderBrush="Black" BorderThickness="5" Height="161" HorizontalAlignment="Left" Margin="46,92,0,0" Name="border1" VerticalAlignment="Top" Width="207" CornerRadius="25"> <Image Height="141" Name="image1" Stretch="Fill" Width="179" Source="Tulips.jpg"/> </Border> </pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
Tasarım özelliklerini düzenlemek için;		
1. Yerleşim panellerini kullandınız mı?		
2. Tuval ayarlarını yaptınız mı?		
3. Izgara düzenlemelerini yaptınız mı?		
4. Kenarlıklarla ilgili ayarlamaları yaptınız mı?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi XAML tabanlı İnternet uygulamalarında tasarım kısımlarını geliştireceğimiz, statik ve dinamik kullanıcı arayüzleri oluşturmamızı sağlayan işaretleme dilidir?
A) HTML B) DHTML C) XAML D) CSS
2. Aşağıdakilerden hangisi kullanıcı ile daha etkileşimli uygulamalar kullanarak ses ve görüntülerle zengin içerikleri uygulamalar yapmamızı sağlayan sistemdir?
A) WPF B) SQL Server C) ASP.NET D) XML
3. I- Nesneleri sadece alt alta ya da sadece yan yana listelemek
II- Nesnelerin yerleşim pozisyonlarını noktasal olarak belirlemek
III- Form alanını satırlar ve sütunlara bölmek
Yukarıdakilerden hangileri Grid kontrolünün görevidir?
A) Yalnız I B) Yalnız II C) Yalnız III D) I ve III
4. Aşağıdakilerden hangisi yatay ve dikey olarak yönlendirilmiş tek satırlık değerler eklemek için kullanılır?
A) Border B) Grid C) Tuval(Canvas) D) Stackpanel
5. Aşağıdakilerden hangisi form içerisinde kontrolleri istenilen x ve y koordinatlarına yerleştirmek için kullanılır.
A) Grid B) Border C) Tuval(Canvas) D) Stackpanel

Aşağıdaki cümleleri dikkatlice okuyarak boş bırakılan yerlere doğru sözcüğü yazınız.

6. Kenarlıklarla ilgili ayarlamalar kontrolü ile yapılır.
7. Satırlarda yükseklik değeri.....özellği kullanılarak belirlenir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-2

AMAÇ

Fırçaları kullanabileceksiniz.

ARAŞTIRMA

- Tek renk (SolidColorBrush) fırçasını araştırınız.
- Doğrusal olarak değişen renk fırçasını (LinearGradientBrush) araştırınız.
- Dairesel renk fırçasını (RadialGradientBrush) araştırınız.
- Resim fırçasını (ImageBrush) araştırınız.
- VideoBrush fırçasını araştırınız.

2. FIRÇALAR

XAML tabanlı *Internet* uygulamalarında nesnelere boyamak için fırçalar kullanılmaktadır. Kullanılan tüm kontrollerin renkleri fırçalar ile değiştirilir. Silverlight uygulamalarında kullanılan fırça çeşitleri ise şunlardır:

- SolidColorBrush
- LinearGradientBrush
- RadialGradientBrush
- ImageBrush
- VideoBrush

2.1. SolidColorBrush

Herhangi bir nesnenin rengini düz renk yapmak için kullanılır. SolidColorBrush tanımlamak için;

```
<SolidColorBrush Color="Red" />
```

 şeklinde yazmamız yeterlidir.

Color özelliği kullanılacak rengi belirler.

Örnek: Aşağıdaki XAML kodunu yazalım.

```
<Grid>
  <Rectangle Width="200" Height="40" Stroke="Blue" StrokeThickness="1">
    <Rectangle.Fill>
      <SolidColorBrush Color="Red" />
    </Rectangle.Fill>
  </Rectangle>
</Grid>
```

Ekran çıktısı:



Resim 2.1: SolidColorBrush kullanımı

2.2. LinearGradientBrush

<LinearGradientBrush> etiketi geçiş renklerinin tanımlandığı etikettir. En önemli özellikleri Offset, StartPoint ve EndPoint özellikleridir.

<LinearGradientBrush> içine, her bir geçiş rengi için <GradientStop> etiketleri oluşturulmalıdır.

<GradientStop> etiketi içinde kullanılan Offset özelliği önemlidir. Bu özellik iki renk arasındaki geçişin mesafesini belirler.

Örnek:

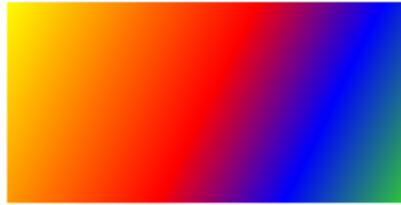
```
<Grid>
  <Rectangle Width="200" Height="100">
    <Rectangle.Fill>
      <LinearGradientBrush StartPoint="0,0" EndPoint="1,1">
        <GradientStop Color="Yellow" Offset="0.0" />
        <GradientStop Color="Red" Offset="0.50" />
        <GradientStop Color="Blue" Offset="0.75" />
        <GradientStop Color="LimeGreen" Offset="1.0" />
      </LinearGradientBrush>
    </Rectangle.Fill>
  </Rectangle>
</Grid>
```

Bu kodlar içinde yer alan aşağıdaki kodlara dikkat edelim.

```
<GradientStop Color="Yellow" Offset="0.0" />  
<GradientStop Color="Red" Offset="0.50" />  
<GradientStop Color="Blue" Offset="0.75" />  
<GradientStop Color="LimeGreen" Offset="1.0" />
```

- İlk satırda doğrusal renk geçişinin ilk rengi sarı olarak belirlenmiştir. İlk renk olduğu için Offset değeri 0.0 yapılmıştır.
- İkinci satırda, ikinci renk olarak kırmızı seçilmiştir. Offset değeri 0.50 yapılmıştır. Bunun anlamı ise şudur. Tüm boyama alanı 1 birim olarak düşünülürse, 0.0 noktasında tam sarı olan renk, kırmızıya doğru geçiş yaparak, alanın ilk (0.0) noktasından 0.50 birim kadar uzaklıkta (yani tüm alanın %50'si kadar uzaklıkta) tam kırmızı olur.
- Üçüncü satırda ise mavi renk tanımlanmış ve Offset değeri 0.75 yapılarak, bir önceki satırda belirlenen kırmızı renkten maviye doğru geçişin, boyama alanının %75'i kadarlık mesafede sona ermesi sağlanır.
- Dördüncü satırda renk olarak yeşil belirlenmiştir. Offset değeri 1.0 yapılarak maviden yeşile tam olarak geçişin boyama alanının son noktasında olması sağlanmıştır. Hatırlanacağı gibi ilk nokta 0.0 ve son nokta 1.0'dır.

Ekran çıktısı:



Resim 2.2: Linear gradient brush

<LinearGradientBrush> içinde kullanılan önemli iki özellik daha vardır. Bunlar StartPoint ve EndPoint özellikleridir. Bu özellikler renk geçişinin hangi yöne doğru olacağını belirler.

Örneğin

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1"> gibi bir kullanımda renk geçişi Resim 2.2 de görüldüğü gibi sol üst köşeden sağ alt köşeye doğru çapraz olacaktır.

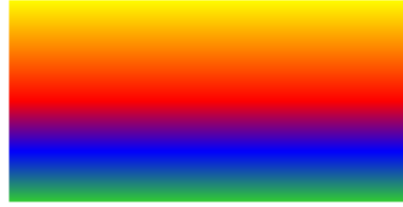
<LinearGradientBrush StartPoint="0,0" EndPoint="1,0"> şeklinde kullanılırsa, renk geçişi aşağıda görüldüğü gibi soldan sağa doğru olacaktır.



Resim 2.3: Soldan sağıya renk geçişi

`<LinearGradientBrush StartPoint="0,0" EndPoint="0,1">`

Şeklinde kullanılırsa, renk geçişi aşağıda görüleceği gibi yukarıdan aşağıya doğru olacaktır.



Resim 2.4: Yukarıdan aşağıya renk geçişi

StartPoint ve **EndPoint** özelliklerinde kullanılan köşe değerleri aşağıda gösterilmiştir.

0,0	1,0
0,1	1,1

Resim 2.5: Köşe değerleri

2.3. RadialGradientBrush

`<RadialGradientBrush>` etiketi dairesel renk geçişleri oluşturmak için kullanılır. Bu etiketin içine, her bir renk tanımlaması için `<GradientStop>` etiketi yazılmalıdır. `<GradientStop>` etiketinin `Color` özelliği, geçiş rengini belirlemek için kullanılır. Doğrusal renk geçişlerinde kullanılan `Offset` özelliği dairesel renk geçişlerinde de kullanılır.

`Offset` özelliği renk geçişleri arasındaki mesafeyi belirler. Minimum 0 ve maksimum 1 değerlerini alabilir.

Önemli bir özelliği de `GradientOrigin` özelliğidir. Bu özellik renk geçişinin başlayacağı merkez noktasını tanımlamak için kullanılır.

Örnek:

```
<Grid>
  <Ellipse Width="300" Height="300">
    <Ellipse.Fill>
      <RadialGradientBrush GradientOrigin="0.3,0.3" >
        <GradientStop Color="White" Offset="0"/>
        <GradientStop Color="Black" Offset="1"/>
      </RadialGradientBrush>
    </Ellipse.Fill>
  </Ellipse>
</Grid>
```

`GradientOrigin="0.3,0.3"` kodu kullanılarak merkez noktası x ekseninde, soldan tüm alanın %30'u kadar uzaklıkta, ve y ekseninde üstten tüm alanın %30'u kadar aşağıda olması sağlanmıştır.

Ekran Çıktısı:



Resim 2.6: RadialGradient Uygulaması

2.4. ImageBrush

`<ImageBrush>` etiketi bir resmi, bir nesneye desen olarak uygulamak amacıyla kullanılır. `<ImageBrush>` etiketinin `ImageSource` özelliği desen yapılacak resmin kaynak dosyasını belirlemek amacıyla kullanılır.

Örnek:

```
<Grid>
  <Rectangle Width="100" Height="100">
    <Rectangle.Fill>
      <ImageBrush ImageSource="Penguins.jpg"></ImageBrush>
    </Rectangle.Fill>
  </Rectangle>
</Grid>
```

Ekran çıktısı:



Resim 2.7: ImageBrush kullanımı

2.5. VideoBrush

Bir video içeriği ile bir alanı boyamak için kullanılır. Kullanılacak video içeriği bir MediaElement kontrolü tarafından sağlanır. VideoBrush geometrik şekillere ya da TextBlock uygulanabilir.

Örnek:

```
<Canvas
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

  <MediaElement
    x:Name="MediaElement1"
    Source="Wildlife.wmv" IsMuted="True"
    Opacity="0.0" IsHitTestVisible="False" />

  <TextBlock Canvas.Left="5" Canvas.Top="30"
    FontFamily="Verdana" FontSize="120"
    FontWeight="Bold" TextWrapping="Wrap"
    Text="Video">

    <TextBlock.Foreground>
      <VideoBrush SourceName="MediaElement1" Stretch="UniformToFill" />
    </TextBlock.Foreground>
  </TextBlock>
</Canvas>
```


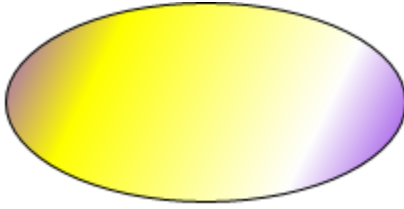
Ekran ıktısı:



Resim 2.8: VideoBrush kullanımı

UYGULAMA FAALİYETİ

Aşağıdaki önerilere göre işlem basamaklarını uygulayınız

İşlem Basamakları	Öneriler
<p>➤ SolidColorBrush fırçasını kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Ellipse Height="100" HorizontalAlignment="Left" Width="200" Canvas.Left="108" Canvas.Top="100"> <Ellipse.Fill> <SolidColorBrush Color="Red"/> </Ellipse.Fill> </Ellipse> </Canvas></pre>
<p>➤ LinearGradientBrush renk fırçasını kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Ellipse Height="100" Name="ellipse1" Stroke="Black" Width="200" Canvas.Left="12" Canvas.Top="85"> <Ellipse.Fill> <LinearGradientBrush> <GradientStop Color="RosyBrown" Offset="0.1"/> <GradientStop Color="Yellow" Offset="0.25"/> <GradientStop Color="White" Offset="0.75"/> <GradientStop Color="BlueViolet" Offset="1"/> </LinearGradientBrush> </Ellipse.Fill> </Ellipse> </Canvas></pre>
<p>➤ RadialGradientBrush renk fırçasını kullanarak aşağıdaki tasarımı yapınız.</p>	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Grid> <Rectangle Height="238" Margin="67, 33, 0, 0" Name="rectangle1" Stroke="Black" Width="343" > <Rectangle.Fill> <RadialGradientBrush GradientOrigin="0.9, 0.3"></pre>

	<pre><GradientStop Color="Black" Offset="0.2"/> <GradientStop Color="Red" Offset="0.50"/> <GradientStop Color="AntiqueWhite" Offset="0"/> <GradientStop Color="AntiqueWhite" Offset="1"/> </RadialGradientBrush> </Rectangle.Fill> </Rectangle> </Grid></pre>
<p>➤ Resim fırçasını (ImageBrush) kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki tasarımı oluşturunuz.</p> <p>➤ Penguins.jpg dosyasını Belgelerim klasöründen ClientBin klasörüne kopyalayınız.</p> <pre><Grid> <Rectangle Height="150" Margin="46, 32, 0, 0" Name="rectangle1" Stroke="Black" Width="200"> <Rectangle.Fill> <ImageBrush ImageSource="Penguins.jpg"/> </Rectangle.Fill> </Rectangle> </Grid></pre>
<p>➤ Video fırçasını kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Wildlife.wmv video dosyasını Belgelerim klasöründen ClientBin klasörüne kopyalayınız.</p> <p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <MediaElement x:Name="MediaElement1" Source="Wildlife.wmv" Visibility="Collapsed" /> <TextBlock Canvas.Left="5" Canvas.Top="30" FontFamily="Verdana" FontSize="60" FontWeight="Bold" Text="Video"> <TextBlock.Foreground> <VideoBrush SourceName="MediaElement1" Stretch="UniformToFill" /> </TextBlock.Foreground> </TextBlock> </Canvas></pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Tek renk fırçasını (SolidColorBrush) kullandınız mı?		
2. Doğrusal olarak değişen (LinearGradientBrush) renk fırçasını kullandınız mı?		
3. Yayılarak değişen(RadialGradientBrush) renk fırçasını kullandınız mı?		
4. Resim fırçasını (ImageBrush) kullandınız mı?		
5. Video fırçasını (VideoBrush) kullandınız mı?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi fırça çeşitlerinden **değildir**?
A) ImageBrush B) SolidColorBrush
C) VisualBrush D) RadialGradientBrush
2. Aşağıdakilerden hangisi kontrol içerisinde düz renklerle boyama yapmak için kullanılan fırça çeşididir?
A) ImageBrush B) SolidColorBrush
C) RadialGradientBrush D) LinearGradientBrush
3. Aşağıdakilerden hangisi dairesel renk geçişleri için kullanılan fırça çeşididir?
A) RadialGradientBrush B) LinearGradientBrush
C) SolidColorBrush D) VideoBrush
4. Aşağıdakilerden hangisi renk geçişlerini yukarıdan aşağıya doğru yapar?
A) StartPoint="0,0" EndPoint="1,0"
B) StartPoint="0,0" EndPoint="0,1"
C) StartPoint="1,1" EndPoint="0,0"
D) StartPoint="1,1" EndPoint="1,0"
5. Aşağıdakilerden hangisi doğrusal renk geçişlerinde kullanılan fırça çeşididir?
A) SolidColorBrush B) RadialGradientBrush
C) VideoBrush D) LinearGradientBrush
6. Aşağıdakilerden hangisi renk geçişlerini soldan sağa doğru yapar?
A) StartPoint="0,0" EndPoint="1,0"
B) StartPoint="0,0" EndPoint="0,1"
C) StartPoint="1,1" EndPoint="0,0"
D) StartPoint="1,1" EndPoint="1,0"
7. Aşağıdakilerden hangisi video fırçasıdır?
A) VideoBrush B) ImageBrush
C) LinearGradientBrush D) SolidColorBrush
8. Aşağıdakilerden hangisi resim fırçasıdır?
A) VideoBrush B) ImageBrush
C) RadialGradientBrush D) LinearGradientBrush

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-3

AMAÇ

Görsel özellikleri kullanarak uygulama geliştirebileceksiniz.

ARAŞTIRMA

- Ölçü ve yerleşim birim ve özelliklerini araştırınız.
- Şeffaflık (Opacity) özelliği hakkında bilgi edininiz.
- İşaretçi (Cursor) tipleri hakkında bilgi edininiz.
- Stroke (Çerçeve rengi) hakkında bilgi edininiz.

3. GÖRSEL ÖZELLİKLER

3.1. Ölçü ve Yerleşim Özellikleri

Genişlik (Width), yükseklik (Height), Margin, Top, Left gibi özelliklerde kullanılabilen ölçü birimleri aşağıdaki tabloda yer almaktadır.

Atama Değeri	Değerin Adı	Açıklama
px	Pixel (Piksel)	
in	Inches (İnç)	1in = 96px
cm	Centimeters (Santimetre)	1cm = 37.7 px
pt	Points (Nokta)	1pt = 1.333 px

Tablo 3.1: Ölçü birimleri

Kontrollere ilişkin bazı önemli özellikler ise aşağıda belirtilmiştir.

- **Width:** Nesnenin genişlik değerini belirler.
- **Height:** Nesnenin yükseklik değerini belirler.
- **Canvas.Top:** Nesnenin bulunduğu canvas'ın üst kenarına göre üst konumunu ayarlar.
- **Canvas.Left:** Nesnenin bulunduğu canvas'ın sol kenarına göre sol konumunu ayarlar.

Örneğin 100 piksel genişliğinde ve 200 piksel yüksekliğinde bir dikdörtgen oluşturmak için şöyle bir XAML kodu kullanmamız gerekecektir.

```
<Rectangle Fill="Black" Width="100" Height="200" />
```

Eğer cm cinsinden genişlik ve yükseklik değerleri kullanmak istiyorsak width ve height özelliklerini aşağıdaki gibi kullanmalıyız.

```
<Rectangle Fill="Black" Width="2cm" Height="4cm" />
```

Margin: Bir nesnenin sol, üst, sağ ve alt kenarlardan uzaklığını belirler. Kullanımı şöyledir. Margin = “sol,üst,sag,alt”

Örnek: Margin=”20,50,20,50”

3.2. Opacity

Opacity, nesnelere saydamlık ayarı yapmak için kullanılan bir özelliktir. 0 ile 1 arasında bir değer alır.

Örnek:

```
<Grid>  
<Image Width="300" Height="300" Source="Penguins.jpg" Opacity="0.25" />  
</Grid>
```



(a)



(b)

Resim 3.1: Resimde %25 opacity ayarı

3.3. Cursor

Her hangi bir kontrol için fare imlecinin şeklini değiştirmek için kullanılan bir özelliktir. Arrow, Eraser, Hand, Default, Hand, Ibeam, None, Wait, SizeWS, SizeNE, Stylus değerlerini alabilir.

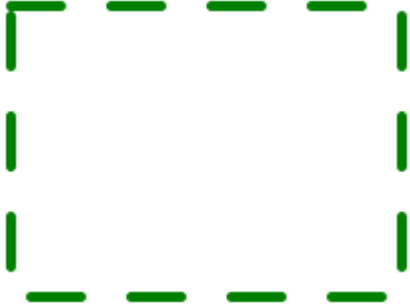
3.4. Stroke

Dış çerçeve rengini belirlemek için kullanılır. Bunun yanı sıra aşağıdaki özellikler de çerçeve ayarlarını yapmak için kullanılır.

- **StrokeThickness:** Dış çerçeve kalınlığını ayarlar.
- **StrokeDashCap:** Şeklin dış kenarlarını nasıl bir kesikli çizgi ile gösterileceği belirtilir.
- **StrokeDashArray:** Kesikli kenarlarda kesikli çizginin ne kadar boşlukta tekrarlanacağını ve kesik parçaların uzunluğunu belirtir.

UYGULAMA FAALİYETİ

Aşağıdaki önerilere göre işlem basamaklarını uygulayınız.

İşlem Basamakları	Öneriler
<p>➤ Width, Height, Top ve Left özelliklerini kullanınız.</p>	<p>➤ Aşağıdaki XAML kodlarını yazınız.</p> <pre><Canvas> <Ellipse Width="100" Height="100" Canvas.Top="50" Canvas.Left="80" Stroke="Black"/> </Canvas></pre>
<p>➤ Opacity ayarını yapınız.</p>	<p>➤ Aşağıdaki XAML kodlarını yazınız.</p> <pre><Canvas> <Button Content="TIKLA" Opacity="0.40" Canvas.Left="133" Canvas.Top="102" /> </Canvas></pre>
<p>➤ Cursor özelliğini değiştiriniz.</p>	<p>➤ Aşağıdaki XAML kodlarını yazınız.</p> <pre><Canvas> <Button Content="TIKLA" Cursor="Stylus" Canvas.Left="133" Canvas.Top="102" /> </Canvas></pre>
<p>➤ Stroke özelliklerini kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Aşağıdaki XAML kodlarını yazınız.</p> <pre><Canvas> <Rectangle Width="200" Height="150" Stroke="Green" StrokeThickness="5" StrokeDashArray="5" StrokeDashCap="Round" Canvas.Left="75" Canvas.Top="57" /> </Canvas></pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Ölçü ve yerleşim özelliklerini kullandınız mı?		
2. Saydamlık (Opacity) ayarını yaptınız mı?		
3. İşaretçi (Cursor) özelliğini kullandınız mı?		
4. Çerçeve (Stroke) özelliklerini kullandınız mı?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümleleri dikkatlice okuyarak boş bırakılan yerlere doğru sözcüğü yazınız.

1. inç px (piksel) değerindedir.
2. cm pixel değerindedir.
3. pt (point) px değerindedir.

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

4. Nesnenin tüm kenarlara göre ayrı ayrı mesafesini ayarlayabilen özellik aşağıdakilerden aşağıdakilerden hangisidir?
A) Piksel B) Margin C) Opacity D) Stroke
5. Nesnelere saydamlık için kullanılan özellik aşağıdakilerden hangisidir?
A) Cursor Behavior
B) Margin
C) Opacity
D) Stroke
6. Nesnelerin üzerinde fare işaretçisi şeklinin değiştirmek için aşağıdaki özelliklerden hangisinin kullanılmalıdır?
A) Cursor
B) Margin
C) Opacity
D) Stroke

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

7. () Dış çerçeve rengine Stroke denir.
8. () StrokeDashCap, dış çerçeve kalınlığını ayarlayan özelliktir.
9. () StrokeDashArray, kesikli kenarlarda kesikli çizginin ne kadar boşlukta tekrarlanacağını belirtir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-4

AMAÇ

Şekil işlemlerini gerçekleştirebileceksiniz.

ARAŞTIRMA

- Silverlight ile kullanılan geometrik şekil kontrollerini araştırınız.
- Path (Yol) nesnesine ait özellikleri araştırınız.

4. ŞEKİLLER

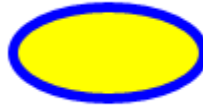
Uygulama formlarında dikdörtgen, elips gibi geometrik şekiller oluşturmanın yanı sıra, eğriler kullanarak da bazı çizimler oluşturabiliriz. Ayrıca geometrik nesnelere gruplar hâline getirebiliriz.

4.1. Elips (Ellipse)

Elips, dairesel çizimler yapmak için kullanılır. Aşağıdaki örneği inceleyelim.

```
<Grid>  
<Ellipse Width="100" Height="50"  
  Stroke="Blue" StrokeThickness="5" Fill="Yellow"  
  HorizontalAlignment="Left" VerticalAlignment="Top"  
  Margin="10, 10, 0, 0" />  
</Grid>
```

XAML kodunu yazdıktan sonra form üzerinde elips şeklinin oluştuğunu görebiliriz.



Resim 4.1: Ellipse kullanımı

Elips çizerken kullandığınız özellikler ise şunlardır:

Stroke: Çerçeve rengi

StrokeThickness: Çerçeve kalınlığı

Fill: Dolgu rengi

4.2. Dikdörtgen (Rectangle)

Dörtgen çizimleri için kullanılır. Aşağıdaki örneği inceleyelim.

```
<Grid>  
  <Rectangle Width="200" Height="100" Fill="Blue" Stroke="Black"  
  StrokeThickness="3"/>  
</Grid>
```

Kodun çalıştığında dolgu rengi mavi olan (Fill="Blue"), çerçeve rengi siyah (Stroke="Black") ve çerçeve kalınlığı 3 olan (StrokeThickness="3") bir dikdörtgen oluşacaktır.



Resim 4.2: Dikdörtgen (Rectangle)

Dikdörtgene ait köşelerin yuvarlatılması için RadiusX ve RadiusY özelliklerine sayısal değer atanması gereklidir.



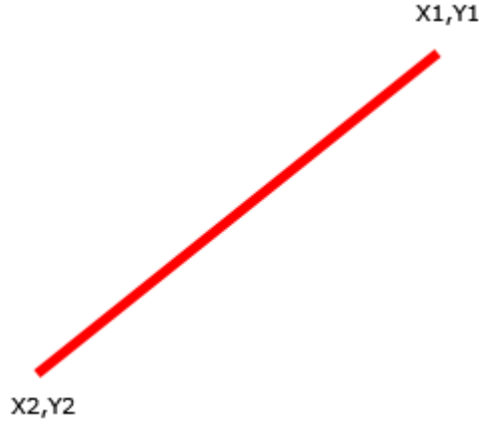
Resim 4.3: Dikdörtgen köşe yuvarlaklığı (RadiusX, RadiusY)

4.3. Çizgi (Line)

Çizgi çizmek için kullanılır. Aşağıdaki örneği inceleyelim.

```
<Grid>  
  <Line X1="250" Y1="40" X2="50" Y2="200" Stroke="Red" StrokeThickness="5"/>  
</Grid>
```

Kod çalıştırıldığında X1=250,Y1=40 koordinatındaki birinci noktadan X2=50,Y2=200 koordinatındaki ikinci noktaya doğru çizgi çizildiği görülecektir.



Resim 4.4:Çizgi kullanımı

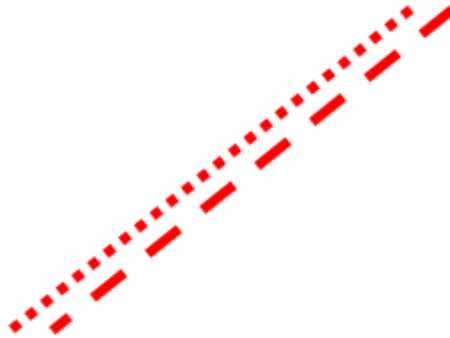
Çizgilerin kesik çizgi olması için StrokeDashArray özelliği kullanılabilir.

Örnek:

```
<Grid>  
<Line X1="250" Y1="40" X2="50" Y2="200" StrokeDashArray="1,1" Stroke="Red"  
StrokeThickness="5" Margin="-33, -12, 33, 12" />  
<Line X1="250" Y1="40" X2="50" Y2="200" StrokeDashArray="4,3" Stroke="Red"  
StrokeThickness="5" Margin="-12, -12, 12, 12" />  
</Grid>
```

StrokeDashArray="4,3" kodunda kullanılan 4 kesik çizgi parçalarının uzunluğunu, 3 değeri ise parçalar arasındaki mesafeyi belirler.

Ekran çıktısı:



Resim 4.5: Kesik çizgi kullanımı

4.4. Yol (Path)

Temel şekillerin uç uca eklenmesi veya birleştirilmesi ile oluşturulan karmaşık çizimlerdir. Path sınıfının data özelliği kullanılarak geometrik şekil çizimleri ya da eğri çizimleri yapılabilir.

Data özelliği içinde şekle ait köşe noktaları, mesafe değerleri gibi veriler bulunmalıdır. Data özelliğinde bazı harfler kullanılarak şekle ait temel parametreler belirlenir. Bu harfler ise aşağıda verilmiştir.

M Harfi: Şeklin ilk noktasını belirler.

L Harfi: Çizgi çizmeye yarar.

H Harfi: Yatay çizgi çizmeye yarar.

V Harfi: Dikey çizgi çizmeye yarar.

C, Q-S-P Harfleri: Çeşitli biçimlerde eğriler çizer.

A Harfi: Elips çizmeye yarar.

Z: İlk nokta ile son nokta arasını kapatmayı sağlar.

Örnek: Path kullanarak üçgen çizimi

```
<Grid>  
<Path Data="M 0,200 L100,200 50,50z"  
Stroke="Black" Fill="Gray"  
Canvas.Left="150" Canvas.Top="70" />  
</Grid>
```

Ekran çıktısı:



Resim 4.6:Path kullanımı

Örnek: Path kullanarak eğri çizimi

```
<Grid>  
<Path Data="M 10,100 C 10,300 300,-200 250,100z"  
Stroke="Red" Fill="Orange"  
Canvas.Left="10" Canvas.Top="10" />  
</Grid>
```

Ekran çıktısı:



Resim 4.7: Path kullanarak eğri çizimi

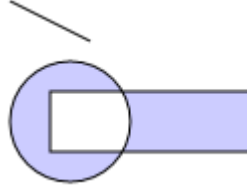
4.5. Çoklu Geometrik Nesnelere (GeometryGroup)

Geometrik şekilleri birleştirmek ve kesişimlerini almak için kullanılır. Path etiketi içerisinde kullanılır. Aşağıdaki örneği inceleyelim.

```
<Canvas>  
<Path Stroke="Black" StrokeThickness="1" Fill="#CCCCFF">  
<Path.Data>  
<GeometryGroup FillRule="EvenOdd">  
<LineGeometry StartPoint="10,10" EndPoint="50,30" />  
<EllipseGeometry Center="40,70" RadiusX="30" RadiusY="30" />  
<RectangleGeometry Rect="30,55 100 30" />  
</GeometryGroup>  
</Path.Data>  
</Path>  
</Canvas>
```

`<GeometryGroup>` etiketinin `FillRule` özelliği "EvenOdd" ya da "NonZero" değerlerinden birini alabilir. Bu değerler geometrik şekillerin kesişimi alınıp alınmayacağını belirler.

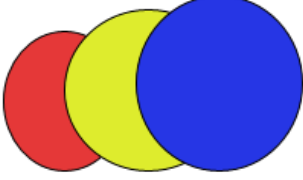

Ekran ıktısı:


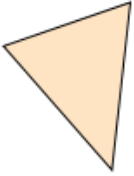
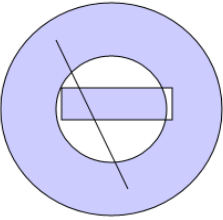


Resim 4.8: GeometryGroup kullanımı

UYGULAMA FAALİYETİ

Aşağıdaki önerilere göre işlem basamaklarını uygulayınız.

İşlem Basamakları	Öneriler
<p>➤ Ellipse kontrolleri kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Grid> <Ellipse Height="91" HorizontalAlignment="Left" Margin="71,64,0,0" Name="ellipse1" Stroke="Black" StrokeThickness="1" VerticalAlignment="Top" Width="81" Fill="#FFE53838" /> <Ellipse Height="105" HorizontalAlignment="Left" Margin="111,50,0,0" Name="ellipse2" Stroke="Black" StrokeThickness="1" VerticalAlignment="Top" Width="111" Fill="#FFDEEC2D" /> <Ellipse Height="114" HorizontalAlignment="Left" Margin="158,41,0,0" Name="ellipse3" Stroke="Black" StrokeThickness="1" VerticalAlignment="Top" Width="110" Fill="#FF2636E5" /> </Grid></pre>
<p>➤ Rectangle kontrolleri kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Rectangle Fill="#FFE5D82F" HorizontalAlignment="Left" Stroke="Yellow" Width="105" RadiusX="20" RadiusY="42" Height="80" Canvas.Top="67" Canvas.Left="118" /> <Rectangle Fill="#FF707087" Stroke="Black" StrokeThickness="8" Height="69" Width="79" Canvas.Left="33" Canvas.Top="73" /> <Rectangle Fill="#FF9191EB" HorizontalAlignment="Right" StrokeDashArray="6" StrokeThickness="5" StrokeDashCap="Flat" RadiusY="1" Stroke="Black" Width="85" Height="78" Canvas.Left="229" Canvas.Top="69" /> </Canvas></pre>

<p>➤ Line kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Grid> <Line X1="50" X2="450" Y1="150" Y2="150" Stroke="Red" StrokeDashArray="1" StrokeDashCap="Flat" StrokeThickness="5"/> </Grid></pre>
<p>➤ Path kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Path Fill="Bisque" Data="M 91,108 L 324,40 290,300z" Margin="77, 51.5, 197.5, 90.5" Stretch="Fill" Stroke="Black" StrokeThickness="1" Width="74" Height="98" Canvas.Left="-17" Canvas.Top="-8" /> </Canvas></pre>
<p>➤ Geometrik gruplar oluşturarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Xaml kodları ile yandaki resimdeki formu oluşturunuz.</p> <pre><Canvas> <Path Stroke="Black" StrokeThickness="1" Fill="#CCCCFF" Margin="123, 80, 82, 12"> <Path.Data> <GeometryGroup FillRule="EvenOdd"> <EllipseGeometry RadiusX="50" RadiusY="50" Center="75, 75"/> <LineGeometry StartPoint="25, 10" EndPoint="90,150"/> <EllipseGeometry RadiusX="100" RadiusY="100" Center="75,75"/> <RectangleGeometry Rect="30, 55 100 30" /> </GeometryGroup> </Path.Data> </Path> </Canvas></pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Uygulama içinde kullanılan elips şekline ait düzenlemeleri yaptınız mı?		
2. Uygulama içinde kullanılan dörtgen şekline ait düzenlemeleri yaptınız mı?		
3. Uygulama içinde kullanılan çizgi şekline ait düzenlemeleri yaptınız mı?		
4. Uygulama içinde tanımlanan yola ait düzenlemeleri yaptınız mı?		
5. Geometrik gruplar oluşturduunuz ve düzenlediniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınızı “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi daire çizimleri için kullanılır?
A) Rectangle B) Ellipse C) Path D) Line
2. Aşağıdakilerden hangisi “Ellipse Width=”100” Height=”50” Stroke=”Yellow” StrokeThickness=”10” ” kodlarının doğru açılımıdır?
A) Elips genişliği 100, yüksekliği 50, dış çerçeve rengi sarı, dış çerçeve kalınlığı 10
B) Daire yüksekliği 100, genişliği 50, dış çerçeve rengi sarı, dış çerçeve kalınlığı 1
C) Elips yüksekliği 100, genişliği 50, dış çerçeve rengi mavi, dış çerçeve kalınlığı 1
D) Daire genişliği 100, yüksekliği 50, dış çerçeve rengi mavi, dış çerçeve kalınlığı 10
3. Aşağıdakilerden hangisi “Rectangle Width=”200” Height=”100” Fill=”Blue” Stroke=”Black “ ” kodlarının doğru açılımıdır?
A) Elips genişliği 200, yüksekliği 100, iç alan rengi mavi, dış çerçeve rengi siyah
B) Dörtgen genişliği 200, yüksekliği 100, iç alan rengi mavi, dış çerçeve rengi siyah
C) Elips genişliği 200, yüksekliği 100, iç alan rengi mavi, iç çerçeve rengi siyah
D) Dörtgen genişliği 200, yüksekliği 100, iç alan rengi mavi, iç çerçeve rengi siyah
4. Aşağıdakilerden hangisi dikdörtgene ait köşelerin yuvarlatılması için kullanılan koddur?
A) Rectangle B) Stroke C) RadiusX D) Path
5. Aşağıdakilerden hangisi çizgi çizmek için kullanılır?
A) Rectangle B) Ellipse C) Path D) Line
6. Aşağıdakilerden hangisi çizgileri kesik çizgi haline getirmeyi sağlayan özelliktir?
A) RadiusX B) StrokeThickness C) StrokeDashArray D) Stroke
7. Aşağıdakilerden hangisi temel şekilleri uç uca eklenmesi veya birleştirilmesi ile oluşturulan karmaşık çizimlerdir?
A) Rectangle B) Ellipse C) Path D) Line
8. Aşağıdakilerden hangisi Geometrik şekilleri birleştirmek ve kesişimlerini almak için kullanılır?
A) Rectangle B) GeometryGroup C) Path D) Line

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-5

AMAÇ

Kontrolleri kullanabileceksiniz.

ARAŞTIRMA

- Resim (Image) kontrolünü araştırınız.
- Metin kontrollerini araştırınız.

5. KONTROLLER

Bu bölümde uygulama geliştirmede kullandığımız bazı kontrollere ilişkin bilgiler verilecektir.

5.1. Resim (Image)

Uygulama formları üzerine resim eklemek için kullanılır. <Image> etiketinin en önemli özelliği Source özelliğidir. Source özelliği, görüntülenecek resmin kaynak dosyasını belirleyen özelliktir.

Aşağıdaki örneği inceleyelim. Uygulamaya geçmeden önce resmin kaynak dosyası Belgelerim klasöründen projenin ClientBin klasörüne kopyalanmalıdır.

Örnek:

```
<Canvas>  
<Image Source="Tulips.jpg" Width="200" Height="200" Stretch="Fill" Canvas.Left="77"  
Canvas.Top="27" />  
</Canvas>
```

Ekran çıktısı:



Resim 5.1: Path kullanarak eğri çizimi

Diğer önemli bir özelliği de Stretch özelliğidir. Stretch özelliği resmin, Image alanına nasıl yerleşeceğini belirleyen özelliktir. Stretch özelliğinin alabileceği değerler ve anlamları ise şöyledir.

Uniform: Stretch özelliğinin varsayılan değeri olan Uniform ile resmin en-boy oranı bozulmadan Image içerisine yerleştirilmesi sağlanır.

UniformToFill: Bu seçenekte resmin en boy oranı korunurken resmin boyu veya eni tam olarak Image içerisini dolduracak şekilde büyütülür ve sığmayan kesimler Image dışında tutularak kesilir.

Fill: Fill seçeneği kullanıldığında resim Image içerisine sığacak şekilde tekrar boyutlandırılır. Bu seçenekte resmin en-boy oranı korunmaz.

5.2. Kabartma (Glyphs)

Kabartma (Glyphs) özel bir font dosyasında bulunan font özelliklerini uygulamamızda kullanabilmemizi sağlar. Aşağıdaki XAML kodunu inceledikten sonra <Glyphs> kontrolüne dair bazı özelliklere değineceğiz.

Örneğe geçmeden önce “webdings.ttf” font dosyasının projenin SilverlightApplication tarafına kopyalanması gerekmektedir.

Örnek:

```
<Canvas>
  <Glyphs Width="147" Height="93" Canvas.Left="29" Canvas.Top="76"
  Fill="#FF000000" FontRenderingEmSize="72" UnicodeString="MERHABA"
  FontUri="webdings.ttf"/>
</Canvas>
```

Ekran görüntüsü:



Resim 5.2: Glyph kullanımı

<Glyph> etiketine dair bazı özellikler ise şunlardır:

FontRenderingEmSize: Font büyüklüğünü ayarlar.

UnicodeString: Yazdırılacak metni belirler.

FontUri: Font dosyasının yolunu belirler.

5.3. Metin Bloęu (TextBlock)

Metin Bloęu (TextBlock) bir metni form üzerinde göstermeyi saęlar. Ařaęıdaki örneęi inceleyiniz.

```
<Canvas>  
  <TextBlock Text="MERHABA" FontSize="20" FontStyle="Italic"  
  FontFamily="Lucida Sans Unicode" FontWeight="Bold" ></TextBlock>  
</Canvas>
```

Ekran görüntüsü:

MERHABA


Resim 5.3: TextBlock kullanımı

TextBlock kontrolüne ilişkin bazı önemli özellikler ise řunlardır.

- **Text:** Görüntülenecek metin içerięi bu özellięe atanmalıdır.
- **FontSize:** Metnin büyüklüğünü ayarlar.
- **FontStyle:** Metnin eğik olup olmayacağını belirler.
- **FontFamily:** Yazı tipini belirler.
- **FontWeight:** Metnin kalın olup olmayacağını belirler.

UYGULAMA FAALİYETİ

Aşağıdaki önerilere göre işlem basamaklarını uygulayınız.

İşlem Basamakları	Öneriler
<p>➤ Resim (Image) kontrolünü kullanarak aşağıdaki tasarımı yapınız.</p> 	<p>➤ Penguins.jpg dosyasını Belgelerim klasöründen projenin ClientBin klasörüne kopyalayınız.</p> <p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Canvas> <Image Source="Penguins.jpg" Width="200" Height="200" Stretch="Fill" Canvas.Left="77" Canvas.Top="27" /> </Canvas></pre>
<p>➤ Metin (TextBlock) kontrolünü kullanarak aşağıdaki metni yazdırınız.</p> <p>WEB TASARIM</p>	<p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Canvas> <TextBlock Text="WEB TASARIM" FontSize="40" Foreground="Red" FontFamily="Trebuchet MS" FontWeight="Bold" > </TextBlock> </Canvas></pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Resim (Image) kontrolünü kullandınız mı?		
2. Glyph (Kabartma) kontrolünü kullandınız mı?		
3. TextBlock kontrolünü kullandınız mı?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi çeşitli font dosyalarını uygulamalarımızda kullanabilmeyi sağlar?
A) Resim (Image)
B) Tuval (Canvas)
C) Metin Bloğu (TextBlock)
D) Kabartma (Glyphs)
2. Aşağıdakilerden hangisi form üzerinde bir metni görüntülemeyi sağlar?
A) Resim (Image)
B) Tuval (Canvas)
C) Metin bloğu (TextBlock)
D) Kabartma (Glyphs)

Aşağıdaki cümleleri dikkatlice okuyarak boş bırakılan yerlere doğru sözcüğü yazınız.

3. Resmin kaynak dosyasını belirlemek için..... özelliğini kullanabiliriz.
4. özelliği TextBlock kontrolünde yazı tipini değiştirir.
5. Glyph nesnesinde font dosyasını özelliği sayesinde ayarlarız.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-6

AMAÇ

Dönüşüm ve animasyon işlemlerini yapabileceksiniz.

ARAŞTIRMA

- Uygulama içinde kullanılan nesnelere döndürme ve eğme ile ilgili özellikleri araştırınız.
- Animasyon oluşturma yöntemlerini araştırınız.

6. DÖNÜŞÜM VE ANİMASYON

Uygulamalarda kullanılan kontrollerin belirli açılarla döndürülmeleri, bükme, boyutlarını değiştirme işlemleri için transform özellikleri kullanılmaktadır. Ayrıca nesnelere hareketlendirmek için de üç farklı animasyon yöntemi kullanılmaktadır.

Temel olarak beş adet dönüşüm işlemi vardır.

- Döndürme (RotateTransform)
- Ölçeklendirme (ScaleTransform)
- Dönüşüm (TranslateTransform)
- Çarpıtma (SkewTransform)
- Matris (MatrixTransform)

6.1. Döndürme (RotateTransform)

Uygulamalarda kullandığımız form kontrollerinin belirli açılarla döndürülmeleri için bu etiket kullanılmaktadır.

Bir nesnenin döndürülebilmesi için o nesnenin etiketleri arasına nesne sınıfı içindeki `RenderTransform` etiketi oluşturulur. `<RotateTransform>` ise `<RenderTransform>` bloğu içerisine yazılır. Bu durum aşağıdaki örnekte daha iyi anlaşılabilir.

Aşağıdaki XAML kodunu inceleyiniz.

Örnek:

```
<Canvas>  
<Image Source="Penguins.jpg" Width="137" Height="146" Canvas.Left="120"  
Canvas.Top="60">  
<Image.RenderTransform>  
<RotateTransform Angle="45" CenterX="50" CenterY="50"/>  
</Image.RenderTransform>  
</Image>  
</Canvas>
```

Ekran görüntüsü:



Resim 6.1: Rotate Transform

<RotateTransform> içerisinde kullanılan en önemli özellik Angle özelliğidir. Bu özellik dönme açısını belirler. Ayrıca CenterX ve CenterY özellikleri de kullanılmaktadır. Bu iki özellik döndürme merkezini sol üst köşeyi referans alarak belli bir mesafeye ayarlamak için kullanılır. CenterX ve CenterY özelliklerine herhangi bir değer atanmadığı takdirde 0,0 kabul edilir ve nesne sol üst köşe merkezli olarak döndürülür.

6.2. Ölçeklendirme (ScaleTransform)

Uygulamalarda kullanılan kontrollerin boyutlarını yatay ve dikey ekseninde değiştirme işlemlerini bu dönüşüm yöntemi ile gerçekleştirebiliriz. <ScaleTransform> etiketi, uygulandığı nesnenin <RenderTransform> etiketi içerisine yazılmalıdır.

ScaleX ve ScaleY özellikleri sayesinde yatay ve dikey eksenindeki büyüme oranlarını belirleyebiliriz.

Örnek:

```
<Canvas>  
<Button Content="Button" Height="23" Margin="79, 160, 0, 0" Name="button1"  
Width="75">  
<Button.RenderTransform>
```

```
<ScaleTransform ScaleX="3" ScaleY="2"></ScaleTransform>
</Button.RenderTransform>
</Button>
<Button Content="Button" Height="23" Margin="91, 110, 0, 0" Name="button2"
Width="75"/>
</Canvas>
```



Resim 6.2: Scale Transform kullanımı

Görüldüğü gibi alttaki butonun genişliği normalin üç katına, yüksekliği ise normalin 2 katına kadar büyümüştür.

6.3. Dönüşüm (TranslateTransform)

Uygulamada kullanılan kontrolleri yatay ve dikey eksenlerinde istenilen birim kadar ötelemek için bu dönüşüm yöntemini kullanabiliriz.

X ve Y özellikleri nesnenin ne kadar öteleneceğini belirler.

Örnek:

```
<Canvas>
<Ellipse Height="100" Margin="114, 90, 0, 0" Name="ellipse1" Stroke="Black"
Width="200" Fill="#FF935E5E">
  <Ellipse.RenderTransform>
    <TranslateTransform X="25" Y="5"></TranslateTransform>
  </Ellipse.RenderTransform>
</ Ellipse >
</Canvas>
```

Ekran görüntüsü:



Resim 6.3: Translate Transform

6.4. Çarpıtma (SkewTransform)

Uygulama tasarımında kullandığımız kontrolleri X ve Y eksenlerinde bükme için kullanılan dönüşüm yöntemidir.

AngleX ve AngleY özelliklerine atanan açı değerleri yardımıyla bükme işlemi gerçekleştirilir.

Örnek:

```
<Canvas>
<Image Height="121" Margin="85,39, 0, 0" Name="image1" Stretch="Fill"
Width="200" Source="Penguins.jpg" Canvas.Left="-50" Canvas.Top="-14">
  <Image.RenderTransform>
    <SkewTransform AngleX="25" AngleY="30"/>
  </Image.RenderTransform>
</Image >
</Canvas>
```

Ekran görüntüsü:



Resim 6.4: Skew Transform

6.5. Matris (MatrixTransform)

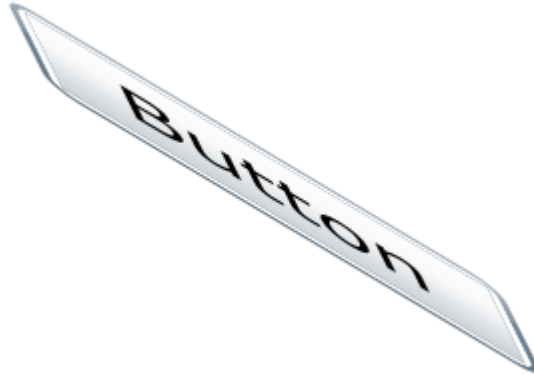
Uygulama tasarımında kullandığımız kontrollere 3x3 matris değerleri kullanarak dönüştürme işlemi yapmayı sağlar. Daha çok resim işleme yöntemlerinde kullanılmaktadır.

- M11: Matrisin (1,1) elemanıdır.
- M12: Matrisin (1,2) elemanıdır.
- M21: Matrisin (2,1) elemanıdır.
- M22: Matrisin (2,2) elemanıdır.
- OffsetX: Matrisin (3,1) elemanıdır.
- OffsetY: Matrisin (3,2) elemanıdır.

Örnek:

```
<Canvas>
  <Button Content="Button" Height="23" Margin="37,20, 0, 0" Name="button1"
  Width="75">
    <Button.RenderTransform>
      <MatrixTransform>
        <MatrixTransform.Matrix>
          <Matrix OffsetX="0.2" OffsetY="1" M11="4.2" M12="2.5" M21="1"
          M22="2"/>
        </MatrixTransform.Matrix >
      </MatrixTransform>
    </Button.RenderTransform>
  </Button >
</Canvas>
```

Ekran görüntüsü:



Resim 6.5: Matrix Transform

6.6. Senaryo (Storyboard)

XAML tabanlı gelişmiş *İnternet* uygulamalarında animasyon işlemlerinin gerçekleştiği sınıf `<Storyboard>` sınıfıdır.

Storyboard sınıfının metodlarından bazıları ve işlevleri aşağıdaki gibidir.

- **Begin():** Animasyonu başlatmak için kullanılır.
- **GetCurrentState():** Animasyon üzerinde mevcut konumu almayı sağlar.
- **GetCurrentTime():** Animasyon üzerinde mevcut zamanı almayı sağlar.
- **Pause():** Animasyonun duraklatılmasını sağlar.
- **Resume():** Animasyonun duraklatıldıktan sonra, kaldığı yerden devamını sağlar.
- **Stop():** Animasyonun durdurulmasını sağlar.
- **Seek():** TimeSpan türünde bir zaman alır ve animasyonun bu zamana atlamasını sağlar.

- **SkipToFill():** Animasyonun sonuna atlama yapmayı sağlar. Animasyonu sonlandırır.

BeginStoryboard olayı gerçekleştiği zaman Storyboard'u başlatır.

6.7. Animasyon Çeşitleri

XAML tabanlı gelişmiş İnternet uygulamalarında kullanılan animasyon türleri aşağıda verilmektedir.

6.7.1. Double Animation

Temel animasyon türlerinden biri olan DoubleAnimation, nesnelerin double türündeki özelliklerine uygulanan bir animasyon yöntemidir.

<DoubleAnimation> oluşturmak için aşağıdaki prensip ve özelliklerin bilinmesi gerekmektedir.

- Animasyon uygulanacak nesneye mutlaka bir isim verilmelidir (Örneğin, bu uygulamada "Resim1" ismi verilmiştir.).
- Width ve Height özellikleri değiştirileceğinden dolayı her iki özellik için ayrı ayrı <DoubleAnimation> oluşturulmuştur.
- Oluşturulan her iki <DoubleAnimation> <Storyboard> bloğu içinde olmalıdır.
- <Storyboard> bloğuna mutlaka bir isim verilmelidir. Çünkü animasyon programlama dili koduyla bu isim üzerinden başlatılacaktır.
- <Storyboard> bloğu <UserControl.Resources> bloğu içinde bulunmalıdır.
- Animasyon hazırlandıktan sonra program koduyla başlatılmalıdır. MainPage.xaml.cs içindeki MainPage() kurucu metodu içerisine Story1.Begin() gibi bir kod yazılarak animasyonun başlatılması sağlanmalıdır.

<DoubleAnimation> animasyonlarında kullanılan önemli özellikler aşağıda belirtilmiştir.

From: Nesnenin animasyon boyunca değişecek özelliğinin başlangıç değerini belirler.

To: Nesnenin animasyon boyunca değişecek özelliğinin bitiş değerini belirler.

By: Form ve To özelliklerinin değerleri arasındaki artış miktarını belirler.

RepeatBehavior: Animasyonun kaç defa tekrar edeceğini belirler. RepeatBehavior="Forever" tanımlanırsa animasyon devamlı tekrar edecektir. Belirli bir sayı kadar tekrar etmesini istersek RepeatBehavior="3x" şeklinde atama yapmanız gerekecektir.

AutoReverse: True atanarak animasyon bittikten sonra animasyonun geriye doğru çalışıp ilk hâline dönmesi sağlanır.

Duration: Animasyonun süresini belirlememizi sağlar. Saat, dakika, saniye şeklinde süre verilmesi gerekir. (Örneğin 1 saniyelik bir animasyon için 0:0:1 şeklinde kullanılmalıdır.

SpeedRatio: Animasyonun hızını belirler. Herhangi bir değer belirtilmez ise, 1 değerini alır. 10 saniyelik bir animasyonda SpeedRatio özelliği 5 atanmışsa animasyon 2 saniye sürecektir.

Storyboard.TargetName: Animasyonu oluşturulacak olan nesnenin adını tutar.

Storyboard.TargetProperty: Animasyonu oluşturulacak olan nesnenin animasyon boyunca değiştirilecek özelliğinin adını tutar.

Örnek: Sayfa üzerinde bulunan bir resmin genişlik ve yüksekliklerini eş zamanlı olarak değiştirerek resmi büyüten ve bu işlemi tekrarlayan animasyon uygulaması

```
<UserControl.Resources>
<Storyboard x:Name="Story1">
<DoubleAnimation Storyboard.TargetName="Resim1"
Storyboard.TargetProperty="Width" From="200" To="400" Duration="0:0:1"
AutoReverse="True" RepeatBehavior="Forever" />
<DoubleAnimation Storyboard.TargetName="Resim1"
Storyboard.TargetProperty="Height" From="200" To="400" Duration="0:0:1"
AutoReverse="True" RepeatBehavior="Forever" />
</Storyboard>
</UserControl.Resources>
<Canvas>
<Image x:Name="Resim1" Source="Penguins.jpg" Width="200" Height="200"
Canvas.Left="65" Canvas.Top="42" />
</Canvas>
```

MainPage sayfasının kod tarafında, MainPage() kurucu metodu içinde animasyonu başlatan kod aşağıdaki gibi yazılmalıdır.

```
public MainPage()
{
    InitializeComponent();
    Story1.Begin();
}
```

Ekran görüntüsü: Uygulama çalıştırıldığında resmin genişlik ve yükseklik değerlerinin büyüüp küçüldüğünü ve bu işlemin tekrar edileceğini göreceğiz.



Resim 6.6: DoubleAnimation uygulaması

6.7.2. ColorAnimation

<ColorAnimation> yöntemi renk geçiş animasyonları oluşturmak için kullanılır. ColorAnimation sınıfında From özelliği ile başlangıç rengi, To özelliği ile bitiş rengi belirlenir. AutoReverse özelliğine true atanarak, animasyon bittikten sonra animasyonun son renkten ilk renge dönmesi sağlanacaktır.

Örnek: Bir saniye süre içerisinde kırmızıdan maviye dönüşen elips animasyonu

```
<UserControl.Resources>
  <Storyboard x:Name="Story1">
    <ColorAnimation Storyboard.TargetName="solid1" Storyboard.TargetProperty="Color"
      From="Red" To="Blue" Duration="0:0:1" AutoReverse="True"
      RepeatBehavior="Forever" />
  </Storyboard>
</UserControl.Resources>
<Canvas>
  <Ellipse x:Name="elips1" Width="100" Height="100" Canvas.Left="65"
    Canvas.Top="42" >
    <Ellipse.Fill>
      <SolidColorBrush x:Name="solid1" Color="Red"/>
    </Ellipse.Fill>
  </Ellipse>
</Canvas>
```

Görüldüğü gibi başlangıçta Ellipse kontrolünü düz kırmızıya boyayan solid1 isimli fırça nesnesine renk animasyonu uygulanmıştır.

MainPage sayfasının kod tarafında, MainPage() kurucu metodu içinde animasyonu başlatan kod aşağıdaki gibi yazılmalıdır.

```
public MainPage()
{
    InitializeComponent();
    Story1.Begin();
}
```

Ekran görüntüsü:



Resim 6.7: ColorAnimation uygulaması

6.7.3. PointAnimation

XAML tabanlı İnternet uygulamalarında kullanılan kontrollerin point türündeki özelliklerini deęiřtiren animasyon türüdür.

Örnek: RadialGradientBrush fırçası ile dairesel renk geçiři kullanarak renklendirilmiş bir GradientOrigin noktasını PointAnimation yöntemiyle kaydıran uygulama

```
<UserControl.Resources>
<Storyboard x:Name="Story1">
<PointAnimation Storyboard.TargetName="ellipseBrush"
Storyboard.TargetProperty="GradientOrigin"
From="0.7,0.3" To="0.3,0.7" Duration="0:0:3" AutoReverse="True"
RepeatBehavior="Forever">
</PointAnimation>
</Storyboard>
</UserControl.Resources>
<Canvas>
<Ellipse x:Name="ellipse" Margin="5" Width="200" Height="200" >
<Ellipse.Fill>
<RadialGradientBrush x:Name="ellipseBrush"
RadiusX="1" RadiusY="1" GradientOrigin="0.7,0.3">
<GradientStop x:Name="ellipseBrushStop" Color="White"
Offset="0"></GradientStop>
<GradientStop Color="Blue" Offset="1"></GradientStop>
</RadialGradientBrush>
</Ellipse.Fill>
</Ellipse>
</Canvas>
```

Kodlardan da anlaşılacağı gibi GradientOrigin noktasını çapraz olarak kaydıracaktır.

MainPage sayfasının kod tarafında, MainPage() kurucu metodu içinde animasyonu başlatan kod aşağıdaki gibi yazılmalıdır.

```
public MainPage()
{
    InitializeComponent();
    Story1.Begin();
}
```

Ekran görüntüsü:



Resim 6.8: ColorAnimation uygulaması

6.8. Anahtar Çerçeveler (Key Frame)

Key Frame animasyonlar, bir zaman çizgisi üzerinde gerçekleşir. Key Frame denilen animasyondaki kilit durumlar zaman çizelgesinin belirli anlarında tanımlanır. Key Frame durumları arasındaki geçiş değerleri otomatik olarak hesaplanır.

Key Frame animasyonlar point, double, color ya da object türündeki özelliklere uygulanabilir. Key Frame animasyon teknikleri aşağıdaki gibidir.

- PointAnimationUsingKeyFrames
- ColorAnimationUsingKeyFrames
- DoubleAnimationUsingKeyFrames
- ObjectAnimationUsingKeyFrames

Key Frame animasyonlarda özellik tipine göre üç farklı geçiş metodu vardır. Bunlar; Linear, Spline, Discrete metodlarıdır.

Linear metodu: Bir cismin, bir özelliğinin, bir değerden bir değere ani olarak değil, kademeli olarak geçiş yaptığı animasyon değeridir. Örneğin, bir elips nesnesinin form üzerinde bir yerden bir yere kaydırılması bu yöntemle yapılmaktadır.

Discrete metodu: Nesne özelliklerinin anlık olarak değiştirildiği geçiş metodudur. İki KeyFrame arasında ara değerler olmaz.

Spline metodu: Nesne özelliklerinin değerleri bir değerden bir değere doğru değişirken belirli anlarda geçişin ivme kazanması ya da yavaşlatılması için kullanılır. Top zıplama animasyonları gibi animasyonlarda kullanılabilir.

6.8.1. <DoubleAnimationUsingKeyFrames> Etiketi

Nesnelerin Double türündeki özelliklerine uygulanacak olan keyframe animasyonları bu etiket bloğunda tanımlanır. Bu etikette kullanılan bazı özellikler ise şunlardır.

- **AutoReverse:** Animasyon bittikten sonra geri döndürmeyi sağlar.
- **Duration:** KeyFrame animasyonunun toplam süresini belirler.
- **RepeatBehavior:** Animasyonun tekrarlanma sayısını belirler.
- **Storyboard.TargetName:** Animasyonu oluşturulacak olan nesnenin adını tutar.
- **Storyboard.TargetProperty:** Animasyonu oluşturulacak olan nesnenin animasyon boyunca değiştirilecek özelliğinin adını tutar.

Her bir keyframe ise geçiş türüne göre <LinearDoubleKeyFrame>, <DiscreteDoubleKeyFrame> ya da <SplineDoubleKeyFrame> etiketlerinden biri ile oluşturulur. Bu etiketlerin herbirinin KeyTime ve Value isimli iki önemli özelliği vardır.

KeyTime özelliği anahtar karenin animasyon içindeki anını belirler. Value özelliği ise KeyTime'da belirtilen anda, TargetName içinde belirtilen nesnenin, TargetProperty içinde belirtilen özelliğine ait alacağı değeri belirler.

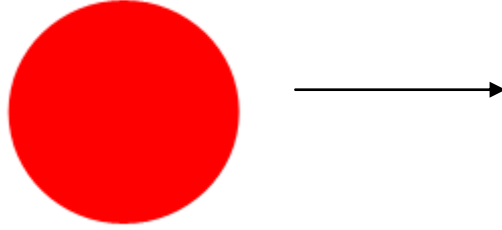
Örnek: <DoubleAnimationUsingKeyFrames> kullanımını aşağıdaki XAML kodunda görelim.

```
<UserControl.Resources>
  <Storyboard x:Name="Story1">
    <DoubleAnimationUsingKeyFrames AutoReverse="True" Duration="0:0:3"
RepeatBehavior="Forever" Storyboard.TargetName="kaydir"
Storyboard.TargetProperty="X">
      <LinearDoubleKeyFrame KeyTime="0:0:0" Value="50"/>
      <LinearDoubleKeyFrame KeyTime="0:0:3" Value="300"/>
    </DoubleAnimationUsingKeyFrames>
  </Storyboard>
</UserControl.Resources>
<Canvas>
<Ellipse x:Name="daire" Fill="Red" Margin="5" Width="115" Height="111"
Canvas.Left="60" Canvas.Top="57">
  <Ellipse.RenderTransform>
    <TranslateTransform x:Name="kaydir" X="0" Y="0"/>
  </Ellipse.RenderTransform>
</Ellipse>
</Canvas>
```

MainPage sayfasının kod tarafında, MainPage() kurucu metodu içinde animasyonu başlatan kod aşağıdaki gibi yazılmalıdır.

```
public MainPage()
{
    InitializeComponent();
    Story1.Begin();
}
```

Ekran görüntüsü:



Resim 6.9: DoubleAnimationUsingKeyFrames uygulaması

6.8.2. <ColorAnimationUsingKeyFrames> Etiketi

Nesnelerin Color(Renk) türündeki özelliklerine uygulanacak olan keyframe animasyonları bu etiket bloğunda tanımlanır. <ColorAnimationUsingKeyFrames> içinde geçerli olan özellikler bu etikette de geçerlidir.

Her bir keyframe ise geçiş türüne göre <LinearColorKeyFrame>, <DiscreteColorKeyFrame> ya da <SplineColorKeyFrame> etiketlerinden biri ile oluşturulur. Bu etiketlerin de herbirinin KeyTime ve Value özellikleri kullanılır.

Örnek:

```
<UserControl.Resources>
  <Storyboard x:Name="Story1">
    <ColorAnimationUsingKeyFrames AutoReverse="True" Duration="0:0:3"
    RepeatBehavior="Forever" Storyboard.TargetName="renk"
    Storyboard.TargetProperty="Color">
      <LinearColorKeyFrame KeyTime="0:0:0" Value="Red"/>
      <LinearColorKeyFrame KeyTime="0:0:3" Value="White"/>
    </ColorAnimationUsingKeyFrames>
  </Storyboard>
</UserControl.Resources>
<Canvas>
  <Ellipse x:Name="daire" Margin="5" Width="115" Height="111" Canvas.Left="60"
  Canvas.Top="57">
```

```
<Ellipse.Fill>
  <SolidColorBrush Color="Red" x:Name="renk"/>
</Ellipse.Fill>
</Ellipse>
</Canvas>
```

MainPage sayfasının kod tarafında, MainPage() kurucu metodu içinde animasyonu başlatan kod aşağıdaki gibi yazılmalıdır.

```
public MainPage()
{
    InitializeComponent();
    Story1.Begin();
}
```

Ekran görüntüsü: Kırmızıdan beyaza dönüşen bir daire görünecektir.

6.8.3. <PointAnimationUsingKeyFrames> Etiketi

Nesnelerin point türündeki özelliklerine uygulanacak olan keyframe animasyonları bu etiket bloğunda tanımlanır. <ColorAnimationUsingKeyFrames> ve <DoubleAnimationUsingKeyFrames> içinde geçerli olan temel özellikler bu etikette de geçerlidir.

Her bir keyframe ise geçiş türüne göre <LinearPointKeyFrame>, <DiscretePointKeyFrame> ya da <SplinePointKeyFrame> etiketlerinden biri ile oluşturulur. Bu etiketlerin de herbirinin KeyTime ve Value özellikleri kullanılır.

6.9. Animasyonların Kontrol Olayları İçinden Tetiklenmesi

Oluşturulan animasyonlar programlama dili kodları kullanılarak, çeşitli kontrollerin çeşitli olaylarına bağlı olarak tetiklenebilir. Örneğin, fare bir nesnenin üzerine getirildiğinde renk değiştirme animasyonunun çalışması ya da bir nesneye tıklandığında o nesnenin dönmesi gibi uygulamalar yapılırken programlama dili kodları kullanılacaktır.

Örnek: Fare bir elipsin üzerine getirildiğinde bir elipsin renk değiştirmesini sağlayan keyframe animasyonu yapalım.

XAML kodları aşağıdaki gibi olacaktır.

```
<UserControl.Resources>
  <Storyboard x:Name="story1">
    <ColorAnimationUsingKeyFrames AutoReverse="False" Duration="0:0:1"
    Storyboard.TargetName="renk" Storyboard.TargetProperty="Color">
      <LinearColorKeyFrame KeyTime="0:0:0" Value="Red"/>
    </ColorAnimationUsingKeyFrames>
  </Storyboard>
</UserControl.Resources>
```

```

        <LinearColorKeyFrame KeyTime="0:0:1" Value="Green"/>
    </ColorAnimationUsingKeyFrames>
</Storyboard>
<Storyboard x:Name="story2">
    <ColorAnimationUsingKeyFrames AutoReverse="False" Duration="0:0:1"
Storyboard.TargetName="renk" Storyboard.TargetProperty="Color">
        <LinearColorKeyFrame KeyTime="0:0:0" Value="Green"/>
        <LinearColorKeyFrame KeyTime="0:0:1" Value="Red"/>
    </ColorAnimationUsingKeyFrames>
</Storyboard>
</UserControl.Resources>
<Canvas>
<Ellipse Width="200" Height="100" MouseLeave="Ellipse_MouseLeave"
MouseEnter="Ellipse_MouseEnter">
    <Ellipse.Fill>
        <SolidColorBrush x:Name="renk" Color="Red"/>
    </Ellipse.Fill>
</Ellipse>
</Canvas>

```

Görüldüğü gibi story1 ve story2 isminde iki adet StoryBoard tanımlanmıştır. Story1 içinde kırmızıdan yeşile geçiş animasyonu keyframe yöntemi ile oluşturulmuştur. Story2 içinde de yeşilden kırmızıya geçiş animasyonu oluşturulmuştur.

Oluşturulan her iki animasyonun elips üzerindeki fare hareketine göre başlatılması gerekmektedir. Yani elipsin üzerine fare getirildiğinde(MouseEnter olayında) story1 isimli animasyonun çalışması gerekiyor. Aynı şekilde fare elipsin üzerinden ayrıldığına(MouseLeave olayında) story2 isimli animasyonun çalışması gerekiyor. Bunun için MainPage sayfasının kod tarafında elips nesnesinin MouseEnter ve MouseLeave metodları içinde animasyonları başlatan kodlar aşağıdaki gibi yazılmalıdır.

```


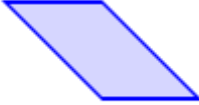
private void Ellipse_MouseLeave(object sender, MouseEventArgs e)
{
    story2.Begin();
}

private void Ellipse_MouseEnter(object sender, MouseEventArgs e)
{
    story1.Begin();
}

```


UYGULAMA FAALİYETİ

Aşağıdaki önerilere göre işlem basamaklarını uygulayınız.

İşlem Basamakları	Öneriler
<p>➤ Döndürme işlemini gerçekleştirerek aşağıdaki tasarımı gerçekleştiriniz.</p> 	<p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Canvas Height="200" Width="200"> <Polyline Points="25,25 0,50 25,75 50,50 25,25 25,0" Stroke="Blue" StrokeThickness="10" Canvas.Left="75" Canvas.Top="50"> <Polyline.RenderTransform> <RotateTransform CenterX="0" CenterY="0" Angle="45" /> </Polyline.RenderTransform> </Polyline> </Canvas></pre>
<p>➤ Ölçülendirme (Scale) işlemini gerçekleştiriniz.</p>	<p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Rectangle Height="50" Width="50" Fill="#CCCCCCFF" Stroke="Blue" StrokeThickness="2" Canvas.Left="100" Canvas.Top="100"> <Rectangle.RenderTransform> <ScaleTransform CenterX="0" CenterY="0" ScaleX="2" ScaleY="2" /> </Rectangle.RenderTransform> </Rectangle></pre>
<p>➤ Öteleme işlemini gerçekleştiriniz.</p>	<p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Rectangle Height="50" Width="50" Fill="#CCCCCCFF" Stroke="Blue" StrokeThickness="2" Canvas.Left="100" Canvas.Top="100"> <Rectangle.RenderTransform> <TranslateTransform X="50" Y="50" /> </Rectangle.RenderTransform> </Rectangle></pre>
<p>➤ Çarpıtma (Skew) işlemini gerçekleştirerek aşağıdaki görüntüyü elde ediniz.</p> 	<p>➤ Aşağıdaki XAML kodunu yazınız.</p> <pre><Rectangle Height="50" Width="50" Fill="#CCCCCCFF" Stroke="Blue" StrokeThickness="2" Canvas.Left="100" Canvas.Top="100"> <Rectangle.RenderTransform> <SkewTransform CenterX="0" CenterY="0" AngleX="45" AngleY="0" /> </Rectangle.RenderTransform> </Rectangle></pre>

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Döndürme işlemi gerçekleştirdiniz mi?		
2. Ölçülendirme işlemi gerçekleştirdiniz mi?		
3. Dönüştürme işlemi gerçekleştirdiniz mi?		
4. Çarpıtma işlemi gerçekleştirdiniz mi?		
5. Matris dönüşümlerini gerçekleştirdiniz mi?		
6. Animasyonların program kodlarıyla kontrol edilmesini gerçekleştirdiniz mi?		
7. Animasyon parametrelerini düzenlediniz mi?		
8. Anahtar çerçevelerle ilgili ihtiyaç duyulan düzenlemeleri gerçekleştirdiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda “Hayır” şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız “Evet” ise “Ölçme ve Değerlendirme”ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi sol üst köşe referans alınarak dönme hareketi gerçekleştirir?
A) Rotate Transform
B) Skew Transform
C) Translate Transform
D) Matrix Transform
2. Aşağıdakilerden hangisi yatay ve dikey eksenlerdeki büyüklükleri değiştirir?
A) Rotate Transform
B) Scale Transform
C) Translate Transform
D) Matrix Transform
3. Aşağıdakilerden hangisi öteleme işlemini gerçekleştirir?
A) Rotate Transform
B) Scale Transform
C) Translate Transform
D) Matrix Transform
4. Aşağıdakilerden hangisi ile nesne X ve Y eksenlerinde tanımlanan açı ile eğilebilir?
A) Rotate Transform
B) Skew Transform
C) Translate Transform
D) Matrix Transform
5. Aşağıdakilerden hangisi resim işleme yöntemlerinde kullanılmaktadır?
A) Rotate Transform
B) Skew Transform
C) Translate Transform
D) Matrix Transform

Aşağıdaki cümleleri dikkatlice okuyarak boş bırakılan yerlere doğru sözcüğü yazınız.

6. Kontrollerin panel içerisindeki bulunuş açılarını değiştirerek rotasyona uğratma işlemlerini.....denir.
7. XAML ile oluşturulan animasyonların ne zaman başlayacağını ya da ne zaman biteceği Özelliği aracılığıyla belirlenir.
8. Temel animasyon türlerinden biri olansınıfı, nesnelerin double türündeki özelliklerine uygulanmaktadır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise “Modül Değerlendirme”ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki cümleleri dikkatlice okuyarak boş bırakılan yerlere doğru sözcüğü yazınız.

1. panelinin içerisinde bulunan kontrol noktasal konumlandırılabilir.
2. XAML’de ilk tanımlanan eleman, eleman olarak adlandırılır.
3. bir root elemandır.
4., bir zaman çizgisi üzerinde gerçekleşir.
5.renklerle ilgili animasyonları oluşturmayı sağlar.
6., animasyon üzerinde mevcut konumu almayı sağlar.
7.animasyonun hızını belirler.
8.TimeSpan türünde bir zaman alır ve animasyonun bu zamana atlamasını sağlar.
9. metodu bir StoryBoard animasyonunu başlatmayı sağlar.

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

10. Aşağıdakilerden hangisi içinde yer alan nesnelere çerçeve ekler?
A) Border B) Tuval (Canvas) C) Stackpanel D) Grid
11. Satır ve sütunları bulunabilen yerleşim paneli hangisidir?
A) Stackpanel B) Tuval (Canvas) C) Border D) Grid
12. Aşağıdakilerden hangisi Key Frame animasyonlarda özellik tipine göre üç farklı interpolasyon metodlarından **değildir**?
A) Linear B) Spline C) Double D) Discrete

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki modüle geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	C
2	A
3	C
4	D
5	C
6	Border
7	Height

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

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

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	96
2	37.7
3	1.333
4	B
5	C
6	A
7	Doğru
8	Yanlış
9	Doğru

ÖĞRENME FAALİYETİ-4'ÜN CEVAP ANAHTARI

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

ÖĞRENME FAALİYETİ-5'İN CEVAP ANAHTARI

1	D
2	C
3	Source
4	FontFamily
5	FontUri

ÖĞRENME FAALİYETİ-6'NİN CEVAP ANAHTARI

1	A
2	B
3	C
4	B
5	D
6	RotateTransform
7	Triggerlar
8	Double Animation

MODÜL DEĞERLENDİRME'NİN SINAVI CEVAP ANAHTARI

1	Canvas
2	Root
3	UserControl
4	Key Frame Animasyonlar
5	Color Animation
6	Get CurrentState
7	SpeedRatio
8	Seek
9	Begin
10	A
11	D
12	C

KAYNAKÇA

- MORONEY Laurence, **Beginning Web Development, Silverlight, and ASP.NET Ajax:From Notice to Professional**, Apress, 2008.
- ÖZDEMİR Selçuk, **Framework 4.0**, Kodlab Yayıncılık, İstanbul, 2010.
- SÜZEN Ahmet Ali, **Silverlight**, Kodlab Yayıncılık, İstanbul, 2011.