자바응용SW(앱)개발자양성과정

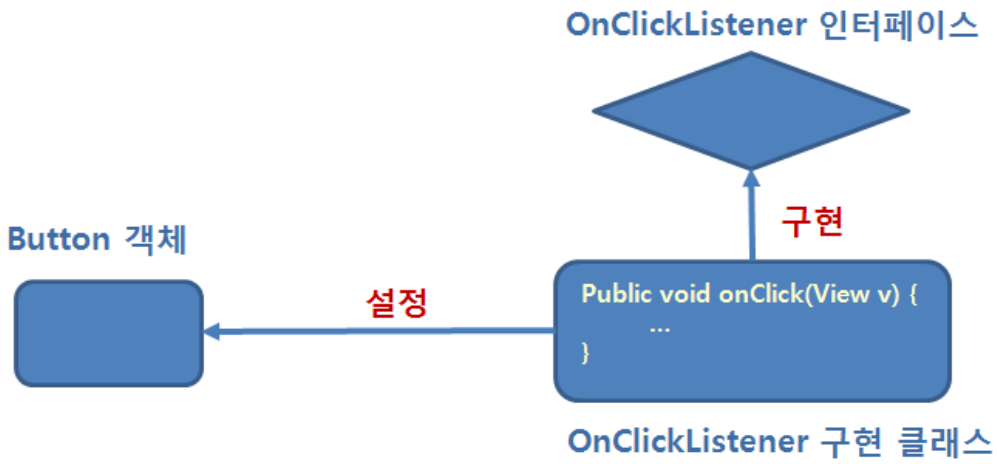
Chapter 07

사용자 이벤트 처리와 리소스 활용

백제직업전문학교

김영준 강사

뷰의 이벤트 처리하기



[버튼에 OnClickListener를 설정할 때의 패턴]

뷰를 상속할 때 이벤트 처리를 위한 메소드 재정의

boolean onTouchEvent (MotionEvent event)

boolean onKeyDown (int keyCode, KeyEvent event)

boolean onKeyUp (int keyCode, KeyEvent event)

뷰 객체에 전달되는 이벤트를 처리하기 위한 리스너 설정

View.OnTouchListener : boolean onTouch (View v, MotionEvent event)

View.OnKeyListener : boolean onKey (View v, int keyCode, KeyEvent event)

View.OnClickListener : void onClick (View v)

View.OnFocusChangeListener : void onFocusChange (View v, boolean hasFocus)

대표적인 이벤트

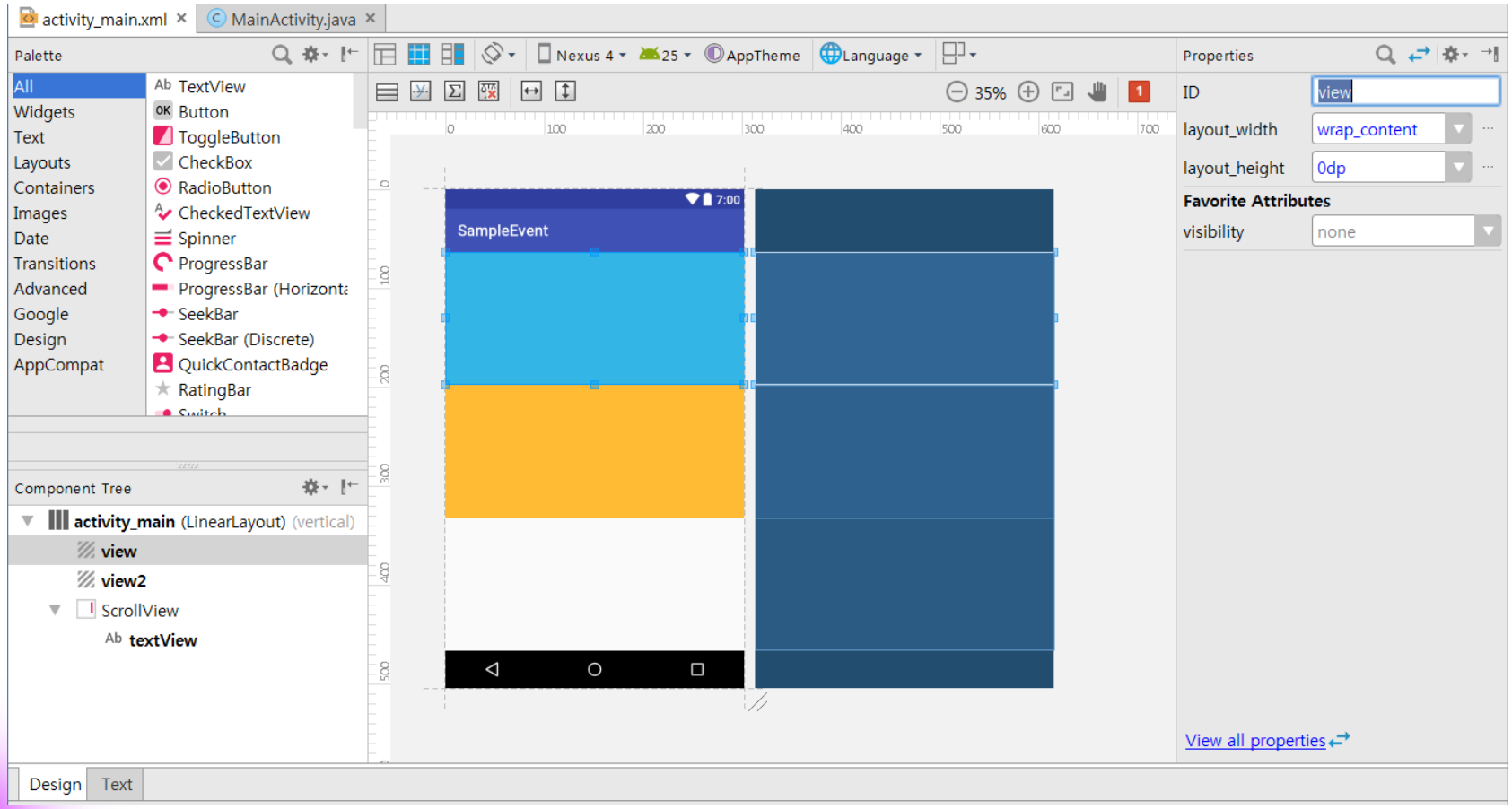
- **터치 이벤트**
  - 화면을 손가락으로 누를 때 발생하는 이벤트
- **키 이벤트**
  - 키패드나 하드웨어 버튼을 누를 때 발생하는 이벤트
- **제스처 이벤트**
  - 터치 이벤트 중에서 일정 패턴을 만들어 내는 이벤트
- **포커스**
  - 뷰마다 순서대로 주어지는 포커스
- **화면 방향 변경**
  - 화면의 방향이 가로/세로로 바뀜에 따라 발생하는 이벤트

제스처를 통해 처리할 수 있는 이벤트

메소드	이벤트 유형
onDown()	- 화면이 눌렸을 경우
onShowPress()	- 화면이 눌렸다 떼어지는 경우
onSingleTapUp()	- 화면이 한 손가락으로 눌렸다 떼어지는 경우
onSingleTapConfirmed()	- 화면이 한 손가락으로 눌려지는 경우
onDoubleTap()	- 화면이 두 손가락으로 눌려지는 경우
onDoubleTapEvent()	- 화면이 두 손가락으로 눌려진 상태에서 떼거나 이동하는 등 세부적인 액션을 취하는 경우
onScroll()	- 화면이 눌린 채 일정한 속도화 방향으로 움직였다 떼는 경우
onFling()	- 화면이 눌린 채 가속도를 붙여 손가락을 움직였다 떼는 경우
onLongPress()	- 화면을 손가락으로 오래 누르는 경우

터치 이벤트 처리하기

- SampleEvent 프로젝트를 만들고 XML 레이아웃 구성



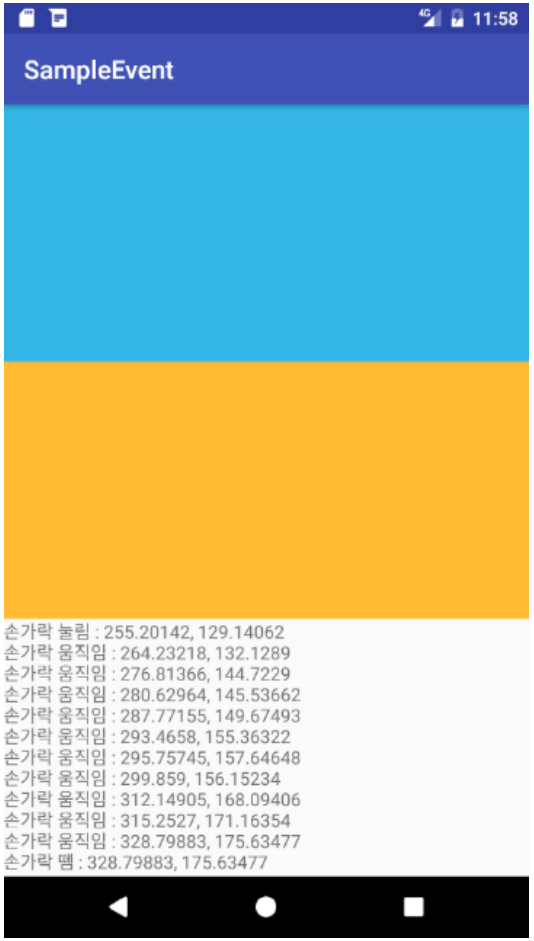
메인 액티비티 코드 만들기

```
View view = findViewById(R.id.view);
view.setOnTouchListener(new View.OnTouchListener() {
    @Override
    public boolean onTouch(View view, MotionEvent motionEvent) {
        int action = motionEvent.getAction();
        float curX = motionEvent.getX();
        float curY = motionEvent.getY();

        if (action == MotionEvent.ACTION_DOWN) {
            println("손가락 눌림 : " + curX + ", " + curY);
        } else if (action == MotionEvent.ACTION_MOVE) {
            println("손가락 움직임 : " + curX + ", " + curY);
        } else if (action == MotionEvent.ACTION_UP) {
            println("손가락 뗌 : " + curX + ", " + curY);
        }

        return true;
    }
});
```

Continued..

앱 실행 결과

- 가장 위쪽에 있는 부분을 터치했을 때 가장 아래쪽에 보이는 로그 메시지 확인



제스처 이벤트 처리하기

- GestureDetector 객체 만들고 터치 이벤트 발생 시 해당 객체 전달

```
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX, float velocityY) {
    println("₩nonFling ₩n₩tx = " + velocityX + "₩n₩ty=" + velocityY);
    return super.onFling(e1, e2, velocityX, velocityY);
}

public boolean onScroll(MotionEvent e1, MotionEvent e2, float distanceX, float distanceY) {
    println("₩nonScroll ₩n₩tx = " + distanceX + "₩n₩ty = " + distanceY);
    return super.onScroll(e1, e2, distanceX, distanceY);
}
```

Continued..

앱 실행 결과

- 중간 부분을 터치했을 때 가장 아래쪽에 보이는 로그 메시지 확인

키 입력 이벤트 처리하기

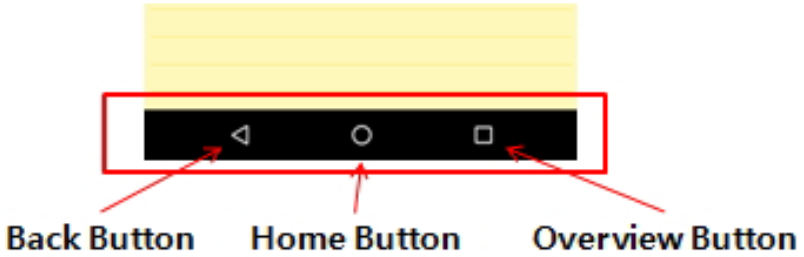
뷰를 상속할 때 키 이벤트 처리를 위한 메소드 재정의

```
boolean onKeyDown (int keyCode, KeyEvent event)
boolean onKey (View v, int keyCode, KeyEvent event)
```

[키를 눌렀을 때 전달되는 대표적인 키값]

키 코드	설 명
KEYCODE_DPAD_LEFT	- 왼쪽 화살표
KEYCODE_DPAD_RIGHT	- 오른쪽 화살표
KEYCODE_DPAD_UP	- 위쪽 화살표
KEYCODE_DPAD_DOWN	- 아래쪽 화살표
KEYCODE_DPAD_CENTER	- [중앙] 버튼
KEYCODE_CALL	- [통화] 버튼
KEYCODE_ENDCALL	- [통화 종료] 버튼
KEYCODE_HOME	- [홈] 버튼
KEYCODE_BACK	- [뒤로 가기] 버튼
KEYCODE_VOLUME_UP	- [소리 크기 증가] 버튼
KEYCODE_VOLUME_DOWN	- [소리 크기 감소] 버튼
KEYCODE_0 ~ KEYCODE_9	- 숫자 0부터 9까지의 키값
KEYCODE_A ~ KEYCODE_Z	- 알파벳 A부터 Z까지의 키값

키 입력 이벤트 처리하기



- onKeyDown: 키가 눌린 순간의 이벤트
- onKeyUp: 키를 떼는 순간의 이벤트
- onKeyLongPress: 키를 오래 누르는 순간의 이벤트

```
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if(keyCode==KeyEvent.KEYCODE_BACK){

    }
    return super.onKeyDown(keyCode, event);
}
```

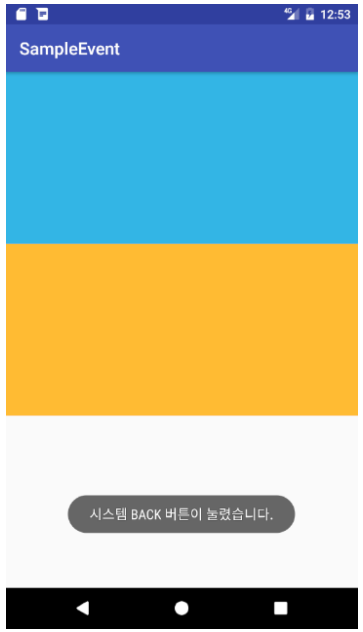
```
public void onBackPressed() {
    super.onBackPressed();
}
```

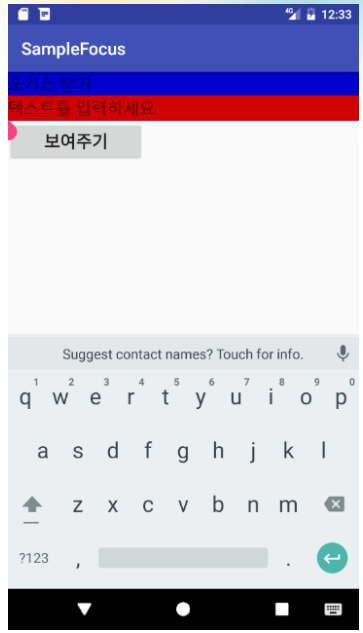
BACK 버튼과 포커스 이벤트 처리 예제

BACK 버튼과 포커스 예제

-BACK 버튼을 눌렀을 때의 이벤트 처리

-입력상자가 포커스를 받았을 때의 이벤트 처리





메인 액티비티의 코드 수정

-BACK 버튼 이벤트 처리하도록 수정

포커스 처리 XML로 정의

-포커스 처리를 위한 XML 정의

EditText의 속성 설정

-입력상자의 속성으로 설정

BACK 버튼 처리를 위한 액티비티 코드 만들기

```
public boolean onKeyDown(int keyCode, KeyEvent event) {
 if(keyCode == KeyEvent.KEYCODE_BACK) {
    close();
    return true;
  }
 return false;
}

private void close() {
  Intent resultIntent = new Intent();
  resultIntent.putExtra("name", "mike");
  setResult(1, resultIntent);
  finish();
 }
}
```

1 하드웨어 [BACK] 버튼이 눌렸을 경우 새로 정의한 close() 메소드 호출

2 호출한 액티비티로 결과값 전송

3 액티비티 없애기

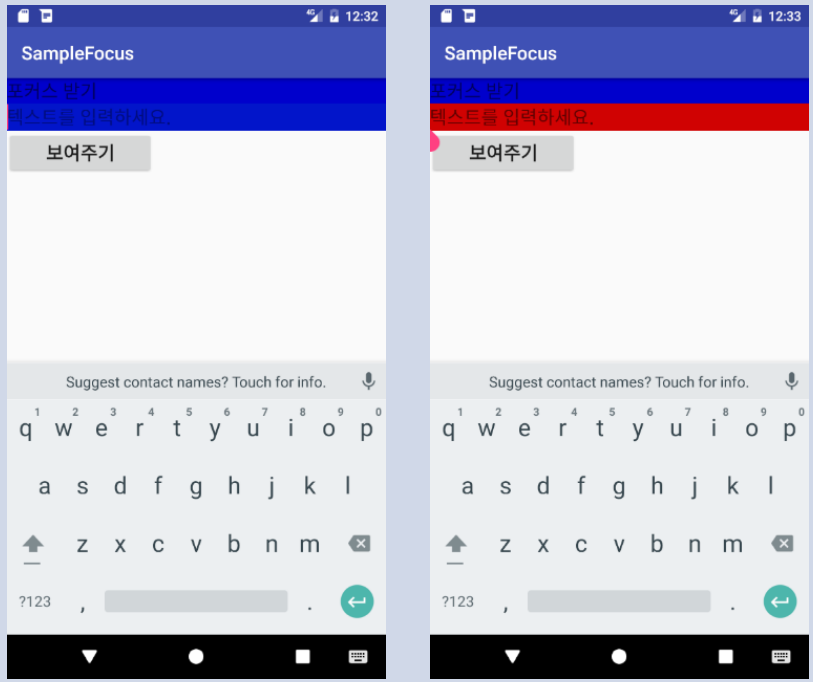
포커스 이벤트 처리를 위한 XML 레이아웃

```
<EditText
 android:id="@+id/editText"
 Android:layout_width="match_parent"
 Android:layout_height="wrap_content"
 Android:hint="텍스트를 입력하세요."
 Android:textSize="20dp"
 Android:background="@drawable/button_selector"
 />

<Button
 Android:id="@+id/showButton"
 android:layout_width="160dp"
 android:layout_height="wrap_content"
 android:text="보여주기"
 android:textSize="20dp"
 android:textStyle="bold"
 />
</LinearLayout>
```

1 입력 상자 정의

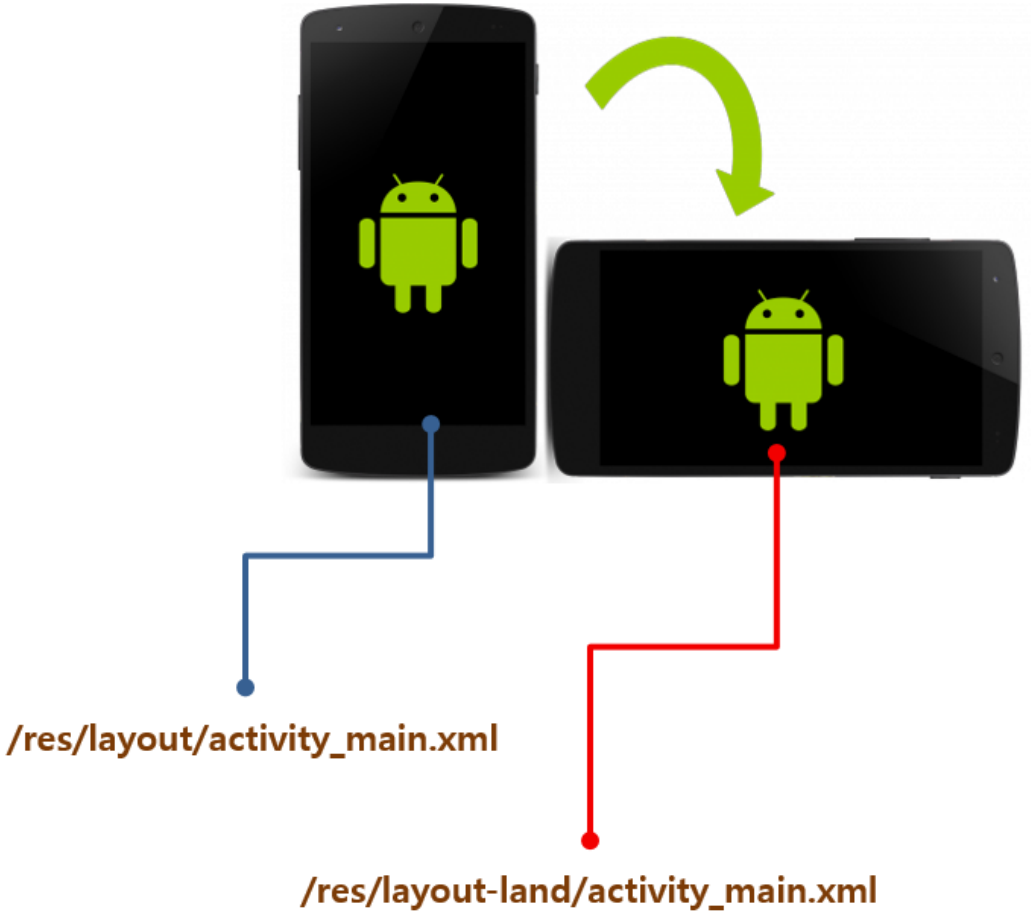
포커스 처리를 위한 XML 정의

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
  <item
   android:state_focused="true"
   android:state_pressed="true"
   android:drawable="@drawable/red"
   />
  <item
   android:state_focused="false"
   android:state_pressed="true"
   android:drawable="@drawable/green"
   />
  <item
   android:drawable="@drawable/blue"
   />
</selector>
```



단말 방향 전환

- 병렬 리소스 로딩 방식 사용
- [res] 폴더 안에 [layout] 폴더와 [layout-land] 폴더 생성

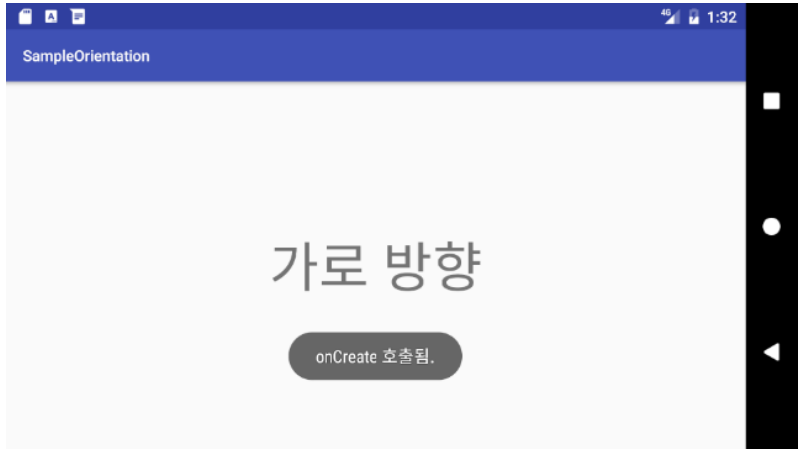


소스 코드에서 토스트 메시지 표시

```
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  showToast("onCreate 호출됨.");
}

@Override
protected void onStart() {
  super.onStart();
  showToast("onStart 호출됨.");
}

@Override
protected void onStop() {
  super.onStop();
  showToast("onStop 호출됨.");
}
```

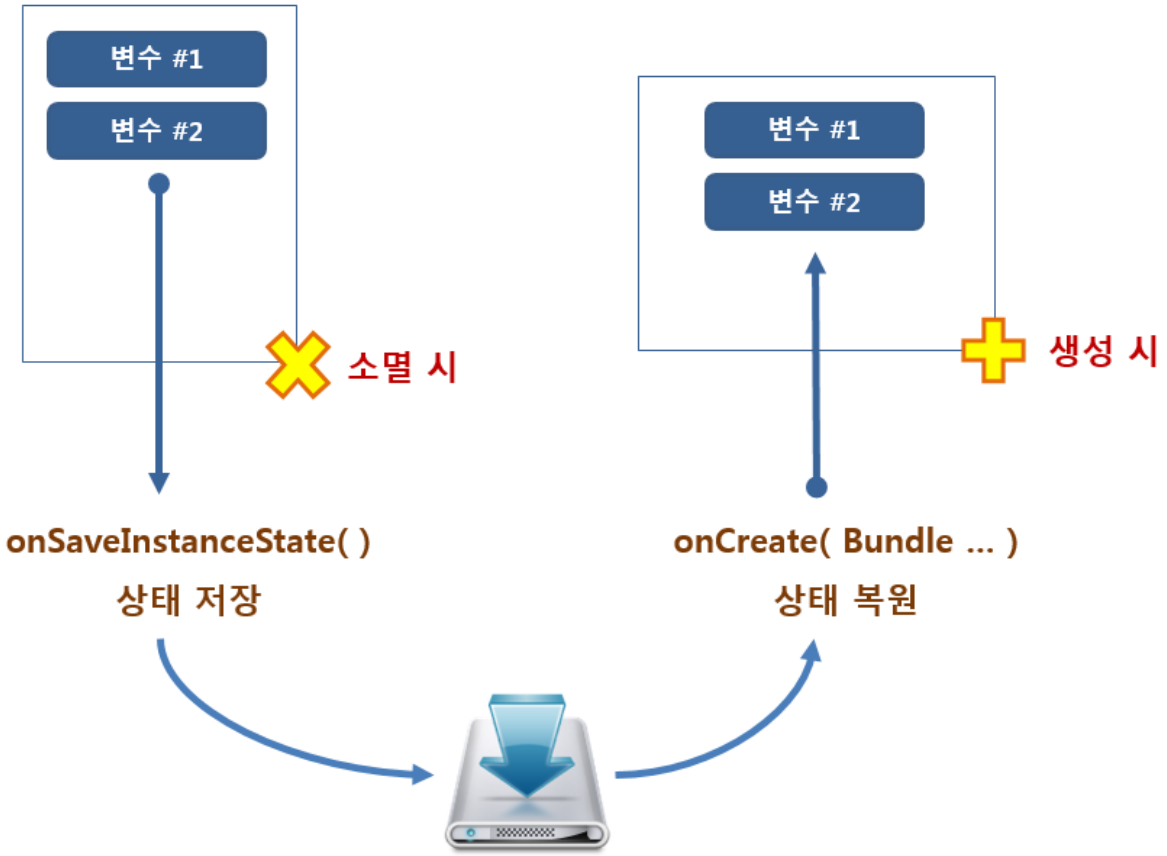
앱 실행 결과

- 단말 방향 바꾸었을 때 액티비티가 새로 생성되므로 토스트 메시지 표시됨



단말 방향 전환 시 상태 저장과 복원

- onSaveInstanceState 메소드에서 상태 저장했다가 onCreate의 파라미터를 이용해 복원



단말 방향전환 상태저장 예제

-단말 방향이 가로와 세로로 바뀌었을 때 상태 저장과 복원

매니페스트 속성 추가

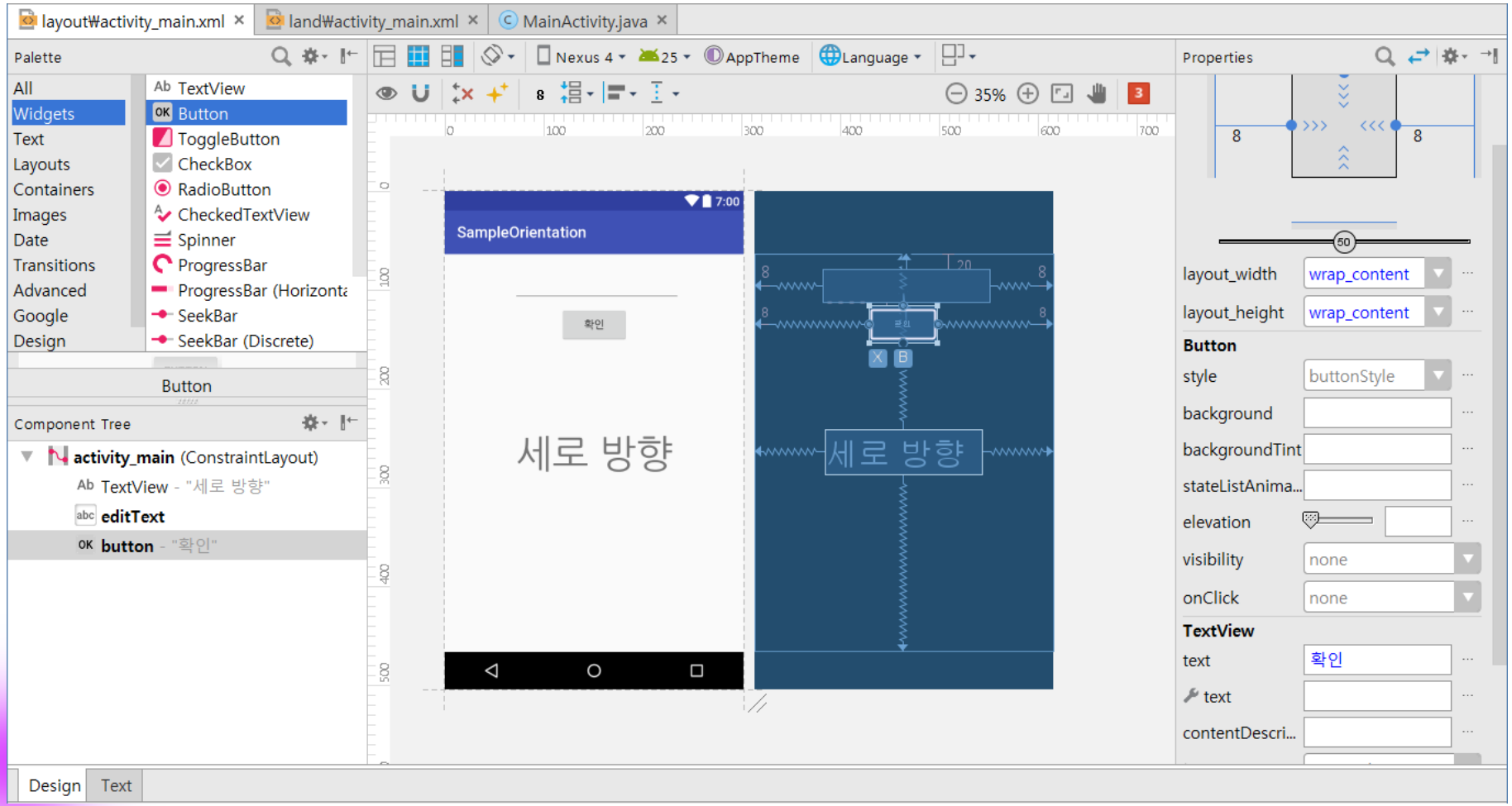
-매니페스트의 액티비티 속성 추가

메인 액티비티 코드 작성

-가로와 세로 방향으로 바뀌었을 때 처리 코드 작성

XML 레이아웃 구성

- 입력상자와 버튼 추가

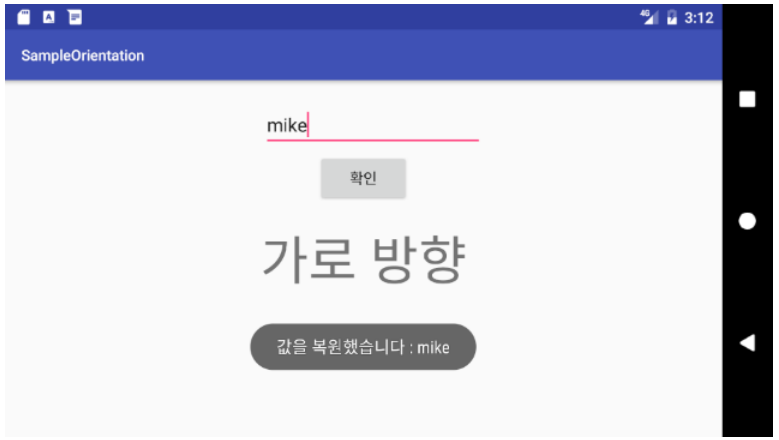


소스 코드에서 값의 저장과 복원

```
protected void onCreate(Bundle savedInstanceState) {
...
  if (savedInstanceState != null) {
    name = savedInstanceState.getString("name");

    Toast.makeText(getApplicationContext(), "값을 복원했습니다 : " + name, Toast.LENGTH_LONG).show();
  }
}

@Override
protected void onSaveInstanceState(Bundle outState) {
  super.onSaveInstanceState(outState);

  outState.putString("name", name);
}
...
```

앱 실행 결과

- 단말 방향 바꾸었을 때 값 저장되었다가 복원됨



단말 방향 전환 시 액티비티 유지

- AndroidManifest.xml 파일에 configChanges 속성 설정

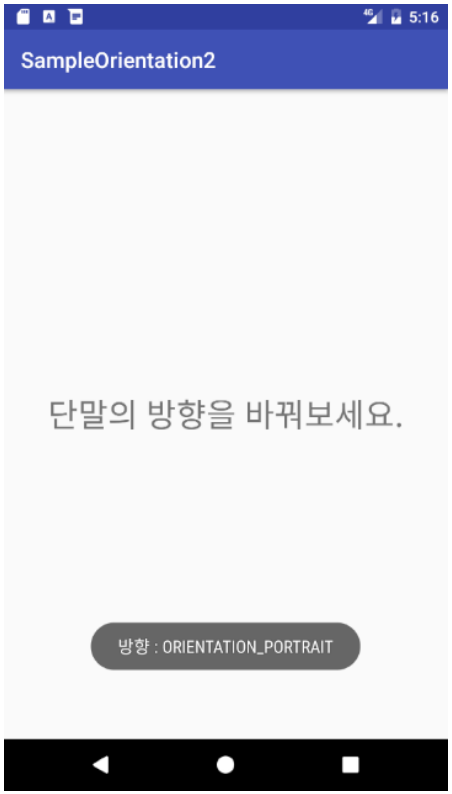
```
...
<activity android:name=".MainActivity"
    android:configChanges="orientation|screenSize|keyboardHidden"
    >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
...
```

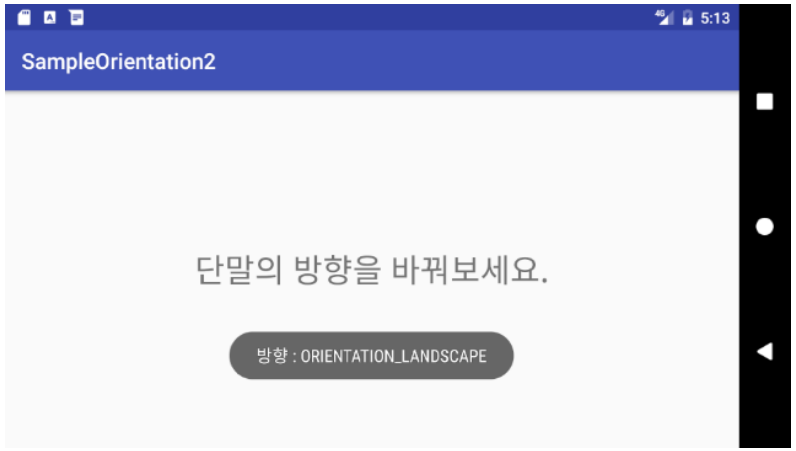
소스 코드에서 방향 전환 이벤트 전달받음

```
public class MainActivity extends AppCompatActivity {
...
    public void onConfigurationChanged(Configuration newConfig) {
        super.onConfigurationChanged(newConfig);

        if (newConfig.orientation == Configuration.ORIENTATION_LANDSCAPE) {
            showToast("방향 : ORIENTATION_LANDSCAPE");
        } else if (newConfig.orientation == Configuration.ORIENTATION_PORTRAIT) {
            showToast("Orientation : ORIENTATION_PORTRAIT");
        }
    }
}

...
```

앱 실행 결과

- 단말 방향 바꾸었을 때 값 액티비티는 유지되고 이벤트 전달받음





1.1. 리소스 종류

- drawable: 이미지, 이미지와 관련된 XML, 그림을 표현한 XML
- layout: 화면 UI를 정의한 레이아웃 XML
- values: 문자열, 색상, 크기 등 여러 가지 값
- menu: 액티비티의 메뉴를 구성하기 위한 XML
- xml: 특정 폴더가 지정되어 있지 않은 기타 XML
- anim: 애니메이션을 위한 XML
- raw: 바이트 단위로 직접 이용되는 이진 파일
- mipmap: 앱 아이콘 이미지

1.2. 다양한 리소스 활용

애니메이션 리소스

- XML 파일이 위치하는 폴더는 res 하위에 anim이라는 폴더

```
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate/>
    <rotate/>
    <alpha/>
    <scale/>
</set>
```

- scale: 크기 변경 애니메이션, 크기 확대/축소
- rotate: 회전 애니메이션
- alpha: 투명도 조정 애니메이션
- translate: 이동 애니메이션

```
<set
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="true">
    <scale
        android:fromXScale="0.0"
        android:toXScale="1.0"
        android:fromYScale="0.0"
        android:toYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:startOffset="0"
        android:duration="2000"
        />
    <alpha
        android:fromAlpha="0.0"
        android:toAlpha="1.0"
        android:startOffset="0"
        android:duration="2000"
        />
</set>
```

- duration: 지속시간
- startOffset: 애니메이션을 시작한 후 얼마 후부터 애니메이션 효과를 적용
- repeatCount: 애니메이션 반복 횟수.
- repeatMode: 애니메이션 반복 시의 방향

```
Animation anim = AnimationUtils.loadAnimation(this, R.anim.in);
imageView.startAnimation(anim);
```

- 애니메이션 이벤트

```
anim.setAnimationListener(new Animation.AnimationListener() {
    @Override
    public void onAnimationStart(Animation animation) {

    }

    @Override
    public void onAnimationEnd(Animation animation) {
    }

    @Override
    public void onAnimationRepeat(Animation animation) {

    }
});
```

크기, 색상 리소스

- 리소스 중 문자열, 배열, 색상, 크기 등 흔히 값이라고 표현되는 리소스는 values 폴더 하 위에 위치

- strings.xml: 문자열 리소스
- colors.xml: 색상 리소스
- styles.xml: 스타일
- arrays.xml: 배열 리소스
- dimens.xml: 크기 리소스

```
<resources>
    <dimen name="my_margin">16dp</dimen>
    <dimen name="my_padding">16dp</dimen>
</resources>
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button1"
    android:padding="@dimen/my_padding"
    android:layout_margin="@dimen/my_margin"
    />
```

```
<resources>
    <color name="my_background">#FFFF0000</color>
    <color name="my_textColor">#FF00FFFF</color>
</resources>
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Button1"
    android:background="@color/my_background"
    android:textColor="@color/my_textColor"
    />
```

스타일 리소스

- 스타일 리소스는 여러 속성을 하나의 스타일로 묶어 필요한 곳에 적용하기 위해 사용

```
<style name="myStyle">
    <item name="android:textColor">#FF0000FF</item>
    <item name="android:textSize">20dp</item>
    <item name="android:textStyle">bold</item>
</style>
```

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="First"
    style="@style/myStyle"
    />
```

- 스타일을 정의할 때 다른 스타일을 상속받아 재정의

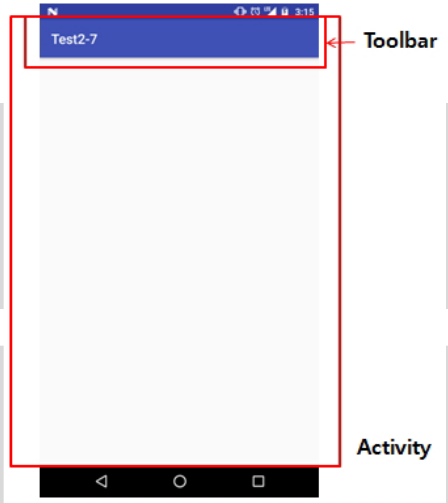
```
<style name="myStyle">
    <item name="android:textColor">#FF0000FF</item>
    <item name="android:textSize">20dp</item>
    <item name="android:textStyle">bold</item>
</style>

<style name="mySubStyle" parent="myStyle">
    <item name="android:textStyle">italic</item>
</style>
```

테마 리소스

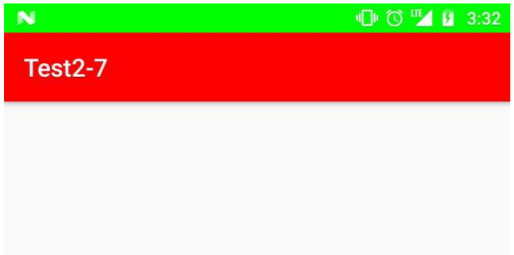
- 액티비티 전체 혹은 앱 전체를 위한 스타일

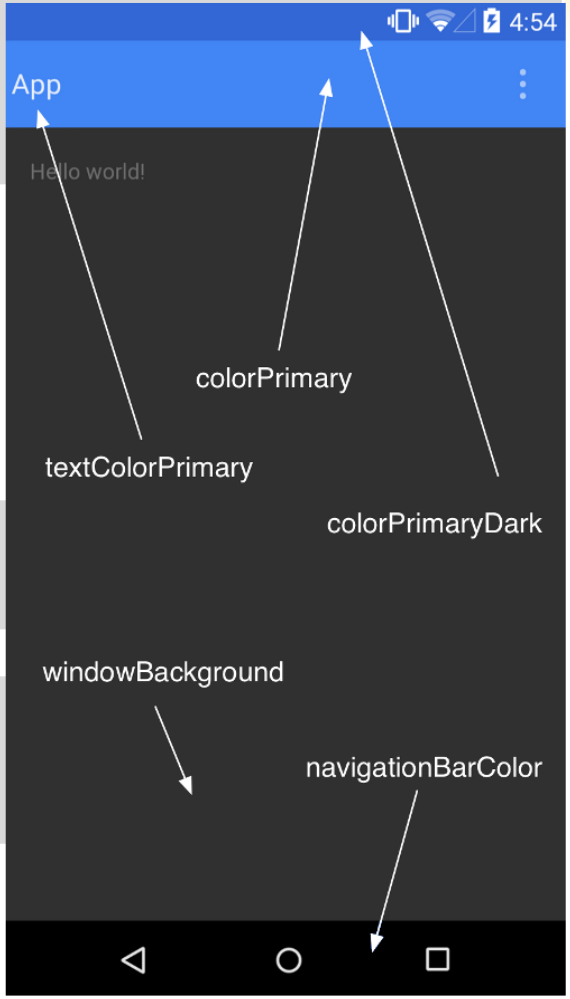
```
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>
```

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
```



```
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="colorPrimary">#FF0000</item>
    <item name="colorPrimaryDark">#00FF00</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>
```



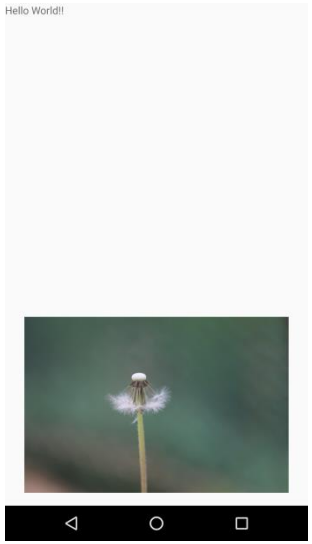



```
<activity android:name=".MainActivity" android:theme="@style/AppTheme">

</activity>
```

```
<style name="myTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="windowNoTitle">true</item>
    <item name="windowActionBar">false</item>
</style>
```

animation resource 활용과 Activity의 Theme 설정 하는 부분에 대한 실습

1. Module 생성
2. 파일 복사
3. anim 폴더 생성
4. animation xml 파일 생성
5. in.xml 파일 작성
6. move.xml 파일 작성
7. styles.xml 파일 추가
8. AndroidManifest.xml 에 style 설정
9. activity_main.xml 파일 추가
10. MainActivity.java 작성
11. 실행



Facebook의 Messenger Intro화면구성

Facebook의 Messenger에서 제공하는 Intro 화면과 동일한 화면을 몇가지 요구사항에 맞춰 작성

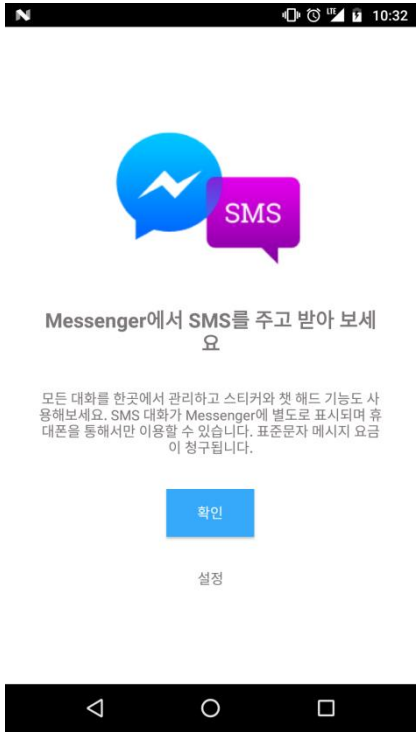
필요기술

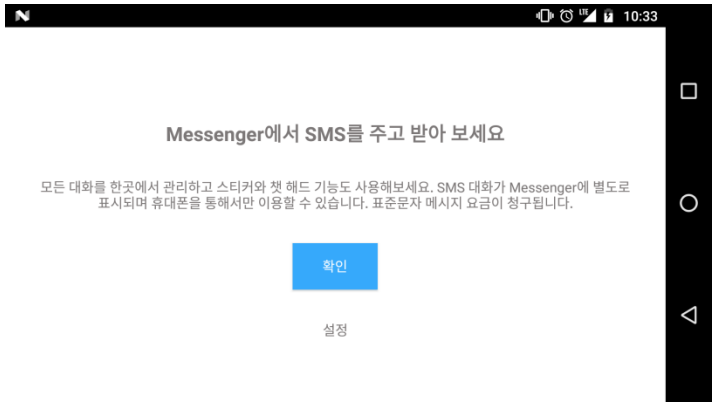
- Layout 구성
- 화면회전에 따른 UI 교체
- 문자열 리소스 국제화
- 버튼 클릭 이벤트
- Toast 출력

Facebook의 Messenger Intro화면구성

폰의 가로 세로 방향 회전에 대응하는 App을 작성

•세로방향은 모든 내용을 출력, 가로 방향에서는 출력 안함.
•세로방향과 가로방향을 위한 각각의 layout xml 파일을 작성하여 대응한다





Facebook의 Messenger Intro화면구성

국제화에 대응하는 App을 작성

•폰의 로케일이 ko인경우 한국어로 문자열을 출력하며 en인경우 영어로 문자열을 출력한다