안드로이드 앱 프로그래밍

Chapter 09

애니메이션과 다양한 위젯 사용하기

이번 장에서는 무엇을 다룰까요?

애니메이션에 대해 알고 싶어요.

화면에 넣을 수 있는 다른 위젯도 있나요?

- 애니메이션을 동작시키는 방법에 대해 알아볼까요?
- 페이지가 스윽~ 나타나도록 하는 방법에 대해 알아볼까요?
- 앱 화면 안에 웹사이트가 보이도록 만들어볼까요?
- 시크바를 사용하는 방법을 알아볼까요?
- 키패드를 상황에 맞게 설정하는 방법에 대해 알아볼까요?

강의 주제 및 목차

강의 주제

애니메이션과 다양한 위젯에 대해 알아보기

1. 애니메이션 사용하기
2. 페이지 슬라이딩 사용하기
3. 뷰페이저 사용하기

1. 애니메이션 사용하기

자주 사용하는 애니메이션 방식

- 확대/축소, 이동, 회전, 투명도 조절 애니메이션을 자주 사용함
- 뷰 객체나 그리기 객체에 애니메이션을 적용할 수 있음

확대/축소

이동

회전

투명도

트윈 애니메이션

- **트윈 애니메이션(Tweened Animation)**

- 뷰 애니메이션이라고도 하며, 보여줄 대상을 적절하게 연산한 후 그 결과를 연속적으로 디스플레이하는 방식임
- 애니메이션 대상과 변환 방식을 지정하면 애니메이션 효과를 낼 수 있도록 만들어 줌
- 따라서 프레임 애니메이션처럼 변경하면서 보여줄 각각의 이미지를 추가할 필요 없이 대상만 지정하면 시스템이 내부적으로 적절하게 연산하는 과정을 거치게 됨

- **트윈 애니메이션을 위한 액션(Action) 정보**

- XML 리소스로 정의하거나 자바 코드에서 직접 객체로 만듬
- 애니메이션을 위한 XML 파일은 [/app/res/anim] 폴더의 밑에 두어야 하며 확장자를 xml로 함
- 리소스로 포함된 애니메이션 액션 정의는 다른 리소스와 마찬가지로 빌드할 때 컴파일되어 설치 파일에 포함됨

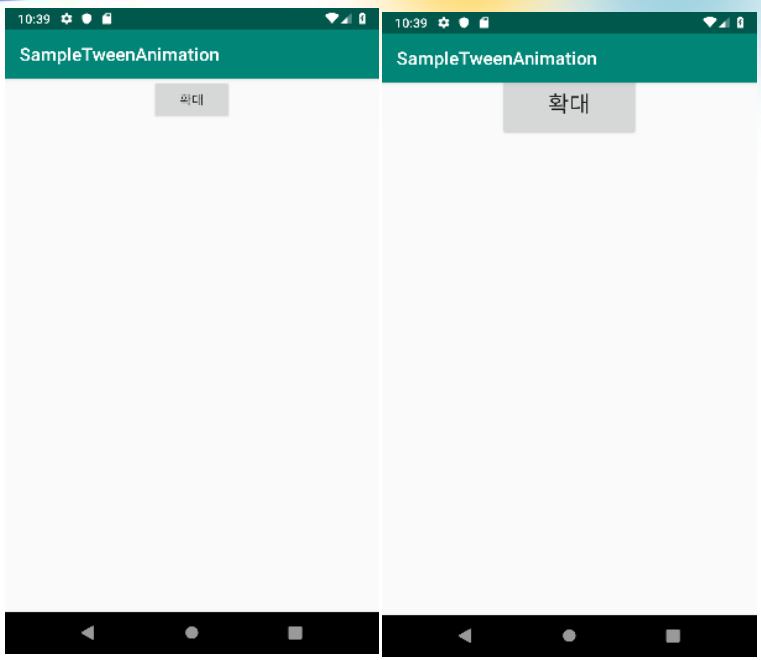
트윈 애니메이션 대상과 애니메이션 효과

구 분	이 름	설 명
대상	뷰	- View는 위젯이나 레이아웃을 모두 포함 - 예를 들어, 텍스트뷰나 리니어 레이아웃에 애니메이션을 적용할 수 있음
	그리기 객체	- 다양한 Drawable에 애니메이션을 적용할 수 있음 - ShapeDrawable은 캔버스에 그릴 도형을 지정할 수 있음 - BitmapDrawable은 비트맵 이미지를 지정할 수 있음
효과	위치 이동	- Translate로 정의되는 액션은 대상의 위치를 이동하기 위해 사용되는 효과
	확대 / 축소	- Scale로 정의되는 액션은 대상의 크기를 크게 하거나 작게 하기 위해 사용되는 효과
	회전	- Rotate로 정의되는 액션은 대상을 회전하기 위해 사용되는 효과
	투명도	- Alpha로 정의되는 액션은 대상의 투명도를 조절하는데 사용되는 효과

버튼 확대 애니메이션 예제

버튼 확대 애니메이션 예제

-버튼에 간단한 트윈 애니메이션 적용



메인 액티비티의 XML 레이아웃 정의

-버튼을 포함하는 레이아웃 정의

애니메이션 액션 정의

-XML로 애니메이션 액션 정의

메인 액티비티 코드 작성

-버튼에 애니메이션 적용

애니메이션 액션 XML 정의

- /app/res/anim 폴더 안에 새로운 XML 파일 생성

참조파일 SampleTweenAnimation>/app/res/anim/scale.xml

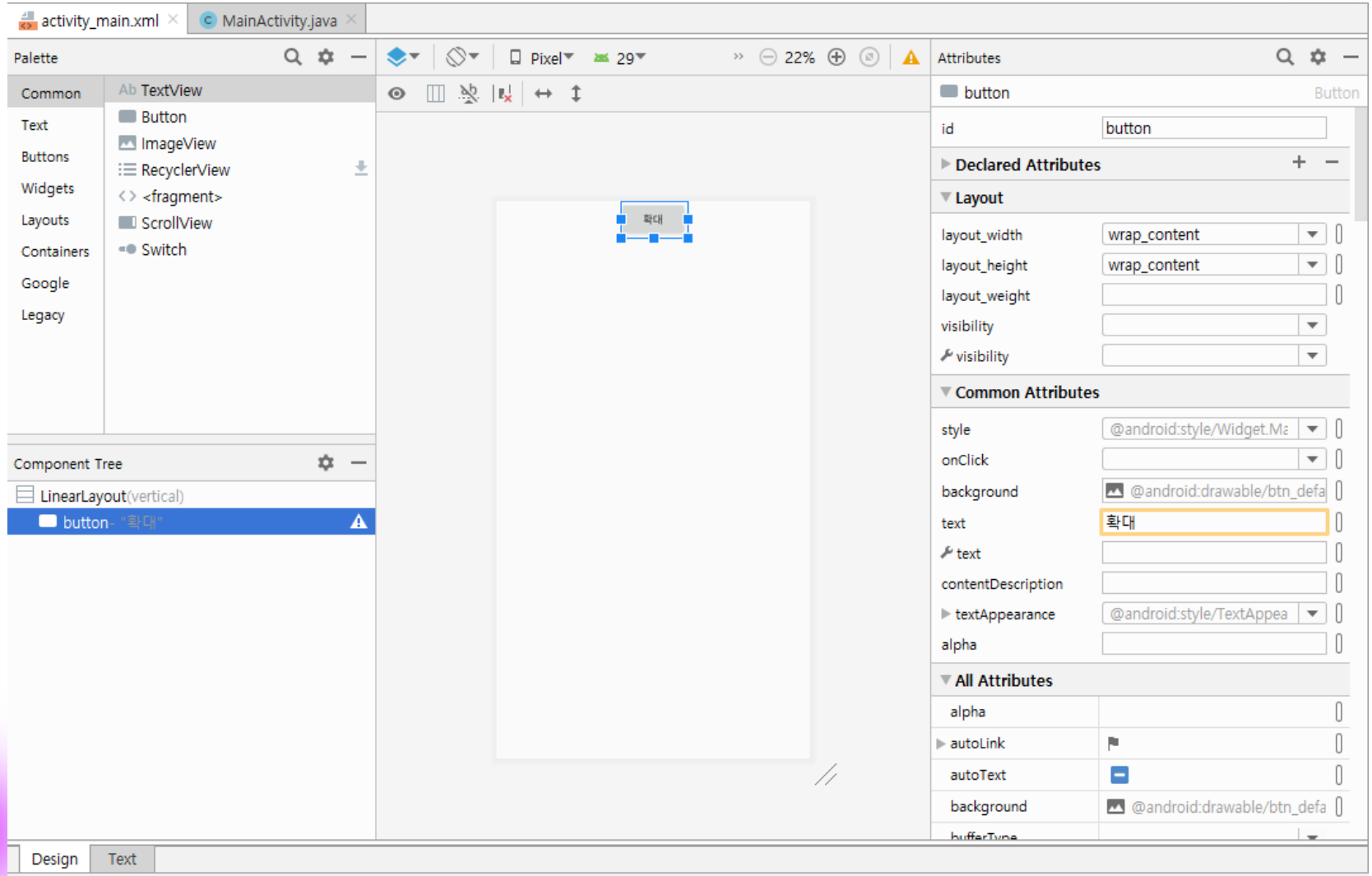
```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
  <scale
    android:duration="2500"
    android:pivotX="50%"
    android:pivotY="50%"
    android:fromXScale="1.0"
    android:fromYScale="1.0"
    android:toXScale="2.0"
    android:toYScale="2.0"
    />
</set>
```

애니메이션 <scale> 태그 사용

- startOffset
  - 시작할 시간을 지정하는 것
  - 애니메이션이 시작한 지 얼마 후에 이 액션이 수행될 것인지를 알 수 있도록 함
- duration
  - 애니메이션이 지속되는 시간으로 여기에서는 2.5초 동안 지속되도록 되어 있음
- <scale> 태그
  - 대상을 확대하거나 축소하는데 사용
  - 크기를 변경하기 위한 축의 정보는 X축과 Y축에 대하여 각각 pivotX와 pivotY로 지정됨
- fromXScale과 fromYScale
  - 시작할 때의 확대/축소 비율
- toXScale과 toYScale
  - 끝날 때의 확대/축소 비율

메인 액티비티의 화면 레이아웃 만들기

- 버튼 하나 추가



메인 액티비티 코드 만들기

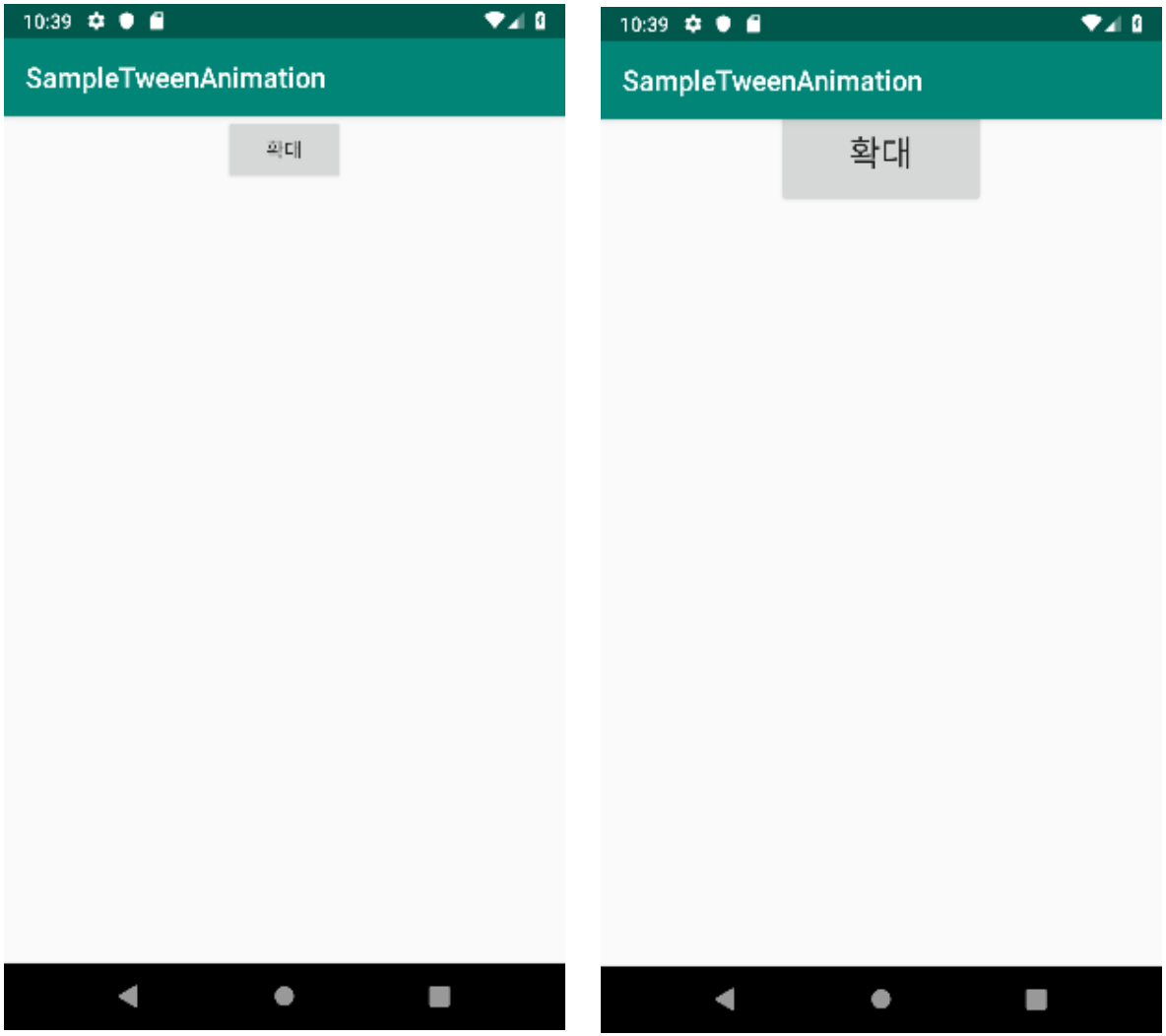
참조파일 SampleTweenAnimation>/app/java/org.techtown.sampletweenanimation/MainActivity.java

```
public class MainActivity extends AppCompatActivity {

  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button button = findViewById(R.id.button);
    button.setOnClickListener(new View.OnClickListener() {
      public void onClick(View v) {

        Animation anim =
            AnimationUtils.loadAnimation(getApplicationContext(),R.anim.scale);

        v.startAnimation(anim);
      }
    });
  }
}
```

❶ 리소스에 정의한 애니메이션 액션 로딩

❷ 뷰의 애니메이션 시작

확대/축소 애니메이션 실행 화면

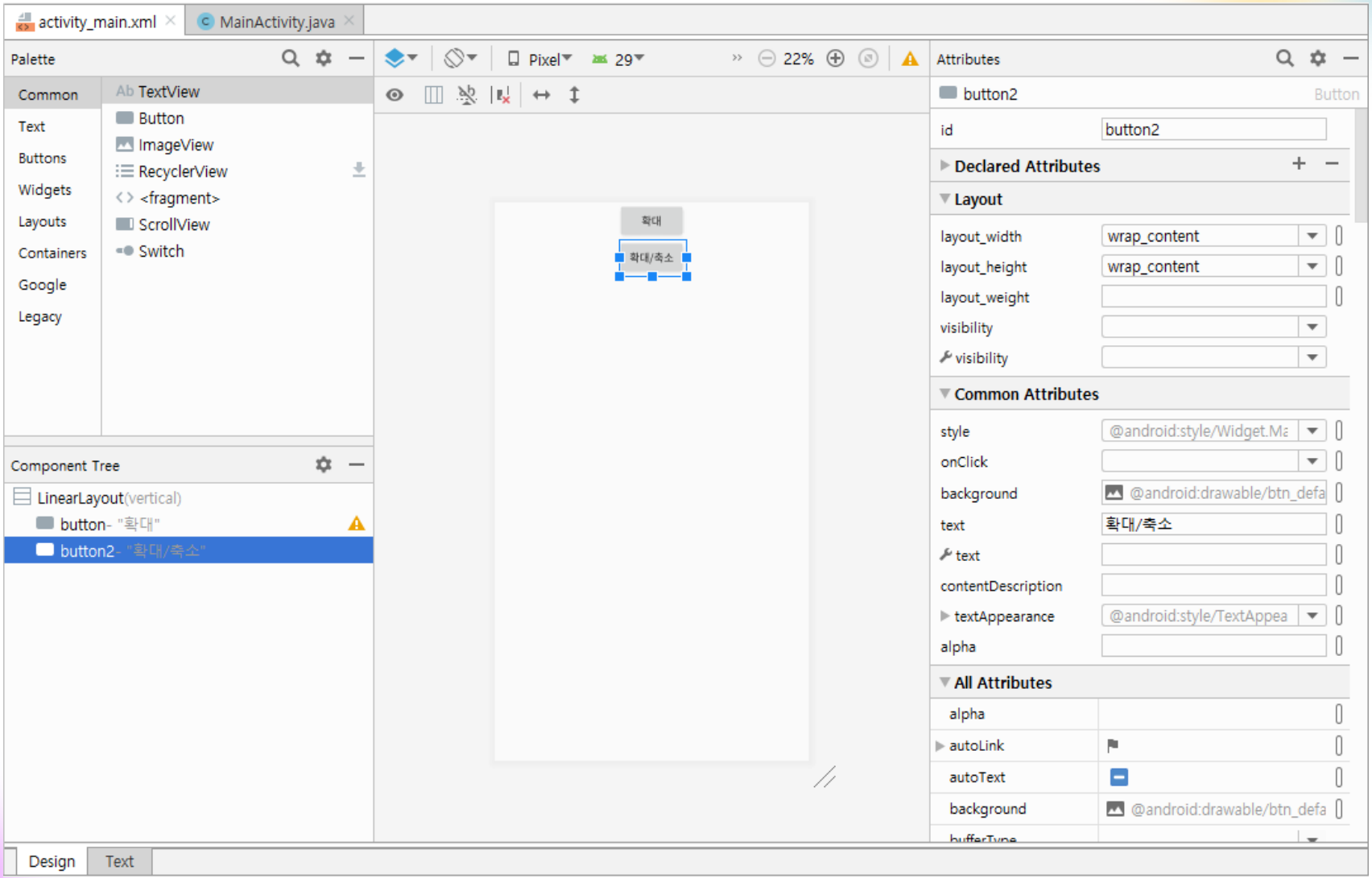


offset을 주어서 일정 시간 후 동작시키기

```
    <scale
        android:startOffset="2500"
        android:duration="2500"
        android:pivotX="50%"
        android:pivotY="50%"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:toXScale="0.5"
        android:toYScale="0.5"
        />
</set>
```

❷ 2.5초 후에 시작할 확대/축소 애니메이션 액션 정의

메인 액티비티에 두 번째 버튼 추가



애니메이션 적용

참조파일 SampleTweenAnimation>/app/java/org.techtown.sampletweenanimation/MainActivity.java

```
public class MainActivity extends AppCompatActivity {

  @Override
  public void onCreate(Bundle savedInstanceState) {
    중략…

    Button button2 = findViewById(R.id.button2);
    button2.setOnClickListener(new View.OnClickListener() {
      public void onClick(View v) {

        Animation anim =
            AnimationUtils.loadAnimation(getApplicationContext(),R.anim.scale2);

        v.startAnimation(anim);
      }
    });
  }
}
```

❶ 애니메이션 정의한 것 로딩하기

❷ 애니메이션 시작하기

트윈 애니메이션 – 위치 이동 액션

참조파일 SampleTweenAnimation>/app/res/anim/translate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<translate xmlns:android="http://schemas.android.com/apk/res/android"
  android:fromXDelta="0%p"
  android:toXDelta="-100%p"
  android:duration="20000"
  android:repeatCount="-1"
  android:fillAfter="true"
  />
```

- 위치 이동은 대상의 위치를 변경하는 것으로 한 곳에서 다른 곳으로 부드럽게 움직이는 효과를 낼 수 있음
- 위치 이동 액션은 <translate> 태그를 사용하여 정의하는데 시작 위치는 fromXDelta와 fromYDelta, 종료 위치는 toXDelta와 toYDelta라는 이름을 가진 속성으로 지정할 수 있음
- fromXDelta 속성이 0%이므로 시작 위치의 X 좌표는 원래 위치의 X 좌표가 됨
- toXDelta 속성이 -100%이므로 대상의 크기만큼 왼쪽으로 이동하게 됨
- 지속 시간은 duration의 값이 20000이므로 20초가 되며 repeatCount 속성이 -1이므로 무한반복하게 됨
- 애니메이션이 끝난 후에 대상이 원래의 위치로 돌아오는 것을 막기 위해서는 fillAfter 속성을 true로 하면 됨

트윈 애니메이션 – 회전 액션

참조파일 SampleTweenAnimation>/app/res/anim/rotate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<rotate xmlns:android="http://schemas.android.com/apk/res/android"
  android:fromDegrees="0"
  android:toDegrees="360"
  android:pivotX="50%"
  android:pivotY="50%"
  android:duration="10000"
  />
```

- 회전은 한 점을 중심으로 대상을 회전시키는 효과를 만드는 액션으로써 시작 각도와 종료 각도를 지정할 수 있음
- 한 바퀴 회전시키려 한다면 fromDegrees 속성의 값을 0으로 하고 toDegrees 속성의 값을 360으로 함
- 시계 반대 방향으로 회전시키고 싶을 경우에는 toDegrees 속성의 값을 -360으로 함. 회전의 중심이 되는 점은 디폴트 값이 (0, 0)이므로 대상의 왼쪽 상단 끝 지점이 됨.
- 대상의 중앙 부분을 회전의 중심으로 만들고 싶다면 pivotX와 pivotY 속성의 값을 지정함
- 값의 단위는 좌표 값 또는 백분율(%)을 사용할 수 있음
- duration 속성의 값이 10000으로 설정되어 있으므로 10초 동안 애니메이션이 진행된 후 원래대로 돌아오게 됨

트윈 애니메이션 – 스케일 액션

- 스케일
  - 대상을 크게 하거나 작게 할 수 있는 액션
  - 확대/축소의 정도는 대상이 갖는 원래 크기에 대한 비율로 결정
  - 1.0이라는 값은 원래 크기와 동일하다는 의미이며, 2.0은 원래 크기의 두 배로 크게 만든다는 의미
- X축으로 늘리거나 줄이고 싶으면?
  - fromXScale과 toXScale 속성을 이용하여 값을 설정
- Y축으로 늘리거나 줄이고 싶으면?
  - fromYScale과 toYScale 속성을 이용하여 값을 설정
- 확대/축소의 경우
  - 중심이 되는 점을 지정할 수 있는데 앞에서와 마찬가지로 pivotX와 pivotY 속성을 이용

트윈 애니메이션 – 투명도 액션

참조파일 SampleTweenAnimation>/app/res/anim/alpha.xml

```
<?xml version="1.0" encoding="utf-8"?>
<alpha xmlns:android="http://schemas.android.com/apk/res/android"
  android:fromAlpha="0.0"
  android:toAlpha="1.0"
  android:duration="10000"
  />
```

- 투명도를 결정하는 알파 값도 뷰나 그리기 객체의 투명도를 점차적으로 바꿀 수 있는 애니메이션 액션으로 정의
- 알파 값을 이용한 투명도 변환은 대상을 천천히 보이게 하거나 보이지 않게 하고 싶을 때 또는 하나의 뷰 위에 다른 뷰를 겹쳐 보이게 할 경우에 사용됨
- 알파 값의 범위는 0.0부터 1.0까지이며 0.0은 알파 값이 0일 때와 마찬가지이므로 완전히 투명한 상태(뷰나 그리기 객체가 보이지 않음)이며 1.0은 알파 값이 1일 때와 마찬가지이므로 완전히 보이는 상태(투명 효과가 적용되지 않음)임

트윈 애니메이션 - 인터폴레이터

- accelerate_interpolator
  - 애니메이션 효과를 점점 빠르게 나타나도록 만듬
- decelerate_interpolator
  - 애니메이션 효과를 점점 느리게 나타나도록 만듬
- accelerate_decelerate_interpolator
  - 애니메이션 효과를 점점 빠르다가 느리게 나타나도록 만듬
- anticipate_interpolator
  - 애니메이션 효과를 시작 위치에서 조금 뒤로 당겼다가 시작하도록 만듬
- overshoot_interpolator
  - 애니메이션 효과를 종료 위치에서 조금 지나쳤다가 종료되도록 만듬

트윈 애니메이션 – 인터폴레이터 (계속)

- anticipate_interpolator
  - 애니메이션 효과를 시작 위치에서 조금 뒤로 당겼다가 시작한 후 종료 위치에서 조금 지나쳤다가 종료되도록 만듬

- bounce_interpolator
  - 애니메이션 효과를 종료 위치에서 튀도록 만듬

- 위치 이동 <translate> → TranslateAnimation
- 회전 <rotate> → RotateAnimation
- 확대/축소 <scale> → ScaleAnimation
- 투명도 <alpha> → AlphaAnimation
- 애니메이션 집합 <set> → AnimationSet

리스너 사용하기

- 리스너를 사용하면 애니메이션이 시작되거나 끝나는 시점을 알 수 있음

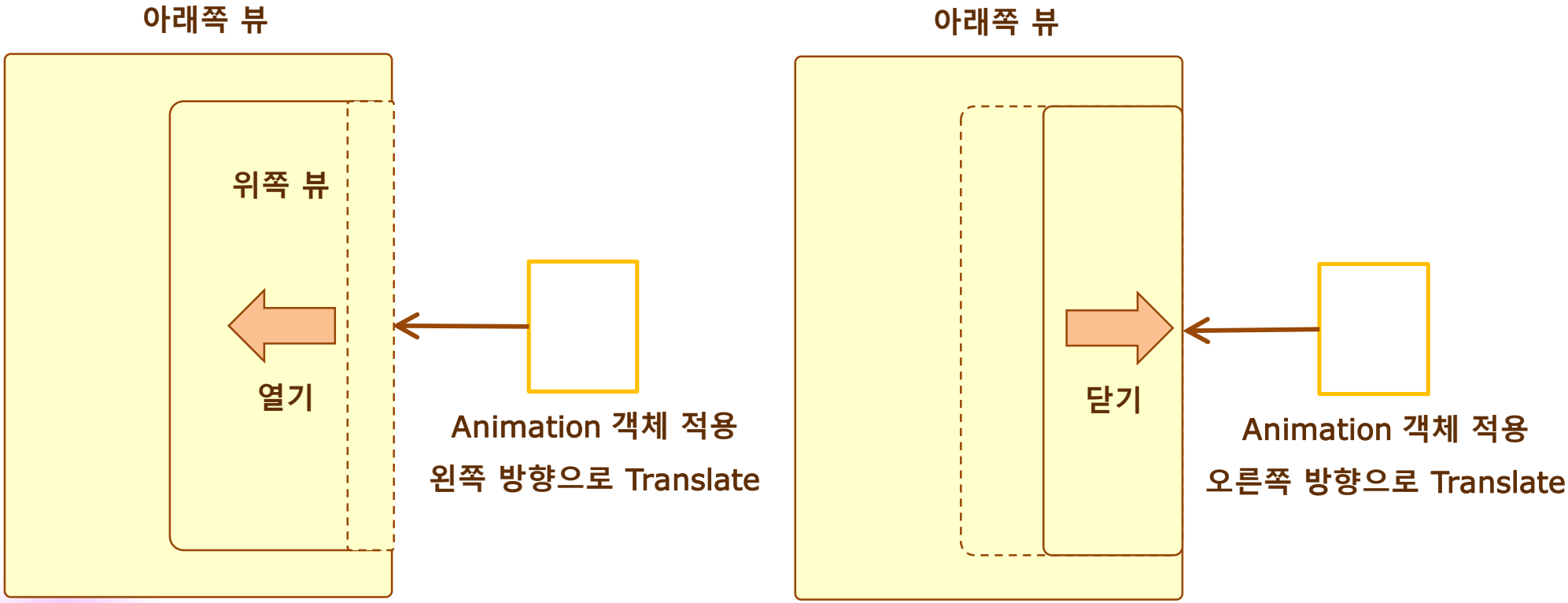
메서드	설명
public void onAnimationStart(Animation animation)	애니메이션이 시작되기 전에 호출됩니다.
public void onAnimationEnd(Animation animation)	애니메이션이 끝났을 때 호출됩니다.
public void onAnimationRepeat(Animation animation)	애니메이션이 반복될 때 호출됩니다.

2. 페이지 슬라이딩 사용하기

페이지 슬라이딩

▶ 뷰의 중첩과 애니메이션을 접목한 방식

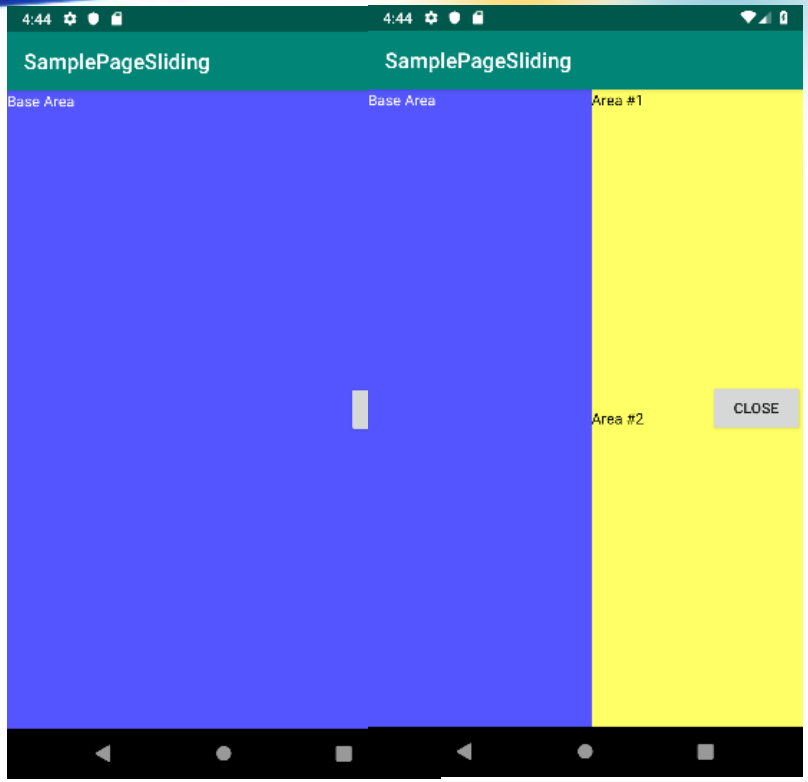
▶ 하나의 뷰 위에 다른 뷰를 올라가 있을 때 보이거나 보이지 않는 과정을 애니메이션으로 적용



페이지 슬라이딩 사용하기

페이지 슬라이딩 사용하기 예제

-페이지 슬라이딩을 이용해 뷰 보여주기



메인 액티비티의 XML 레이아웃 정의

-메인 액티비티 레이아웃 정의

메인 액티비티 코드 작성

-슬라이딩 기능 넣기

```
<LinearLayout
 android:orientation="vertical"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:background="#ff5555ff">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Base Area"
      android:textColor="#ffffffff"
      />
</LinearLayout>
```

1 첫 번째 레이아웃 : 바탕 레이아웃

Continued..

레이아웃 만들기 (계속)

```
<LinearLayout
 android:id="@+id/slidingPage01"
 android:orientation="vertical"
 android:layout_width="200dp"
 android:layout_height="match_parent"
 android:layout_gravity="right"
 android:background="#ffffff66"
 android:visibility="gone">
 <TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:layout_weight="1"
  android:text="Area #1"
  android:textColor="#ff000000"
  />
```

2 두 번째 레이아웃 :
슬라이딩으로 보일 레이아웃

Continued..

레이아웃 만들기 (계속)

```
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="Area #2"
      android:textColor="#ff000000"
      />
</LinearLayout>

<LinearLayout
 android:orientation="vertical"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_gravity="right|center_vertical"
 android:background="#00000000">

  <Button
   android:id="@+id/openBtn01"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Open"
   />
  </LinearLayout>
</FrameLayout>
```

3 세 번째 레이아웃 :
버튼이 들어 있는 레이아웃

메인 액티비티 코드 만들기

참조파일 SamplePageSliding>/app/java/org.techtown.sliding/MainActivity.java

```
public class MainActivity extends AppCompatActivity {
  boolean isPageOpen = false;

  Animation translateLeftAnim;
  Animation translateRightAnim;

  LinearLayout page;
  Button button;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    page = findViewById(R.id.page);

    translateLeftAnim = AnimationUtils.loadAnimation(this, R.anim.translate_left);
    translateRightAnim = AnimationUtils.loadAnimation(this, R.anim.translate_right);

    SlidingPageAnimationListener animListener = new SlidingPageAnimationListener();
    translateLeftAnim.setAnimationListener(animListener);
    translateRightAnim.setAnimationListener(animListener);
```

메인 액티비티 코드 만들기 (계속)

```
        button = findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (isPageOpen) {
                    page.startAnimation(translateRightAnim);
```

```
                } else {
                    page.setVisibility(View.VISIBLE);
                    page.startAnimation(translateLeftAnim);
                }
            }
        });
    }
```

메인 액티비티 코드 만들기 (계속)

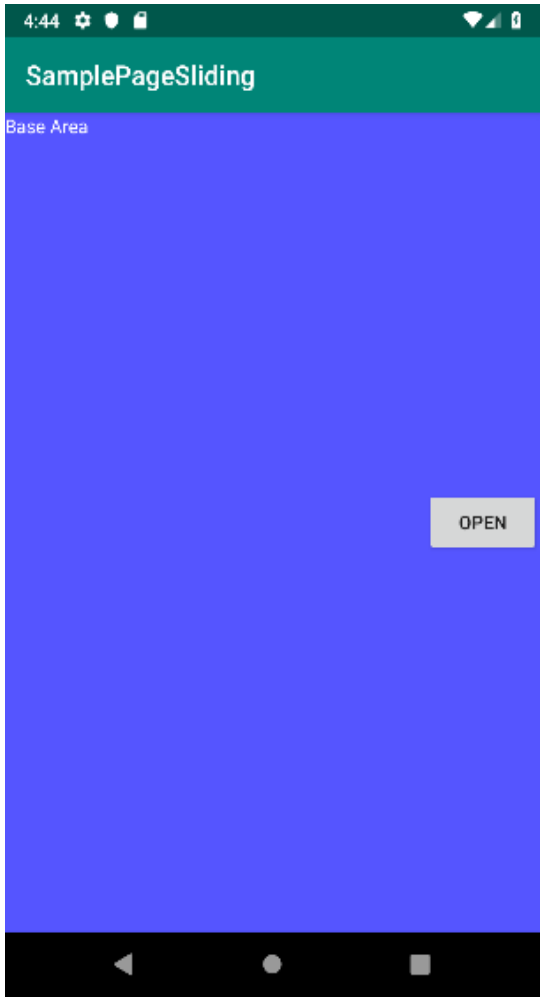
```
private class SlidingPageAnimationListener implements Animation.AnimationListener {

  public void onAnimationEnd(Animation animation) {
    if (isPageOpen) {
      page.setVisibility(View.INVISIBLE);

      button.setText("Open");
      isPageOpen = false;
    } else {
      button.setText("Close");
      isPageOpen = true;
    }
  }

  @Override
  public void onAnimationStart(Animation animation) { }

  @Override
  public void onAnimationRepeat(Animation animation) { }
}
```

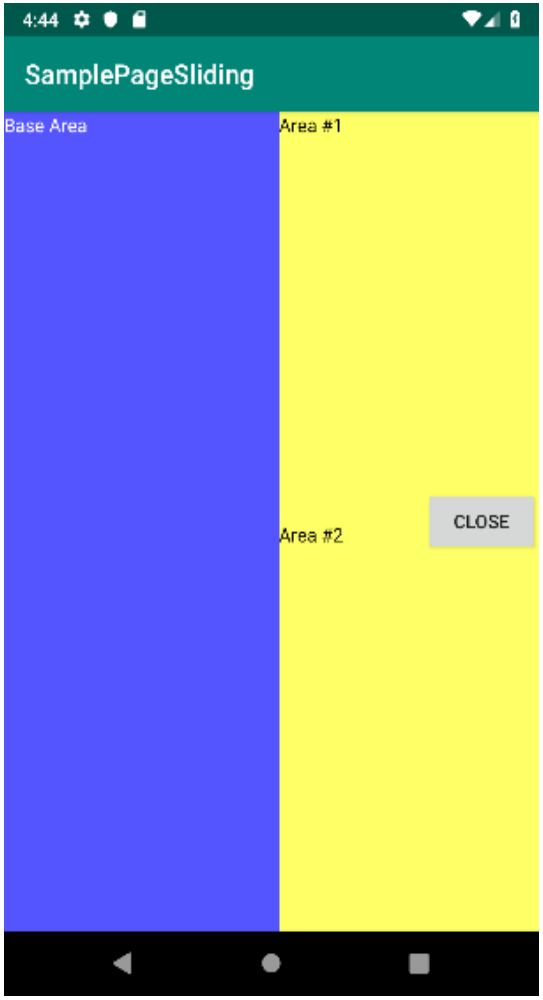
애니메이션을 위한 XML 정의

참조파일 SamplePageSliding>/app/res/anim/translate_left.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
  android:interpolator="@android:anim/accelerate_decelerate_interpolator">
  <translate
    android:fromXDelta="100%p"
    android:toXDelta="0%p"
    android:duration="500"
    android:repeatCount="0"
    android:fillAfter="true"
    />
</set>
```

실행 화면

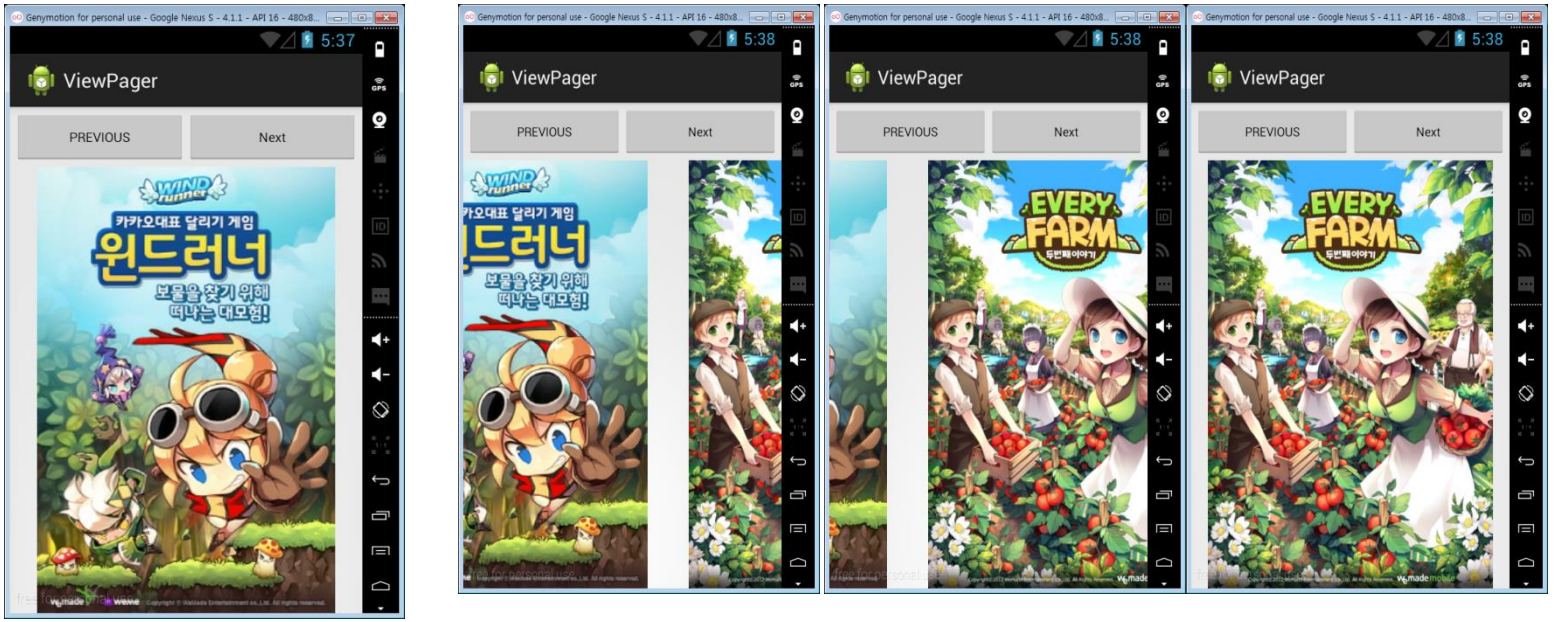

4:44
SamplePageSliding
Base Area
OPEN


4:44
SamplePageSliding
Base Area
Area #1
Area #2
CLOSE

3. 뷰페이저 사용하기

1. Viewpager 개요

- Viewpager
  - 수평으로 View를 좌/우 로 스크롤 할 때 사용 사용하는 클래스
  - View들을 Page를 넘기듯 보여주는 AdapterView의 일종



1. Viewpager 개요

- PagerAdapter의 종류 : PagerAdapter, FragmentPagerAdapter, FragmentStatePagerAdapter

- FragmentPagerAdapter
  - 화면을 슬라이딩으로 전환할 때 한 번 생성된(화면에 보인) Fragment를 계속 메모리상 가지고 있음
  - 이전 Fragment로 슬라이딩을 해서 돌아간다고 하면 이전에 생성된 Fragment로 돌아감.

- FragmentStatePagerAdapter
  - 화면이 전환될 때 이전(화면에서 보이지 않는) Fragment는 메모리상 제거(destroy) 됨
  - Adapter의 Fragment가 많거나 개수를 알 수 없을 때 메모리 관련 이슈를 위해 사용

2. PagerAdapter 사용

- 영화포스트 이미지 넘기기

  activity_main.xml

  viewpager_view.xml

  CustomAdapter.java

  MainActivity.java

Veiwpage-PagerAdapter 사용예제

activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent“>

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:layout_width="0dp"
            android:layout_weight="1"
            android:layout_height="wrap_content"
            android:text="PREVIOUS"
            android:textSize="16sp"
            android:onClick="mOnClicked"
            android:id="@+id/btnPrevious" />

        <Button
            android:layout_width="0dp"
            android:layout_weight="1"
            android:layout_height="wrap_content"
            android:text="NEXT"
            android:textSize="16sp"
            android:onClick="mOnClicked"
            android:id="@+id/btnNext" />
    </LinearLayout>
    <android.support.v4.view.ViewPager
        android:id="@+id/pager"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
</android.support.v4.view.ViewPager>
</LinearLayout>
```

Veiwpage-PagerAdapter 사용예제

viewpager_childview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/viewpagerImageView" />
</LinearLayout>
```

Adapter 클래스 작성

```
public class CustomAdapter extends PagerAdapter {
    LayoutInflater inflater;

    public CustomAdapter(LayoutInflater inflater){
        this.inflater=inflater;
    }
    @Override
    public int getCount() {
        return 10;
    }

    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        View view=inflater.inflate(R.layout.viewpager_childview, null);
        ImageView img=(ImageView)view.findViewById(R.id.viewpagerImageView);
        img.setImageResource(R.drawable.mov01+position);
        container.addView(view);
        return view;
    }
... 계속
```

Adapter 클래스 작성 계속

```
    .....
    @Override
     public void destroyItem(ViewGroup container, int position, Object object) {
         container.removeView((View)object);
     }

     @Override
     public boolean isViewFromObject(View view, Object object) {
         return view==object;
     }
}
```

Veiwpage-PagerAdapter 사용예제

MainActivity 클래스 작성

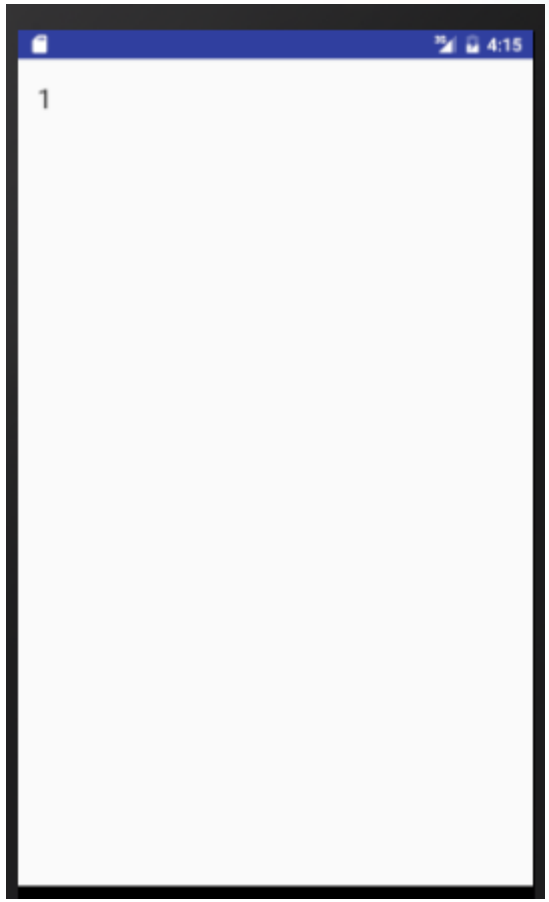
```
public class MainActivity extends AppCompatActivity {
    ViewPager pager;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        pager=(ViewPager)findViewById(R.id.pager);
        CustomAdapter adapter=new CustomAdapter(getLayoutInflater());
        pager.setAdapter(adapter);
    }

    public void mOnClicked(View view){
        int position;
        switch (view.getId()){
            case R.id.btnPrevious:
                position=pager.getCurrentItem();
                pager.setCurrentItem(position-1,true);
                break;
            case R.id.btnNext:
                position=pager.getCurrentItem();
                pager.setCurrentItem(position+1,true);
                break;
        }
    }
}
```

3. FragmentPagerStateAdapter 사용

Fragment를 사용하여 페이지 번호 표시

구성 파일

- activity_main.xml
- Fragment_page.xml
- MainActivity
- MyPagerAdapter
- PageFragment



FragmentPagerStateAdapter 사용

Activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.nt500.myapplication.MainActivity">

    <android.support.v4.view.ViewPager
        android:id="@+id/pager"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

    </android.support.v4.view.ViewPager>

</RelativeLayout>
```

FragmentPagerStateAdapter 사용

fragment_page.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Large Text"
        android:id="@+id/number" />
</LinearLayout>
```

PageFragment 클래스

```
public class PageFragment extends Fragment {
    private int mPageNumber;
    public static PageFragment create(int mPageNumber){
        PageFragment fragment=new PageFragment();
        Bundle args=new Bundle();
        args.putInt("page", mPageNumber);
        fragment.setArguments(args);
        return fragment;
    }
    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mPageNumber=getArguments().getInt("page");
    }
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
                @Nullable Bundle savedInstanceState) {
        ViewGroup rootView=(ViewGroup)inflater.inflate(R.layout.fragment_page, container, false);
        ((TextView)rootView.findViewById(R.id.number)).setText(mPageNumber+"");
        return rootView;
    }
}
```

FragmentPagerStateAdapter 사용

MyPagerAdapter 클래스 작성

```
public class MyPagerAdapter extends FragmentStatePagerAdapter {
    public MyPagerAdapter(FragmentManager fm) {
        super(fm);
    }

    @Override
    public Fragment getItem(int position) {
        return PageFragment.create(position) ;
    }

    @Override
    public int getCount() {
        return 5;
    }
}
```

FragmentPagerStateAdapter 사용

MainActivity 클래스

```
public class MainActivity extends FragmentActivity {
    private ViewPager mViewPager;
    private PagerAdapter mPagerAdapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mViewPager=(ViewPager)findViewById(R.id.pager);
        mPagerAdapter=new MyPagerAdapter(getSupportFragmentManager());
        mViewPager.setAdapter(mPagerAdapter);

    }
}
```