

그래픽스와 애니메이션



시작하면서



◎ 목차

- 이미지 출력
- 커스텀 뷰와 캔버스
- 셰이프드로어블
- 그래디언트
- 애니메이션

이미지 출력



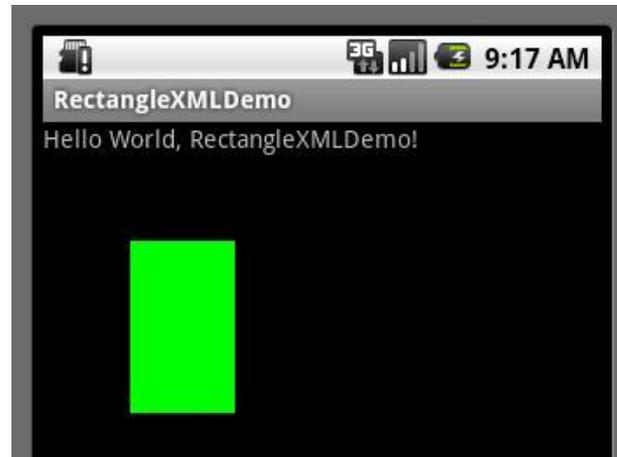
- 뷰 객체에 이미지 또는 그래픽들을 출력
- ◎ **이미지 출력**
 - res/drawable에 이미지 저장
 - icon.png
 - `<ImageView android:src="@drawable/icon" />`
 - PNG, JPG, BMP, GIF 형식 지원
 - 간단한 게임, 성능이 중요하지 않은 응용
- ◎ **<실습 7-1> 두 개의 이미지 출력**
 - AndroidIconDemo 프로젝트 생성
 - <코드 7-1>을 main.xml에 복사
 - res/drawable-mdpi/ 폴더에 icon_contribute.jpg,
 - res/drawable-hdpi/ 폴더에 icon_super.png를 저장



그래픽 출력



- 리소스 파일로 그래픽을 정의하여 출력
 - main.xml: ImageView 영역 설정
 - rect.xml: 직사각형을 정의
- ◎ **<실습 7-2> 그래픽 출력**
 - `imageView.setImageResource(R.drawable.rect);`
- RectangleXMLDemo 프로젝트 생성
 - <코드 7-2>을 main.xml에 복사
 - <코드 7-3>을 res/drawable/ 폴더의 rect.xml에 복사
 - <코드 7-4>를 RectangleXMLDemo.java에 복사



커스텀 뷰



◎ 절차

- View 클래스를 상속하여 커스텀 뷰를 생성
- onDraw() 콜백 메소드 사용
 - 캔버스, Paint 클래스들을 사용하여 이 메소드에서 그림

◎ 캔버스 클래스

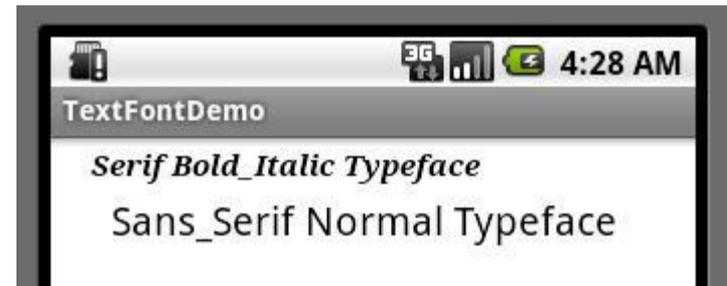
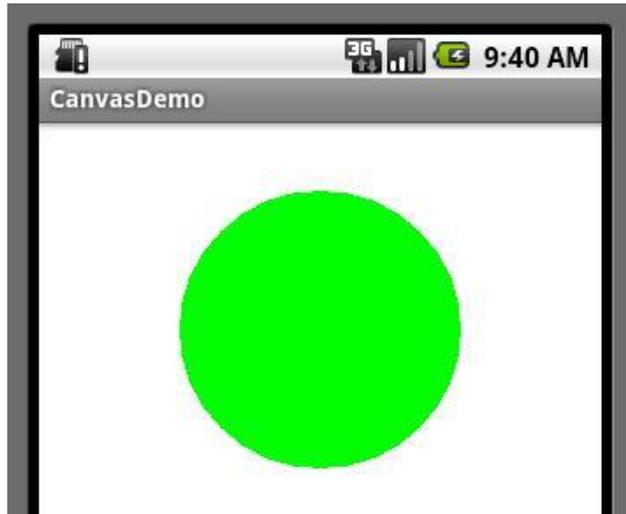
- '화폭'을 의미
- 그림을 그리는 실제 화폭에 대응
 - drawColor(Color.WHITE): 흰색으로 캔버스를 칠함(배경을 흰색으로 설정)
 - drawCircle(), drawText() 등으로 그래픽 도형 등을 그림

◎ Paint 클래스

- '물감'에 대응
- 문자 크기, 색, 스타일, 글꼴, 렌더링 정보를 보관
 - setTextSize(15), setTypeface(Typeface.SERIF, Typeface.BOLD_ITALIC)



- ◎ <실습 7-3> 색이 채워진 원 그리기
 - CanvasDemo 프로젝트 생성
 - <코드 7-5>를 CanvasDemo.java에 복사
- ◎ <실습 7-4> 글자 크기, 글꼴 설정
 - TextFontDemo 프로젝트 생성
 - <코드 7-6>을 TextFontDemo.java에 복사





- 사실 Canvas에서 그리는 내용들이 Bitmap 객체로 표현되어 화면으로 출력
- 이중 버퍼링
 - 하위 Bitmap 객체 위에 먼저 그려진 다음, 완성된 Bitmap을 화면으로 출력하는 방식
 - 화면의 깜박거리는 현상 제거
 - 안드로이드가 처리
 - 예)
 - `Bitmap pic1 = BitmapFactory.decodeResource(..., R.drawable.flowerdew);`
 - `canvas.drawBitmap(pic1);`
 - `Bitmap pic1 = Bitmap.createScaleBitmap(pic1, ...);`



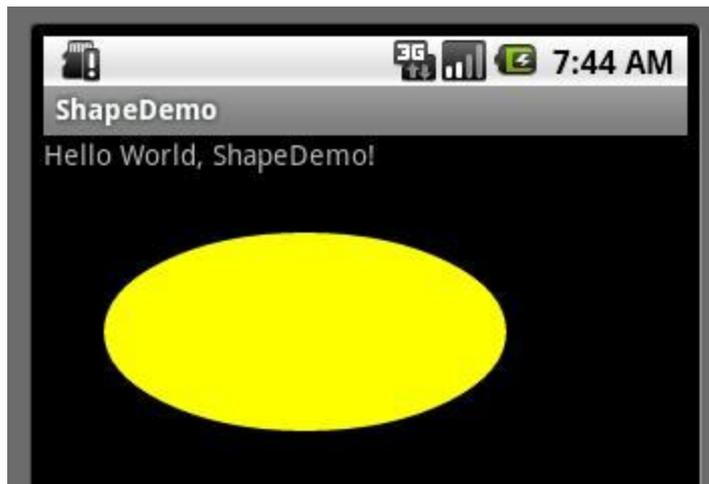
- ◎ <실습 7-5> 비트맵 그리기
 - BitmapDemo 프로젝트 생성
 - <코드 7-8>을 main.xml에 복사
 - <코드 7-9>을 ShapeDemo.java 에 복사



ShapeDrawable



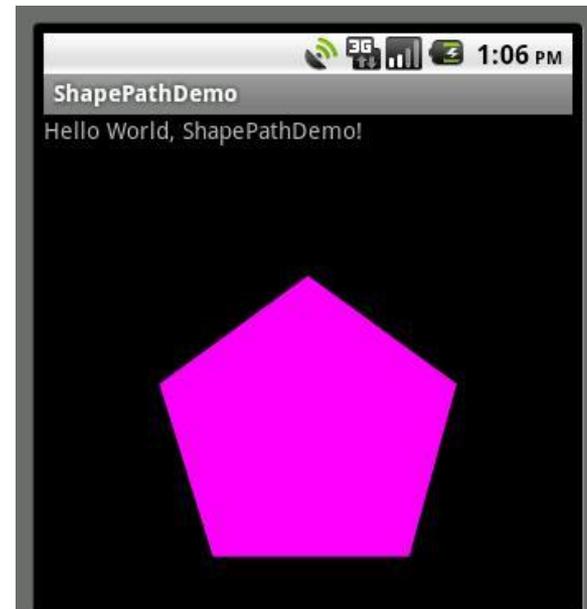
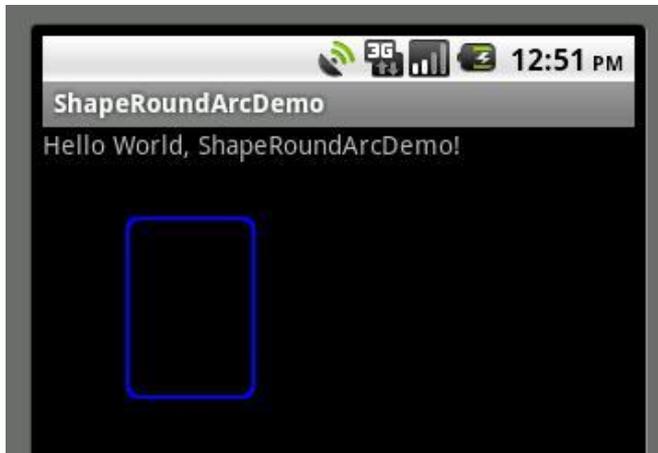
- 동적으로 2차원 도형을 그리는데 적합
- Shape 클래스를 이용하여 도형을 그림
 - PathShape, RectShape
 - ArcShape, OvalShape, RoundRectShape
 - ShapeDrawable oval = new ShapeDrawable(new OvalShape());
 - imageView.setImageDrawable(oval);
- Drawable의 파생 클래스: <표 7-1>
- ◎ **<실습 7-6> 타원 그리기(노란색)**
- ShapeDemo 프로젝트 생성
 - <코드 7-8>을 main.xml에 복사
 - <코드 7-9>를 ShapeDemo.java에 복사



ShapeDrawable



- ◎ **<실습 7-7> 둥근 모서리 직사각형 그리기**
 - ShapeRoundArcDemo 프로젝트 생성
 - <코드 7-10>을 main.xml에 복사
 - <코드 7-11>를 ShapeRoundArcDemo.java에 복사
- ◎ **<실습 7-8> 경로 그리기**
 - ShapePathDemo 프로젝트 생성
 - <코드 7-12>을 main.xml에 복사
 - <코드 7-13>을 ShapePathDemo.java에 복사

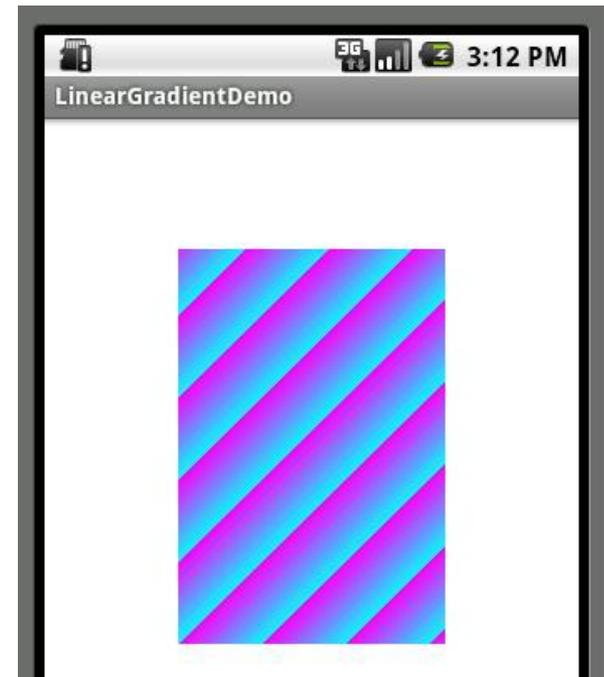




- 설정한 색들을 부드럽게 연결하여 색의 변화를 자연스럽게 표현하는 시각적 효과를 추구하는 기법
- 종류
 - 선형 그래디언트(Linear Gradient), 방사형 그래디언트(Radial Gradient), Sweep Gradient

◎ <실습 7-9> 선형 그래디언트

- LinearGradientDemo 프로젝트 생성
 - <코드 7-14>를 LinearGradientDemo.java에 복사



애니메이션

프레임 Animation



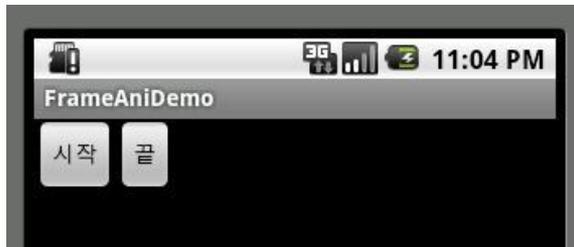
- 전통적인 애니메이션
- 프레임, 배경 또는 키 프레임 애니메이션이라 불림
- 다수의 이미지를 연속적으로 제시하여 움직이는 효과
- 눈의 잔상 효과에 의해 자연스러운 움직임으로 인식:
 - 16 프레임/초 이상
- 특수한 형태: AnimatedGIF
 - 여러 이미지들을 하나의 GIF 형식으로 묶음
- AnimationDrawable: 프레임 애니메이션의 기본 클래스

프레임 Animation



◎ <실습 7-10> 프레임 애니메이션 실습

- 달이 움직이는 모습: 프레임 moon01.png~12.png
- FrameAniDemo 프로젝트 생성
 - <코드 7-16>를 res/layout 폴더의 frame.xml에 복사
 - res/drawable-hdpi/ 폴더에 이미지 moon01.png~moon12.png를 복사
 - <코드 7-17>~<코드 7-18>을 FrameAniDemo.java에 복사



Tween Animation



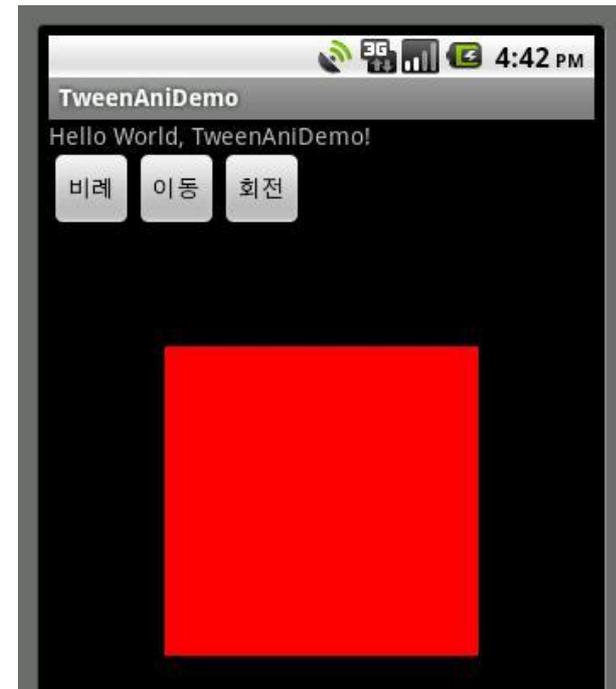
- 그래픽스 또는 임의의 뷰에 대한 변환으로 움직임 효과
- 변환 종류: 이동 변환(translate), 비례 변환(scale), 회전 변환(rotate), 투명 변환(alpha)
- Transition 애니메이션이라고도 불림
- 변환 속도 조절 속성
 - interpolator 3종류: <표 7-3>
- 애니메이션용 XML 파일은 res/anim에 위치
 - <set>
 - <scale ... />
 - <set>
 - <rotate ... />
 - </set>
 - </set>

Tween Animation



◎ <실습 7-11> 트윈 애니메이션 실습

- 빨간색 사각형의 비례, 이동, 회전 변환
- TweenAni 프로젝트 생성
 - res/layout 폴더에 main.xml을 생성, <코드 7-23>을 res/anim/ 폴더의 tween.xml에 복사
 - <코드 7-24>~<코드 7-26>을 res/anim/ 폴더의 translate.xml, scale.xml, rotate.xml에 복사
 - <코드 7-20>~<코드 7-22>를 TweenAni.java에 복사



Tween Animation

