WPF 그래픽

  - 기존과 다르게 다시 그리기를 할 필요가 없음

  - 사용자 인터페이스의 모든 요소와 통합 할 수 있음

  - 해상도에 독립적인 UI(확대해도 지정된 크기에서 최대한 잘 보이도록 렌더링 함)

  - 장치 독립적인 픽셀을 1/96 인치로 정의하여 사용함

  - 크게 도형, 브러시, 펜으로 분류할 수 있음

  - 모든 도형의 테두리에 안티-알리아싱을 지원하고 투명도도 지원함

  - 최신 그래픽 카드의 기능을 사용하며 내부적으로는 Direct 3D 기반으로 구현되어 있음

도형

  - System.Windows.Shapes 네임스페이스 안에 기본적인 드로잉 객체들이 정의되어 있음

  - Syetem.Windows.Media 네임스페이스 안에 Geometry 클래스 버전의 객체들이 정의되어 있음

  - FIll 속성과 Stroke 속성을 지정하지 않으면 기본적으로 null 이므로 도형이 보이지 않음

  - Stretch 속성은 도형이 필요로 하는 공간이 충분하지 못할 경우 어떻게 조정할지를 결정함

  - Stretch 속성은 Fill , None , Uniform , UniformToFill 의 값을 가짐

  - Rectangle 객체

    - Rectangle 객체는 속이 채워진 형태, 외곽선만 가진 형태, 두 가지를 합친 형태로 그릴수 있음

    - 부모 패널이 제공하는 좌표시스템을 통해 정렬됨

    - Rectangel 객체가 부모 패널과 동일하게 회전하려면 RenderTransform 속성을 사용함(p508)

       <Rectangle.RenderTransform>

         <RotateTransform Angle="135" />

       </Rectangle.RenderTransform>

    - 모서리가 둥근 사각형을 만드려면 RadiusX="값" RadiusY="값" 속성을 사용함

  - Ellipse 객체

    - Rectangle 객체와 유사하며 타원을 그림

  - Line 객체

    - 한점에서 다른점으로 직선을 그리며 X1, X2, Y1, Y2 를 사용함

    - Line 요소가 레이아웃에 따라 자동적으로 조정되도록 하려면 Stretch 속성을 이용함

  - Polyline 객체

    - 직선을 연결하여 그릴 수 있으며 Points 속성을 통해 리스트를 저장함

    - 항상 열린 도형을 만들어 냄

  - Polygon 객체

    - Polyline 객체와 비슷하며 항상 닫힌 도형을 만들어 냄

    - FillRule 속성을 통해 내부로 취급하기 모호한 영역을 처리하는 방법을 지정할 수 있음

    - FillRule 속성에는 EvenOdd(기본값) 와 Nonzero 값이 있음

  - Path 객체

    - 도형을 그릴 수 있는 강력한 객체로 앞의 모든 도형을 Path 객체로 그릴 수 있음

    - 앞의 도형들은 Geometry 객체를 기반으로 한 래퍼들임

    - Path 객체는 Geometry 를 직접 사용하며 GeometryGroup 라는 특별한 객체를  사용할 수 있음

    - Grometry 종류

|  |  |
| --- | --- |
| 타입 | 설명 |
| CombinedGeometry | 두 개의 Geometry 객체를 교집합 또는 합집합   개념으로 합침 |
| EllipseGeometry | 타원을 표현함 |
| GeometryGroup | 복수의 Geometry를 하나의 Geometry로 취급 |
| LineGeometry | 직선을 표현함 |
| PathGeometry | 직선과 타원호, 베지어 곡선 등을 조합하여   도형을 정의함 |
| RectangleGeometry | 사각형을 표현함 |
| StreamGeometry | PathGeometry를 효과적으로 변형한 형태로   모든 도형 정의 가능(생성 후 변경 불가능) |

- GeometryGroup과 CombinedGeometry의 차이점은 GeometryGroup은 도형을 따로 합쳐서

       2가지의 모형을 갖는 반면 CombinedGeometry은 도형을 하나의 도형으로 만들어 4가지의

       모형을 가질 수 있어 모드에 GeometryCombineMode 속성을 통해 결정됨

    - Path 객체 사용법

      <Path.Data>

        //<RectangleGeometry Rect="0, 0, 40, 80" />

        <GeometryGroup>

          <EllipseGeometry Center="20, 40" RadiusX="20" RadiusY="40" />

          <EllipseGeometry Center="20, 40" RadiusX="10" RadiusY="30" />

        </GeometryGroup>

      <Path.Data>

  - PathGeometry 는 PathFigure 속성을 가지고 있으며 하나의 모형을 나타냄(복수 가능)

  - PathFigure 객체의 시작점은 StartPoint 속성이며 하나 이상의 Segment 객체를 사용하여 정의함

  - 닫힌 도형의 여부는 IsClosed 속성에 의해 결정됨

- PathFigure 의 Segment 타입

|  |  |
| --- | --- |
| Segment | 설명 |
| LineSegment | 하나의 직선 |
| PolyLineSegment | 여러 개의 직선 |
| ArcSegment | 타원 호 |
| BezierSegment | 3차 베지어 곡선 |
| QuadraticBezierSegment | 2차 베지어 곡선 |
| PolyBezierSegment | 연속된 3차 베지어 곡선 |
| PolyQuadraticBezierSegment | 연속된 2차 베지어 곡선 |

  - ArcSegment 요소는 SweepDirection="Clockwise , Counterclockwise" 값을 가질 수 있으며

    방향을 나타내며 Clockwise는 시계방향, Counterclockwise는 시계반대방향을 나타냄

  - ArcSegment 요소는 IsLargeArc="False, True" 값을 가질 수 있으며 원의 큰 부분인지 나나탬

  - BezierSegment 예제

    <PathFingure StartPoint="0, 50">

      <BezierSegment Point1="60,50" Point2="100, 0" Point3="100, 50" />

    </PathFingure>

    //StartPoint 가 선의 시작점이고 Point3 이 선의 끝점임, Point1, Point2 가 제어점을 나타냄

  - Path.Data 명령들

    명령 (명령 이름) [Segment 종류] : 매개 변수

    M (Move) : 새로운 PathFigure 의  StartPoint 가 됨

    L (Line) [LineSegment] : 끝점을 위한 좌표와 짝

    H (Horizontal Line) [LineSegment] : x좌표를 위한 하나의 숫자(y좌표는 이전과 동일값 사용)

    V (Vertical Line)  [LineSegment] : y좌표를 위한 하나의 숫자(x좌표는 이전과 동일값 사용)

    C (Cubic Bezier Curve) [BezierSegment] : 세 개의 좌표 값(제어점 2개, 끝점 1개)

    Q (Quadratic Bezier Curve) [QuadraticBezierSegment] : 두 개의 좌표 값(제어점 1개, 끝점 1개)

    S (Smooth Bezier Curve) [BezierSegment] : 두번째 제어점과 끝점을 위한 두개의 좌표값

    T (Smooth Quadratic Bezier Curve) [QuadraticBezierSegment] : 끝점을 위한 하나의 좌표값

    A (Ellipse Arc) [ArcSegment] : x 반지름, y 반지름, RotationAngle, IsLarge, SweepDirection,

                                                그리고 양 끝점을 위한 7개의 좌표

    Z (Close Path) : 없음

    F0 (EvenOdd 채우기 규칙) : 없음

    F1 (Nonzero 채우기 규칙) : 없음

비트맵

  - System.Windows.Media.Imaging 네임스페이는 이미지와 관련된 클래스가 있음

  - Image 요소는 단순히 이미지를 표시하는 역활을 담당함

  - 이미지를 사용 가능한 영역의 크기에 맞추어 표시하고 싶으면 Image 요소의 Stretch 속성을 지정

  - 간단한 이미지 사용법

    <Image Source="<http://www.xxx.co.kr/x.jpg>" />

    <Image Source="/MyPicture.jpg" />

  - ImageSource 타입은 추상 클래스이며 이미지를 표현하는 모든 클래스의 기반 클래스임

  - ImageSource 타입에서 파생되는 클래스는 DrawingImage 와 BitmapSource 두 가지임

  - DrawingImage 클래스는 비트맵에 대해 아무 작업도 수행하지 않으며 단지 해상도에 독립적인

    드로잉 객체에 대한 래퍼로서의 역활을 담당함

  - RenderTargetBitmap 객체를 이용하면 모든 비주얼 요소를 바탕으로 새로운 비트맵을 만듬

    RenderTargetBitmap bmp = new RenderTargetBitmap(

                                          300, 150,    //픽셀 기반의 물리적 크기

                                          300, 300,    //픽셀 해상도(dpi)

                                          PixelFormats.Pbgra32);

      .....  bmp.Render(e);

  - RenderTargetBitmap 주의사항

    1. 비주얼 요소가 UI 내에서 가시화되지 않았다면 Measure 메소드와 Arrange 메소드를 직접

       호출해 주어야 함

    2. 부모가 없는 독립적인 비주얼 요소를 생성하면 컨트롤들은 자신의 기본 스타일을 가져오지

       못하므로 Render 메소드에 전달하기 전에 UI들이 실제 윈도우 객체에서 보여지도록 해야함

  - RenderTargetBitmap 은 비트맵을 만들거나 수정할 때 유용함

  - WriteableBitmap 은 픽셀의 원본 데이터를 이용할 때 유용함

비트맵 인코더와 디코더

  - 비트맵 인코더는 특정 포맷으로 비트맵 스트림을 생성하는 방법을 알고 있는 클래스임

  - PngBitmapEncoder, JpegBitmapEncoder 등과 같이 지원하는 포맷을 덧붙인 이름을 사용함

  - 비트맵을 JPEG 파일로 디스크에 저장하는 예제

    static void WriteJpeg(string fileName, int quality, BitmapSource bmp) {

      JpegBitmapEncoder encoder = new JpegBitmapEncoder();

      BitmapFrame outputFrame = BitmapFrame.Create(bmp);

      encoder.Frames.Add(outputFrame);

      encoder.QualityLevel = quality;

      using (FileStream file = File.OpenWrite(fileName))

        encoder.Save(file);

    }

  - 비트맵 디코더는 특정 포맷의 비트맵 스트림을 읽는 방법을 알고 있는 클래스임

  - 비트맵의 메타데이터에 엑세스하거나 다중 프레임 이미지 파일의 모든 프레임을 조회할 때 사용

  - 디코더를 이용해 JPEG 이미지를 로드하는 예제

    static string GetCamera(string myJpegPath) {

      JpegBitmapDecoder decoder = new JpegBitmapDecoder(new Uri(myJpegPath),

                      BitmapCreateOptions.None, BitmapCacheOption.None);

      BitmapMetadata bmpData = (BitmapMetadata) decoder.Frames[0].Meradata;

      return bmpData.CameraModel;

    }

비트맵 효과

  - 모든 사용자 인터페이스 요소들은 BitmapEffects 속성을 가지고 있음

  - 비트맵 효과들

|  |  |
| --- | --- |
| 타입 | 설명 |
| BevelBitmapEffect | 콘텐츠에 3차원 양각 테두리 효과 |
| BitmapEffectGroup | 하나의 요소에 다중 효과 |
| BlurBitmapEffect | 이미지의 포커스가 흐릿한 효과 |
| DropShadowBitmapEffect | 콘텐츠의 테두리에 부드러운 그림자 효과 |
| EmbossBitmapEffect | 콘텐츠 전체에 3D 양각 효과(범프매핑알고리즘) |
| OuterGlowBitmapEffect | 콘텐츠 주변에 부드러운 후광 효과 |

- BlurBitmapEffect 효과 예제

    <StackPanel.BitmapEffect>

      <BlurBitmapEffect Radius="1" />

    </StackPanel.BitmapEffect>

색상

  - System.Windows.Media 네임스페이스에 구현된 Color 클래스를 이용하여 색상을 표현함

  - wpf 색상은 A(투명도), R(레드), G(그린), B(블루) 4채널을 사용함

  - A(투명도) 의 값이 '0' 이면 완전히 투명함을 의미하고 '1' 이면 완전히 불투명함을 의미함

  - 8비트 색상 채널을 사용하려면 A, R, G, B 4가지 속성을 사용해야 함

  - 16비트 색상 채널을 사용하려면 ScA, ScR, ScG, ScB 4가지 속성을 사용해야 함

  - Color 클래스도 범위를 벗어나는 색상 값을 허용함(이미지 프로세싱 후 결과만 범위에 있으면 됨)

  - Colors 클래스도 제공하며 표준 색상이름과 예전에 자주 사용한 색상 이름을 제공함

브러시 종류

  SolidColorBrush

  - SolidColorBrush 는 가장 간단한 브러시로 전체 영역에 걸쳐 하나의 색상을 칠할 때 사용함

  - XAML 에서 색상의 이름을 지정하면 자동으로 SolidColorBrush 를 만들어 사용함

  - Brushes 클래스가 존재하기 때문에 별도의 브러시를 생성할 필요가 없음

  - 마크업에서 숫자로 된 색상 코드를 사용해도 SolidColorBrush 를 얻을 수 있음(p546)

  LinearGradientBrush

  - 영역에 칠해진 색상을 다른 색상으로 변환하거나 순차적으로 색상을 변화시킬 수 있음

  - 페이드 효과는 직선으로만 만들어지며 곡선을 이용한 효과는 만들 수 없음

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

    <GradientStop Color="Black" Offset="0" />

    <GradientStop Color="White" Offset="1" />

  </LinearGradientBrush>

  // StartPoint 와 EndPoint 속성은 색상의 변환이 어디에서 시작해서 어디에서 끝나는지 지정함

  RadialGradientBrush

  - LinearGradientBrush 와는 달리 시작점에서 원을 그려가며 색상의 변환을 수행함

  <RadialGradientBrush Center="0.45, 0.5" RadiusX="0.3" RadiusY="0.5"

                                  GradientOrigin="0.25, 0.4">

    <GradientStop Color="Black" Offset="0" />

    <GradientStop Color="White" Offset="1" />

  </RadialGradientBrush>

  // RadiusX , RadiusY 속성을 이용하여 색을 채우기 위한 원의 크기를 결정함

  // Center 속성을 통해 원의 위치를 결정함

  // GradientOrigin 속성을 통해 원의 초점을 결정함

  ImageBrush , DrawingBrush , VisualBrush

  - ImageBrush 는 비트맵을 이용하며 DrawingBrush 는 스케일 조정이 가능한 드로잉 객체를 이용

    하며 VisualBrush 는 UI 요소들을 브러시 이미지로 사용함

  - TileBrush 기반 추상 클래스로부터 파생되며 TileBrush 클래스는 이미지를 어떻게 늘이거나

    줄일지 , 이미지가 반복될지 , 도형 내에 이미지의 위치는 어디인지를 결정함

  - Stretch(채우기) , AlignmentX , AlignmentY(정렬) , Viewbox(특정 부분 초점) , ViewboxUnits

    (초점 모드) , Viewport(나타낼 초점 위치) , Tilemode(남은 공간 반복 채우기) 이 있음

  - ImageBrush 는 스크린의 정해진 영역에 비트맵으로 채움(해상도에 독립적이지 못함)

    <ImageBrush ImageSource="Images\aa.jpg" />

  - DrawingBrush 는 스케일 조정이 가능한 벡터 이미지를 사용하여 해상도에 독립적임

  - 벡터 이미지는 GeometryDrawing 클래스와 ImageDrawing 과 VideoDrawing 클래스를 통해

    지원되며 DrawingGroup 객체를 이용하여 그룹화 할 수 있음

    <DrawingBrush>

      <DrawingBrush.Drawing>

        <DrawingGroup>

          <DrawingGroup.Children>

            <GeometryDrawing>

              <GeometryDrawing.Brush>

                <LinearGradientBrush ..... >

                  .....

                </LinearGradientBrush>

              </GeometryDrawing.Brush>

            </GeometryDrawing>

          </DrawingGroup.Children>

        </DrawingGroup>

      </DrawingBrush.Drawing>

    </DrawingBrush>

  - VisualBrush 는 Visual 클래스로부터 파생된 모든 요소들을 콘텐츠로 활용하여 그릴수 있음

    <VisualBrush>

      <VisualBrush.Visual>

        <Grid .....>

          <Rectangle ..... />

        </Grid>

      </VisualBrush.Visual>

    </VisualBrush>

Pen

  - 도형의 외곽선을 그릴 때 사용함

  - Thickness 속성을 선의 굵기를 Brush 속성을 선의 색상을 지정함

  - StrokeDashArray ="선길이, 공백, 선길이, 공백" 속성을 이용하여 선의 패턴을 지정할 수 있음

  - StrokeThickness 속성은 패턴의 굵기를 Stroke 속성은 패턴의 색상을 지정함

  - LineJoin 속성으로 모서리를 그릴 수 있음 Miter(직각) Bevel(잘린) Round(둥근)

  - StartLineCap 과 EndLineCap 속성을 이용하여 선의 시작과 끝 모양을 지정할 수 있음

    Round, Triangle, Flat(모서리가 교차), Square(선의 끝이 경계를 넘어 확장) 값이 있음

Transform 타입들

|  |  |
| --- | --- |
| Transform 클래스 | 설명 |
| MatrixTransform | 3 X 3 매트릭스에 기반한 범용의 변형 기능 |
| RotateTransform | 점을 기준으로 회전하는 기능 |
| ScaleTransform | X 또는 Y 좌표를 기준으로 스케일 변형 기능 |
| SkewTransform | 찌그러뜨리는 기능(정사각형 -> 마름모) |
| TransformGroup | 여러 변형 클래스들을 하나로 조합하는 기능 |
| TranslateTransform | 지전된 벡터로 아이템을 이동하는 기능 |