E05 Drawable 예제

2014-11-25

이승진

**목차**

[1. E05Drawable 프로젝트 2](#_Toc404915192)

[1) 프로젝트 생성 2](#_Toc404915193)

[2. FrameAnimationActivity 액티비티 생성 3](#_Toc404915194)

[1) 프로젝트에 FrameAnimationActivity 액티비티 추가 생성 3](#_Toc404915195)

[2) 프레임 애니메이션 그림 파일 추가 4](#_Toc404915196)

[3) res/drawable-nodpi 폴더 추가 6](#_Toc404915197)

[4) res/drawable-nodpi/frame\_animation1.xml 생성 6](#_Toc404915198)

[5) activity\_frame\_animation.xml 수정 7](#_Toc404915199)

[6) FrameAnimationActivity.java 수정 9](#_Toc404915200)

[3. MainActivity 수정 10](#_Toc404915201)

[1) activity\_main.xml 파일 수정 10](#_Toc404915202)

[2) MainActivity.java 수정 11](#_Toc404915203)

[3) 실행 11](#_Toc404915204)

**학습목표**

간단한 메모장 앱을 만든다.

다음과 같은 기능을 구현한다

- 메모를 데이터베이스에 저장

- 메모 목록 보기

- 새 메모 작성 기능

- 메모 편집

- 메모 삭제

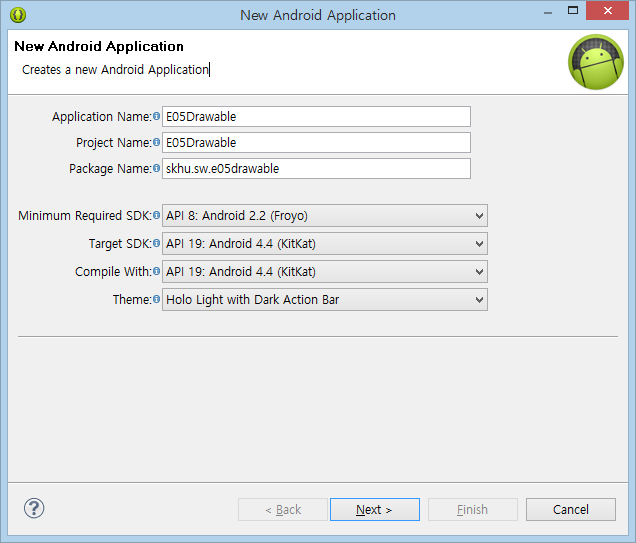
# E05Drawable 프로젝트

Drawable 리소스를 활용하는 간단한 예제들을 만들어보자

먼저 메인 화면을 보여주는 액티비티부터 만들자

## 프로젝트 생성

이클립스 메뉴: File – New – Andoid Applicaton Project



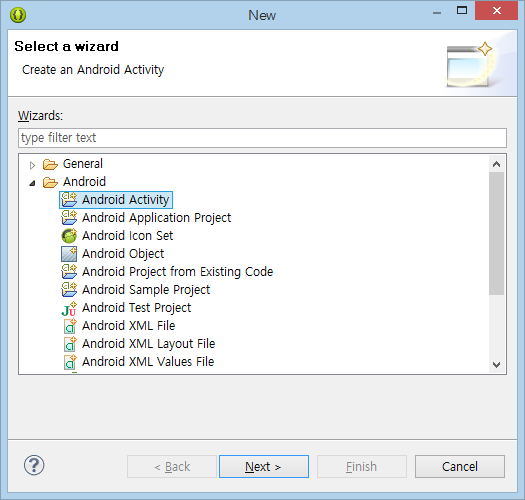
위와 같이 입력하고 Next를 계속 눌러서 진행하자.

# FrameAnimationActivity 액티비티 생성

## 프로젝트에 FrameAnimationActivity 액티비티 생성

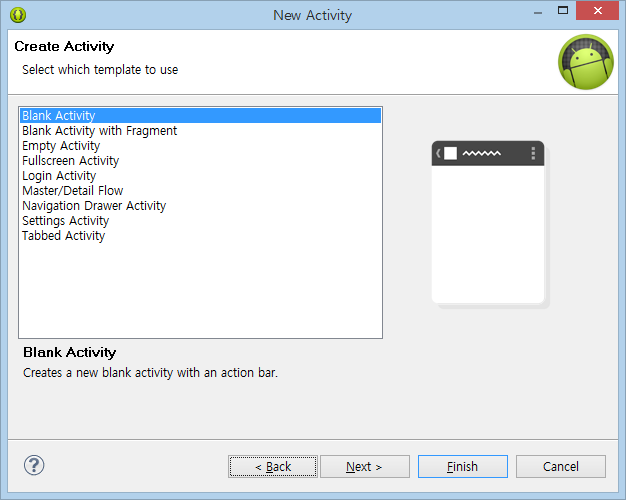
간단한 프레임 애니메이션(frame animation)을 보여주기 위한 FrameAnimationActivity 클래스를 구현하자.

메뉴: File – New – Other



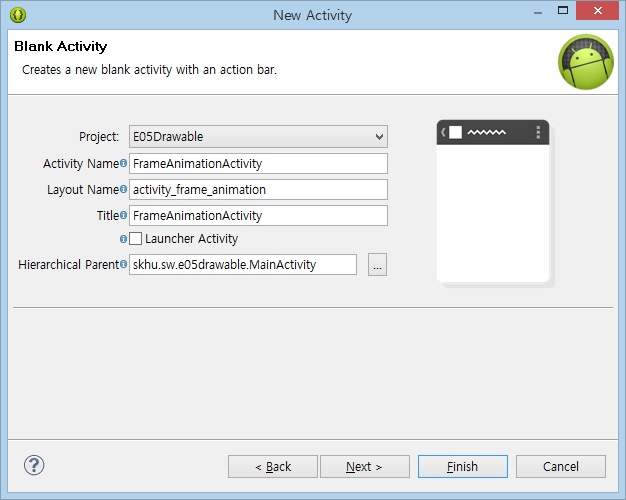
Android Activity 선택

Next 버튼 클릭



Blank Activity 선택

Next 버튼 클릭



|  |  |  |
| --- | --- | --- |
| 입력 항목 | 입력 값 | 입력 항목 설명 |
| Project: | E05Drawable | 현재 프로젝트명 |
| Activity Name: | FrameAnimationActivity | 액티비티 클래스명 |
| Layout Name: | activity\_frame\_animation | 레이아웃 리소스 XML 파일명 |
| Title: | FrameAnimatioActivity | 액티비티 제목줄에 표시될 문자열 |
| Launcher Activity: | 체크 안함 | 이 액티비티를 앱을 시작할 때 처음 실행되는 시작 액티비티로 설정할 것인가? |
| Hierarchical Parent: | skhu.sw.e05drawable.MainActivity | 이 액티비티에서 백 버튼을 눌렀을 때 자동으로 넘어갈 목적지 액티비티 클래스 지정 |

위와 같이 입력하고 Next 버튼 클릭

다음 화면에서 그냥 Finish 버튼을 클릭.

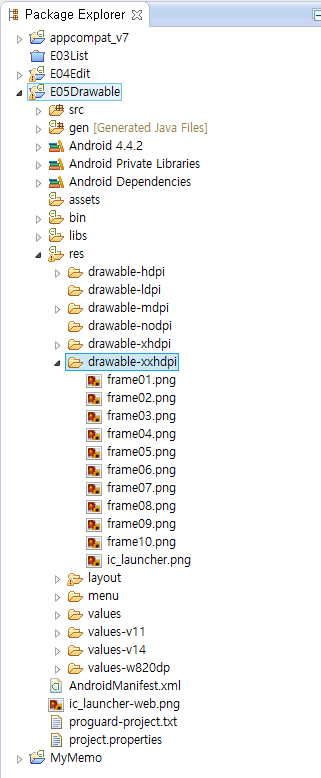
## 프레임 애니메이션 그림 파일 추가

프레임 애니메이션이란 조금씩 다르게 그린 그림 여러장을 순서대로 재빨리 교체해서 보여주어, 마치 그림이 움직이는 것 같은 시각적 효과를 일으키는 애니메이션 기법을 말한다. 전통 만화 영화 제작 기법이기도 하다.

프레임 애니메이션에 보여줄 그림 여러장을 프로젝트의 리소스(resource)에 추가하자.

강의노트 웹폴더에서 frame\_animation\_images.zip 압축파일을 다운로드하자.

이 압축파일에 들어있는 이미지 파일(frame01.png ~ frame10.png)들을 프로젝트의 res/drawable-xxhdpi 폴더 아래에 복사하자.



추가된 그림 파일들의 ID는 다음과 같다.

XML 파일에서 ID를 사용할 때: @drawable/frame01, @drawable/frame02, ~ @drawable/frame10

Java 파일에서 ID를 사용할 때: R.drawable.frame01, R.drawable.frame02, ~ R.drawable.frame10

안드로이드 기기의 해상도별로 적당한 크기의 이미지를 따로 저장하기 위해서, drawable 리소스 폴더가 여러개이다 (drawable-ldpi, drawable-mdpi, drawable-hdpi, drawable-xhdpi, drawable-xxhdpi).

안드로이드 앱에서 사용할 그림 파일을 해상도별로 다른 크기로 여러개를 각각 따로 만들었다면, 그림 파일들을 해상도별 drawable 폴더에 각각 넣어주면 된다.

그림 파일을 해상도별로 따로 만들지 않고 그냥 적당한 크기로 리사이즈(resize)만 해도 충분하다면, 굳이 여러개 따로 만들 필요없이, drawable-xxhdpi 해상도에 적당한 크기로 하나만 만들어서 이 폴더에만 저장해도 된다. 다른 해상도에 적당한 크기로 리사이즈 하는 것은 안드로이드 시스템이 자동으로 처리해준다.

## res/drawable-nodpi 폴더 추가

화면 해상도와 무관한 drawable 리소스는 drawable-nodpi 폴더 아래에 저장하는 것이 좋다.

res 폴더 아래에 drawable-nodpi 폴더를 만들자.

이클립스의 Package Explorer 창에서 res 폴더를 우클릭하고 메뉴에서 New – Folder를 클릭하자.

Folder name: drawable-nodpi

## res/drawable-nodpi/frame\_animation1.xml 생성

이클립스의 Package Explorer 창에서 res/drawable-nodpi 폴더를 우클릭하고 메뉴에서 New – File을 클릭하자.

File name: frame\_animation1.xml

### res/drawable-nodpi/frame\_animation1.xml

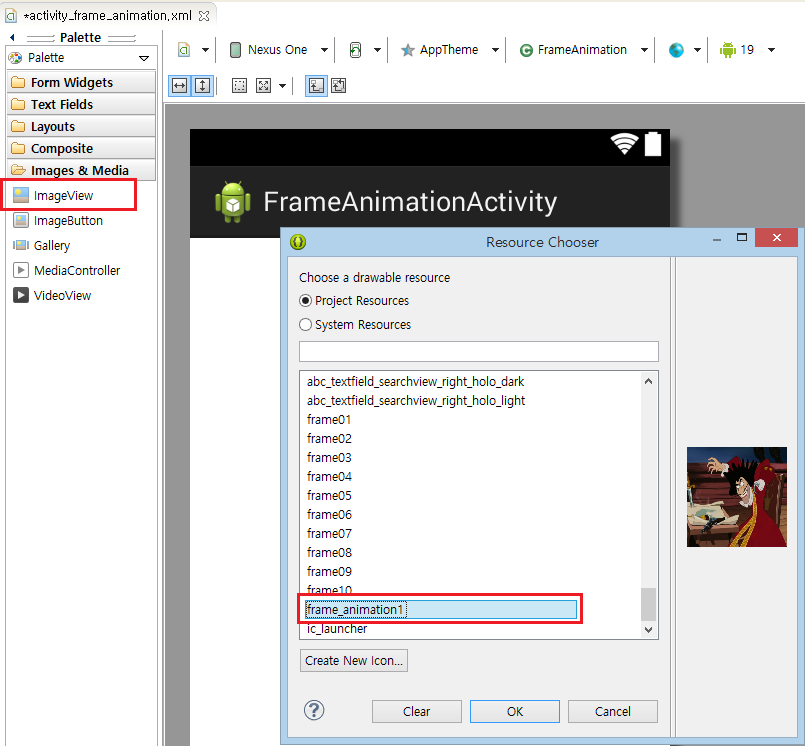
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <?xml version="1.0" encoding="utf-8"?>  <animation-list xmlns:android="http://schemas.android.com/apk/res/android"  android:oneshot="false">  <item android:drawable="@drawable/frame01" android:duration="200" />  <item android:drawable="@drawable/frame02" android:duration="200" />  <item android:drawable="@drawable/frame03" android:duration="200" />  <item android:drawable="@drawable/frame04" android:duration="200" />  <item android:drawable="@drawable/frame05" android:duration="200" />  <item android:drawable="@drawable/frame06" android:duration="200" />  <item android:drawable="@drawable/frame07" android:duration="200" />  <item android:drawable="@drawable/frame08" android:duration="200" />  <item android:drawable="@drawable/frame09" android:duration="200" />  <item android:drawable="@drawable/frame10" android:duration="200" />  </animation-list> |

프레임 애니메이션을 정의한 xml 리소스 파일이다.

frame01.png ~ frame10.png 파일들을 200 밀리초의 간격으로 연속해서 보여주는 애니메이션을 정의하였다.

## activity\_frame\_animation.xml 수정

FrameAnimationActivity의 레이아웃 리소스 XML 파일인 activity\_frame\_animation.xml을 수정하자.



액티비티에 이미 추가되어있는 TextView를 제거하고, ImageView를 추가하자.

화면 왼쪽 팔레트(Palette)에서 ImageView를 드래그하여 액티비티에 드롭하면,

위 그림에서 볼 수 있듯이 Resource Chooser 창이 표시된다.

지금 드래그 드롭하여 추가하려는 ImageView에 표시될 drawable 리소스를 선택하라는 창이다.

프로젝트에 포함되어있는 drawable 리소스들의 목록을 이 창에서 확인할 수 있다.

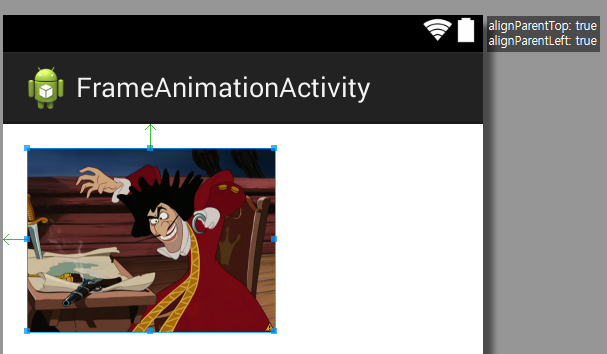
앞에서 우리가 추가한 frame01.png ~ frame10.png 그림 파일들도 이 목록에 포함되어 있다.

이 그림 파일 중 하나를 선택하면 ImageView에 그 그림이 표시된다.

방금 우리가 추가한 frame\_animation1.xml 파일도 이 목록에 포함되어 있다.

이 애니메이션 XML 파일을 선택하면 ImageView에 그 애니메이션이 표시된다.

frame\_animation1을 선택하자.



## FrameAnimationActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | package skhu.sw.e05drawable;  import android.support.v7.app.ActionBarActivity;  import android.graphics.drawable.AnimationDrawable;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;  import android.widget.ImageView;  public class FrameAnimationActivity extends ActionBarActivity {    @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_frame\_animation);  }  @Override  public void onWindowFocusChanged(boolean hasFocus) {  super.onWindowFocusChanged(hasFocus);  ImageView imageView1 = (ImageView)findViewById(R.id.imageView1);  AnimationDrawable animation = (AnimationDrawable)imageView1.getDrawable();  if (hasFocus)  animation.start();  else  animation.stop();  }    @Override  public boolean onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.frame\_animation, menu);  return true;  }  @Override  public boolean onOptionsItemSelected(MenuItem item) {  int id = item.getItemId();  if (id == R.id.action\_settings) {  return true;  }  return super.onOptionsItemSelected(item);  }  } |

(줄19) onWindowsFocusChanged 메소드

메소드 이름이 on으로 시작하고 있으므로 특정 이벤트에 반응하여 저절로 호출되는 이벤트 핸들러 메소드이다.

메소드 이름에서 Window는 액티비티 화면을 뜻한다.

Focus는 화면에서 지금 현재 터치 이벤트를 받을 수 있는 최상단임을 뜻한다.

액티비티 위에 메뉴나 대화상자가 출력되면, 액티비티는 Focus를 잠시 잃었다가, 메뉴나 대화상자가 닫히면 다시 Focus를 얻는다. 다른 액티비티로 전환될 때도 Focus를 잃는다.

포커스(focus)를 얻거나 잃을 때 onWindowsFocusChanged 메소드가 즉시 호출된다.

(줄21) 액티비티 내부의 ImageView 객체에 대한 참조를 얻는다.

(줄22) ImageView에서 보여지는 내용에 해당하는 drawable 리소스 객체에 대한 참조를 얻는다.

그것이 애니메이션 drawable 리소스일 경우에 리턴되는 객체는 AnimationDrawable 클래스이다.

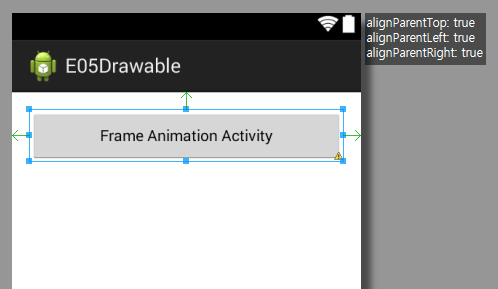
(줄24) 액티비티가 포커스(focus)를 소유하게 될 때 애니메이션을 시작한다.

(줄25) 액티비티가 포커스를 잃을 때 애니메이션을 종료한다.

# MainActivity 수정

## activity\_main.xml 파일 수정

MainActivity의 레이아웃 리소스 파일인 activity\_main.xml 파일을 열어서 수정하자



FrameAnimationActivity를 실행하기위한 버튼을 위와같이 생성하자.

## MainActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 | package skhu.sw.e05drawable;  import android.support.v7.app.ActionBarActivity;  import android.content.Intent;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;  import android.view.View;  import android.view.View.OnClickListener;  public class MainActivity extends ActionBarActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);    OnClickListener myOnClickListener = new OnClickListener() {  @Override  public void onClick(View button) {  if (button.getId() == R.id.button1)  startActivity(new Intent(MainActivity.this, FrameAnimationActivity.class));  }  };    findViewById(R.id.button1).setOnClickListener(myOnClickListener);  }  @Override  public boolean onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.main, menu);  return true;  }  @Override  public boolean onOptionsItemSelected(MenuItem item) {  int id = item.getItemId();  if (id == R.id.action\_settings) {  return true;  }  return super.onOptionsItemSelected(item);  }  } |

(줄19~27) FrameAnimationActivity를 실행하기 위한 버튼 R.id.button1에 OnClickListener를 등록하였다.

(줄23) FrameAnimationActivity를 호출한다.

## 실행

앱을 실행하여 프레임 애니메이션을 확인하자.

메뉴 버튼을 눌르면 프레임 애니메이션이 잠시 멈추는 것을 확인하자.

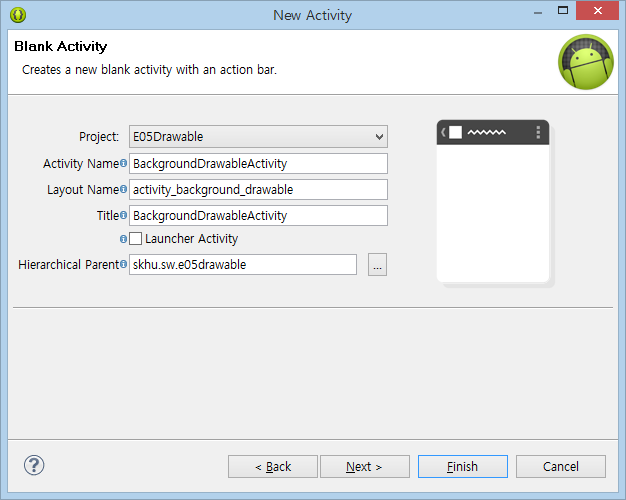
결과: E05Drawable\_#1.zip

# BackgroundDrawableActivity 액티비티 생성

## 프로젝트에 BackgroundDrawableActivity 액티비티 생성

뷰(View)와 액티비티(Activity)의 백그라운드(background)를 설정 방법을 살펴볼 액티비티를 생성하자.

메뉴: File – New – Other – Android Acitivty



## Shape Drawable 리소스 생성

shape drawable 리소스는 화면에 그림을 그리는 명령이 내용인 XML 파일이다.

이클립스의 Package Explorer 창에서 res/drawable-nodpi 폴더를 우클릭하고 메뉴에서 New – File을 클릭하자.

File name: shape1.xml

### res/drawable-nodpi/shape1.xml

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <shape xmlns:android="http://schemas.android.com/apk/res/android" >  <solid android:color="#00FFFF" />  <stroke  android:width="1px"  android:color="#0000FF" />  </shape> |

(줄2) 도형의 내부를 노란색으로 채우라는 명령이다.

(줄3~5) 도형의 경계선을 1px 파란색으로 그리라는 명령이다.

### res/drawable-nodpi/shape2.xml

|  |  |
| --- | --- |
| 1  2  3  4  5 | <shape xmlns:android="http://schemas.android.com/apk/res/android" >  <gradient  android:startColor="#FFFFFF"  android:endColor="#0000FF" />  </shape> |

(줄2~4) 도형의 내부를 흰색에서 시작하여 점점 파란색으로 변해가며 칠하라는 명령이다.

색이 변화하는 방향은 디폴트 값인 수직 방향이다.

### res/drawable-nodpi/shape3.xml

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <shape xmlns:android="http://schemas.android.com/apk/res/android" >  <gradient  android:startColor="#FFFFFF"  android:endColor="#88FF88"  android:angle="45" />  </shape> |

(줄2~4) 도형의 내부를 흰색에서 시작하여 점점 밝은 녹색으로 변해가며 칠하라는 명령이다.

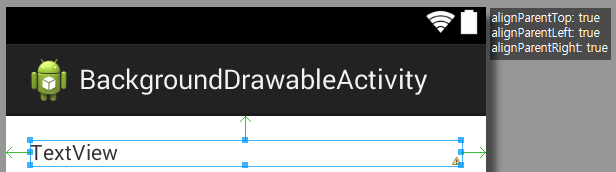
(줄5) 색이 변화하는 방향은 45도 방향이다.

## acivity\_background\_drawable.xml 수정

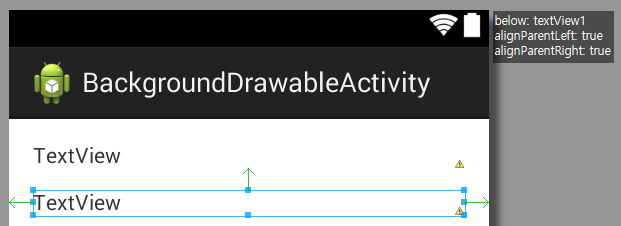
BackgroundDrawableActivity의 레이아웃 리소스 리소스 XML 파일인 activity\_background\_drawable.xml을 수정하자.

먼저 자동으로 만들어져 있는 "Hello World" TextView를 제거하자.

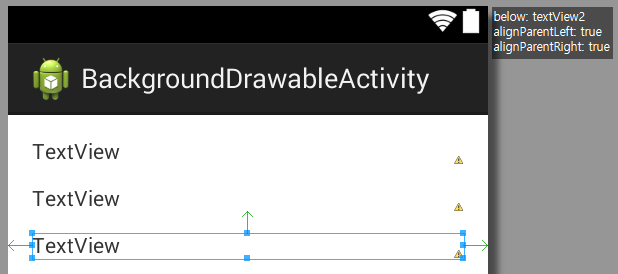
첫째 TextView를 다음과 같이 추가하자



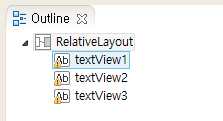
둘째 TextView를 다음과 같이 추가하자



셋째 TextView를 다음과 같이 추가하자



위 3개의 TextView의 ID는 다음과 같아야 한다.

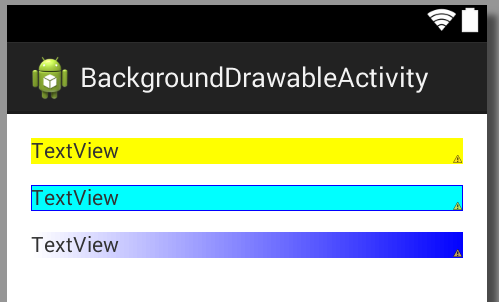


위 3개의 TextView의 Background 속성을 다음과 같이 설정하자.

textView1의 Background 속성: #FFFF00

textView2의 Background 속성: @drawable/shape1

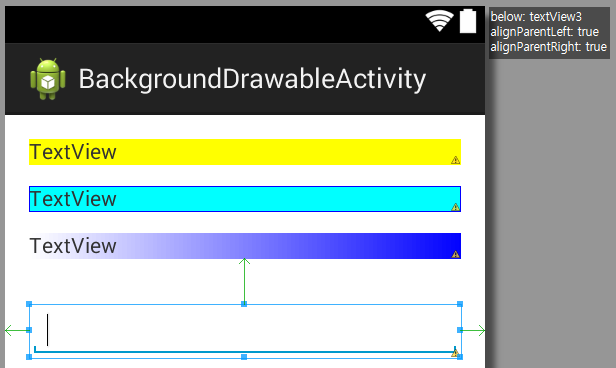
textView3의 Background 속성: @drawable/shape2



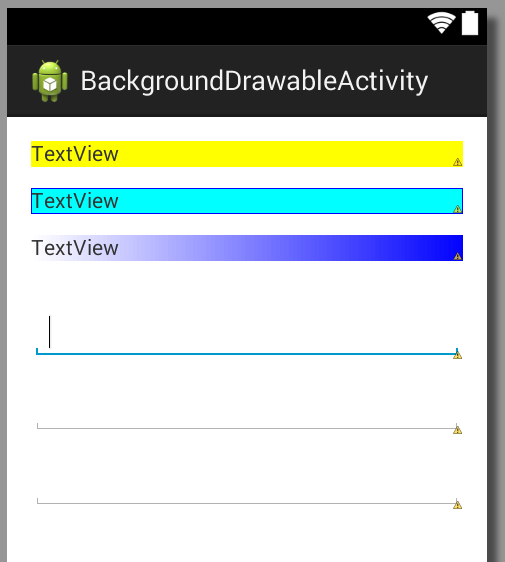
textView1의 Background 속성은 직접 색을 노란색으로 지정하였다 (#FFFF00)

textView2와 textView3의 Background 속성은 shape drawable resource의 ID로 지정하였다.

다음과 같이 EditText를 추가하자



그 아래에 EditText를 두 개 더 추가하자

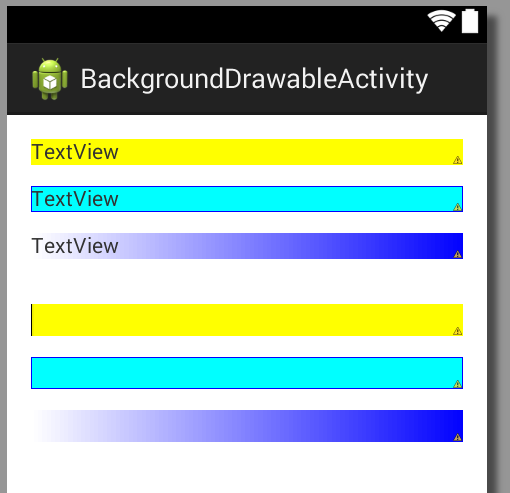


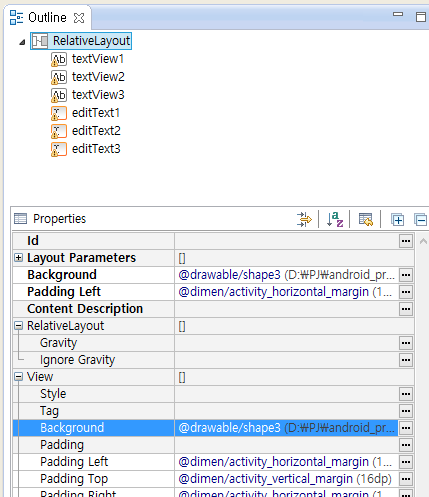
위 3개의 EditText의 Background 속성을 다음과 같이 설정하자.

editText1의 Background 속성: #FFFF00

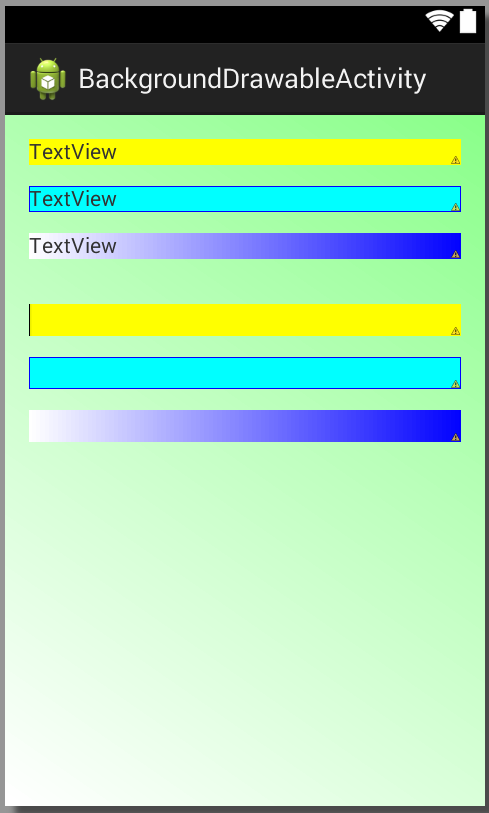
editText2의 Background 속성: @drawable/shape1

editText3의 Background 속성: @drawable/shape2





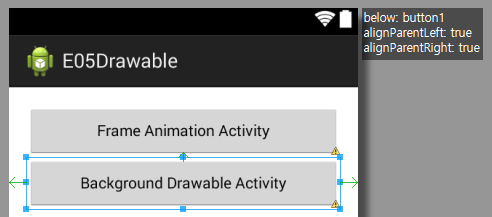
RelativeLayout의 Background 속성을 @drawable/shape3 로 설정하자.



# MainActivity 수정

## activity\_main.xml 파일 수정

MainActivity의 레이아웃 리소스 파일인 activity\_main.xml 파일을 열어서 수정하자



BackgroundDrawableActivity를 실행하기위한 버튼을 위와같이 생성하자.

## MainActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48 | package skhu.sw.e05drawable;  import android.support.v7.app.ActionBarActivity;  import android.content.Intent;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;  import android.view.View;  import android.view.View.OnClickListener;  public class MainActivity extends ActionBarActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);    OnClickListener myOnClickListener = new OnClickListener() {  @Override  public void onClick(View button) {  if (button.getId() == R.id.button1)  startActivity(new Intent(MainActivity.this, FrameAnimationActivity.class));  else if (button.getId() == R.id.button2)  startActivity(new Intent(MainActivity.this, BackgroundDrawableActivity.class));  }  };    findViewById(R.id.button1).setOnClickListener(myOnClickListener);  findViewById(R.id.button2).setOnClickListener(myOnClickListener);  }  @Override  public boolean onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.main, menu);  return true;  }  @Override  public boolean onOptionsItemSelected(MenuItem item) {  int id = item.getItemId();  if (id == R.id.action\_settings) {  return true;  }  return super.onOptionsItemSelected(item);  }  } |

## 실행

앱을 실행하여 백그라운드 색 변경을 확인하자.

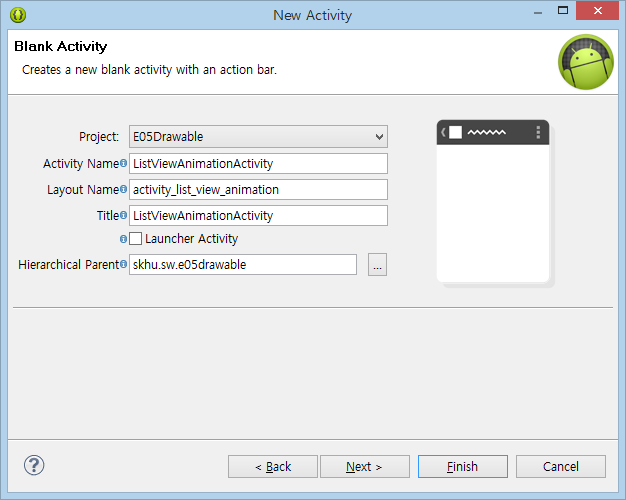
결과: E05Drawable\_#2.zip

# ListViewAnimationActivity 액티비티 생성

## 프로젝트에 ListViewAnimationActivity 액티비티 생성

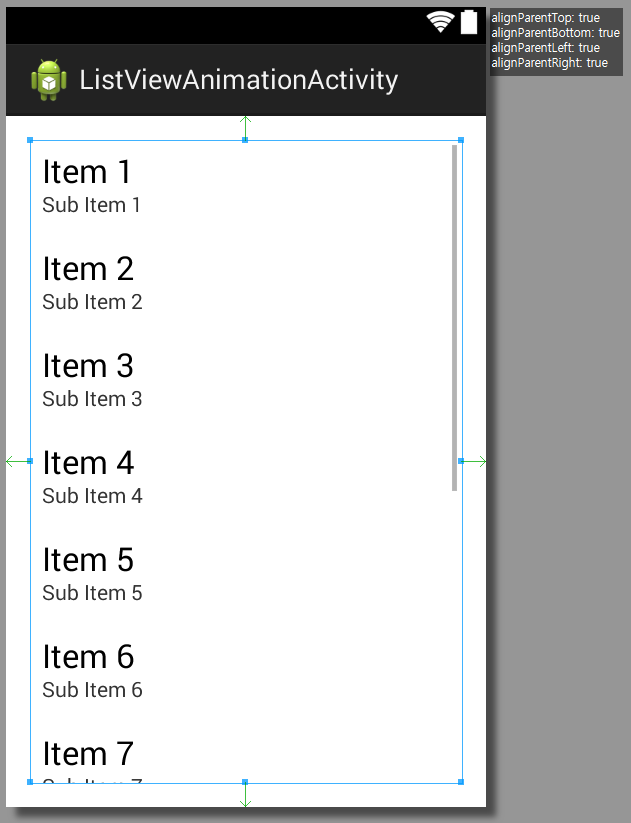
뷰(View)의 애니메이션 효과를 방법을 살펴볼 액티비티를 생성하자.

메뉴: File – New – Other – Android Acitivty



## activity\_list\_view\_animation.xml 수정

ListViewAnimationActivity의 레이아웃 리소스 XML 파일인 activity\_list\_view\_animation.xml을 수정하자.



위와 같이 ListView를 추가하자

## res/vaues/arrays.xml 파일 생성

화면에 표시할 문자열 목록을 저장할 res/values/arrays.xml 파일을 생성한다.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | <?xml version="1.0" encoding="utf-8"?>  <resources>  <string-array name="list1">  <item>안드로이드 프로그래밍 기술</item>  <item>자바 웹 서버 프로그래밍 기술</item>  <item>자바 스프링 프로그래밍 기술</item>  <item>데이터베이스 설계 기술</item>  <item>HTML 자바스크립트 프로그래밍 기술</item>  <item>자바스크립트 서버 프로그래밍 기술</item>  <item>안드로이드 프로그래밍 기술</item>  <item>자바 웹 서버 프로그래밍 기술</item>  <item>자바 스프링 프로그래밍 기술</item>  <item>데이터베이스 설계 기술</item>  <item>HTML 자바스크립트 프로그래밍 기술</item>  <item>자바스크립트 서버 프로그래밍 기술</item>  <item>안드로이드 프로그래밍 기술</item>  <item>자바 웹 서버 프로그래밍 기술</item>  <item>자바 스프링 프로그래밍 기술</item>  <item>데이터베이스 설계 기술</item>  <item>HTML 자바스크립트 프로그래밍 기술</item>  <item>자바스크립트 서버 프로그래밍 기술</item>  </string-array>  </resources> |

수정될 필요없고 화면에 표시할 문자열 목록은 위와같이 string-array 리소스로 저장하는 것이 좋다.

이 리소스의 ID는 R.array.list1 이다.

## ListViewAnimationActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | package skhu.sw.e05drawable;  import android.support.v7.app.ActionBarActivity;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;  import android.widget.ArrayAdapter;  import android.widget.ListView;  public class ListViewAnimationActivity extends ActionBarActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_list\_view\_animation);    ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,  R.array.list1, android.R.layout.simple\_list\_item\_1);  ListView listView1 = (ListView)findViewById(R.id.listView1);  listView1.setAdapter(adapter);  }  @Override  public boolean onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.list\_view\_animation, menu);  return true;  }  @Override  public boolean onOptionsItemSelected(MenuItem item) {  int id = item.getItemId();  if (id == R.id.action\_settings) {  return true;  }  return super.onOptionsItemSelected(item);  }  } |

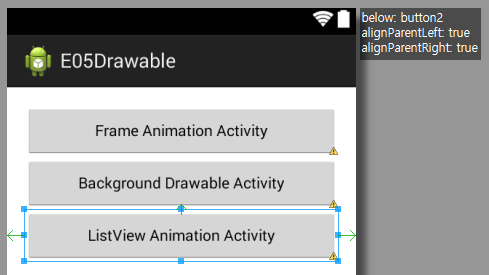
(줄17~18) 문자열 목록 리소스를 ListView로 보여주기 위해 ArrayAdapter 객체를 생성한다.

(줄20) ListView에 ArrayAdapter를 연결한다.

# MainActivity 수정

## activity\_main.xml 파일 수정

MainActivity의 레이아웃 리소스 파일인 activity\_main.xml 파일을 열어서 수정하자



ListViewAnimationActivity를 실행하기위한 버튼을 위와같이 생성하자.

## MainActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | package skhu.sw.e05drawable;  import android.support.v7.app.ActionBarActivity;  import android.content.Intent;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;  import android.view.View;  import android.view.View.OnClickListener;  public class MainActivity extends ActionBarActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);    OnClickListener myOnClickListener = new OnClickListener() {  @Override  public void onClick(View button) {  if (button.getId() == R.id.button1)  startActivity(new Intent(MainActivity.this, FrameAnimationActivity.class));  else if (button.getId() == R.id.button2)  startActivity(new Intent(MainActivity.this, BackgroundDrawableActivity.class));  else if (button.getId() == R.id.button3)  startActivity(new Intent(MainActivity.this, ListViewAnimationActivity.class));  }  };    findViewById(R.id.button1).setOnClickListener(myOnClickListener);  findViewById(R.id.button2).setOnClickListener(myOnClickListener);  findViewById(R.id.button3).setOnClickListener(myOnClickListener);  }  @Override  public boolean onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.main, menu);  return true;  }  @Override  public boolean onOptionsItemSelected(MenuItem item) {  int id = item.getItemId();  if (id == R.id.action\_settings) {  return true;  }  return super.onOptionsItemSelected(item);  }  } |

## 실행

아직 애니메이션 효과를 추가하지 않았다.

결과: E05Drawable\_#3.zip

# ListViewAnimationActivity에 애니메이션 효과 추가

## res/anim 폴더 생성

애니메이션 효과를 정의하는 XML 파일은 res/anim 폴더에 저장해야 한다.

res 폴더 아래에 anum 폴더를 생성하자.

이클립스의 Package Explorer 창에서 res 폴더를 우클릭하고 메뉴에서 New – Folder를 클릭하자.

Folder name: anim

## res/anim/animation1.xml 파일 생성

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <?xml version="1.0" encoding="utf-8"?>  <set xmlns:android="http://schemas.android.com/apk/res/android"  android:interpolator="@android:anim/accelerate\_interpolator" >  <translate  android:duration="500"  android:fromXDelta="100%"  android:toXDelta="0" />  </set> |

뷰 객체 하나에 대한 애니메이션을 정의한 XML 파일이다.

(줄5) translate 애니메이션을 정의한다. translate 애니메이션은 좌표 이동이다.

(줄6) 애니메이션은 500 밀리초 동안 실행된다

(줄7) 이동 애니메이션이 시작할 시작 x 좌표는 화면의 오른쪽 끝(100%)이다.

(줄8) 이동 애니메이션이 끝나는 마지막 x 좌표는 화면의 왼쪽 끝이다.

## res/anim/layout\_animation1.xml 파일 생성

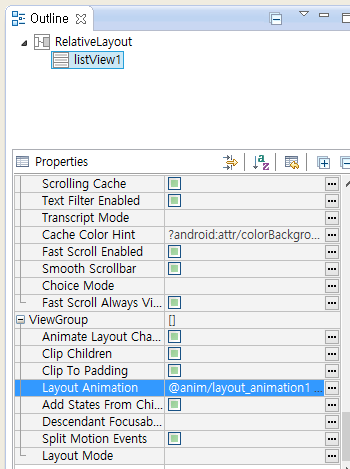
|  |  |
| --- | --- |
| 1  2  3  4 | <?xml version="1.0" encoding="utf-8"?>  <layoutAnimation xmlns:android="http://schemas.android.com/apk/res/android"  android:animation="@anim/animation1"  android:delay="10%" /> |

ListView와 같은 뷰그룹(ViewGroup) 내부의 뷰(View) 항목들 각각에 대한 애니메이션을 정의한 XML 파일이다.

(줄3) 뷰 항목 각각에 대해 적용할 애니메이션을 지정한다.

(줄4) 뷰 항목 각각의 애니메이션 사이 대기 시간을 지정한다. 뷰 항목 하나의 애니메이션 길이(duration)의 10% 라는 의미.

## activity\_list\_view\_animation.xml 수정



listView1의 Layout Animation 속성을 @anim/layout\_animation1 으로 설정한다.

## 실행

실행하여 레이아웃 애니메이션의 효과를 확인하라.

결과: