

## Handout sheet 2: 사용자 인터페이스

Subject : Mobile Software

prepared by Professor Youn-Sik Hong

All rights reserved 2012, 2014, 2015, 2016, 2017, 2019

(※ 이 자료의 무단 전재, 수정, 배포는 허용하지 않음.)

### ◎ 실습 목적

- ☐ Layout Editor를 사용한 정적 UI 디자인 기초 이해
- ☐ Kotlin 소스 코드를 사용한 동적 UI 디자인 기초 이해

### ◎ 실습 준비

- ☐ <http://cyber.inu.ac.kr> 에서 실습2-Handout.pdf 다운로드


### ◎ 실습 내용

- ☐ STEP 1: 프로젝트 생성
- ☐ STEP 2: UI 디자인
- ☐ STEP 3: layout XML 파일 내용 살펴보기
- ☐ STEP 4: 소스 코드에서 UI 생성 - ViewGroup과View 객체 생성(1/2)
- ☐ STEP 5: 소스 코드에서 UI 생성 - ViewGroup과View 객체 생성(2/2)
- ☐ STEP 6: Button에 대한 constraint 속성 설정
- ☐ STEP 7: EditText 추가 및 constraint 속성 설정
- ☐ STEP 8: EditText 객체의 너비(width) 값 지정

### ◎ 유의 사항

- ☐ Q1, Q2, ... 로 표시된 내용은 실습과제임. - **질문은 모두 12개**
- ☐ 실습과제 내용을 작성할 때
  - 답안 작성은 한글(hwp), 워드(doc), 텍스트(txt) 등 본인이 선호하는 편집기 중 선택.
  - **파일 이름**은 제출자를 확인할 수 있도록 **학번을 사용**할 것.
  - 답을 작성할 때 질문 번호(Q1)만 쓰고, **질문 내용은 쓸 필요 없음.**
  - 답안에 불필요한 내용은 포함하지 말 것(예: 불필요한 코드 캡처 이미지)
- ☐ 평가 기준 : ASAP(as soon as possible) + 정확도.
- ☐ 다른 학생 작성 내용을 copy한 경우 모두 0점 처리.
- ☐ **제출 마감 시간이 지나면 업로드 불가.** 마감 시간 경과 후 제출은 감점 있음.

## □ STEP 1: 프로젝트 생성

1-1. Android studio 아이콘  클릭

1-2. Welcome screen > **Start a new Android Studio project** 클릭

1-3. Choose your project 대화 창

Phone and Tablet > **Empty Activity** 선택 > Next 클릭

1-4. Configure your project 대화 창

Name: **MyLayout-학번 (반드시 자신의 학번을 추가)**

Package name: **edu.ourincheon.mylayout-학번** (자동 생성. 변경하지 말 것)

Save location (변경하지 말고 그대로 둬. 단, 프로젝트 저장 위치는 확인할 것!)

Language: **Kotlin** (변경하지 말고 그대로 둬)

Minimum API level : **API 24: Android 7.0 (Nougat)**

> **Finish** 클릭

### 잠깐 !

따로 언급하지 않는 한, Project 창은 android 모드로 설정.

1-5. **AndroidManifest.xml** 더블 클릭

Q1: **android:theme** 속성 값(쌍 따옴표로 묶인 문자열)을 Ctrl 키를 누른 상태에서 마우스를 클릭하면 내용이 바뀐다. 이 속성은 어느 파일에 정의되어 있는가?(파일 이름을 적을 것)

Q2: Q1에서 찾은 파일에서 아래 속성(DartActionBar)을 찾으시오.

parent= **"Theme.AppCompat.Light.DarkActionBar"**

이 속성을 아래와 같이 바꾸면(NoActionBar) 어떻게 달라질까?

parent= **"Theme.AppCompat.Light.NoActionBar"**

(각각 AVD 실행 결과를 캡처 -> 2개의 캡처 이미지)

참고 : UTF-8 - 문자 인코딩(encoding) 방식. 즉 컴퓨터에 문자를 저장하기 위한 코드 체계.

Universal Coded Character Set + Transformation Format - 8-bit 의 약자.

유니코드(unicode)를 위한 가변 길이 문자 인코딩 방식(<http://www.unicode.org>).

유니코드 한 문자를 나타내기 위해서는 1바이트에서 최대 4바이트까지 사용.

영어 알파벳, 숫자 등 **U+0000~U+007F** (0~255) 범위에 속한 ASCII 문자는 1바이트로 표시.

한글, 일본어, 중국어 등은 유니코드에서 3바이트로 표시됨.


(euc-kr 또는 ksc\_c\_5601-1987 방식에서 한글은 2바이트로 표시)

## □ STEP 2: UI 디자인

2-1. app > res > layout > **activity\_main.xml** 더블 클릭

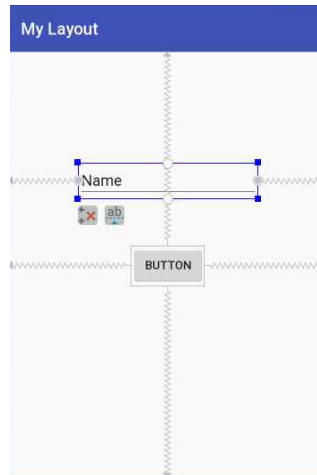
2-2. Design 탭 클릭

TextView 위젯(“Hello world!” 문자열) 클릭 > Delete 키를 눌러 삭제

2-3. Autoconnect 가 turn on (  ) 상태인지 확인

2-4. Palette 창 > **Common** > **Button** → 화면 정 중앙에 배치(id= “**button**” )

2-5. Palette 창 > **Text** > **Plain Text** → Button 객체 위에 배치(id= “**editText**” )



2-6. Plain Text 클릭

> Attribute 창 > inputType 속성 클릭

> drop down list에서 **textPersonName** 체크 상자의 체크 표시 없앰

> “**text**” 체크 상자 클릭

> Apply

> width 속성 > “**350dp**” > Enter

※ “**layout\_width**” 가 아님. 그냥 **width** 임.

Q3: 속성 **width**와 **layout\_width** 속성의 차이점을 설명하시오.

2-7. Button 객체 클릭

> Attribute 창 > text 속성 > 오른쪽 직사각형 클릭

> Add new resource > New string Value

> Resource name : **button\_string**,

Resource value : **Press Me!** > OK

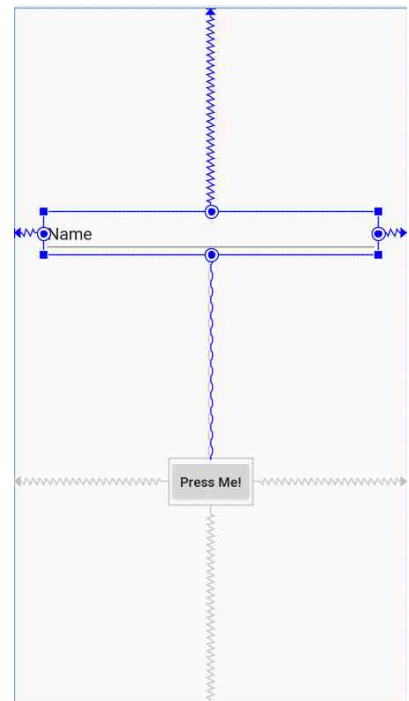
> textAllCaps 속성 > 체크 상자 클릭 > false

2-8. vertical chain 만들기

> Plain Text 클릭 > Ctrl 키를 누른 상태에서 Button 클릭

> 오른쪽 버튼 > Chains > Create Vertical Chain

Q4. 현재는 **portrait mode**임. **Landscape mode**로 변경했을 때 레이아웃 캡처(layout editor 화면에서 이미지 캡처할 것).



### □ STEP 3: layout XML 파일 내용 살펴보기

3-1. app > res > layout > activity\_main.xml 더블 클릭

3-2. Text 탭 클릭

3-3. EditText에 관한 constraint layout 속성 값이 아래와 같다.

Q5 : app:layout\_constraintStart\_toStartOf 속성에서 Start의 의미는?

Q6 : app:layout\_constraintEnd\_toEndOf 속성에서 End의 의미는?

Q7 : app:layout\_constraintHorizontal\_bias 속성은 속성값 0.5의 의미는?

```
<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:inputType="text"
    android:text="Name"
    android:ems="10"
    android:width="350dp"
    android:id="@+id/editText"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintBottom_toTopOf="@+id/button"
    app:layout_constraintHorizontal_bias="0.5" />
```

### □ STEP 4: 소스 코드에서 UI 생성 - ViewGroup과View 객체 생성(1/2)

4-1. MainActivity.kt > onCreate 메소드

setContentView( . . . ) 문장을 주석 처리.

> 아래와 같이 문장을 추가(단계별로 코드를 추가

※ 다음 단계에서는 새롭게 바뀐 내용만 입력하면 됨

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        // setContentView(R.layout.activity_main)

        val myButton = Button(this)
        val myLayout = ConstraintLayout(this)
        myLayout.addView(myButton)
        setContentView(myLayout)
    }
}
```

4-2. MyLayout 실행

Q8 : 실행 화면 캡처

## □ STEP 5: 소스 코드에서 UI 생성 - ViewGroup과View 속성 설정(2/2)

### 5-1. MainActivity.kt > onCreate 메소드

소스 코드를 다음과 같이 수정(새롭게 바뀐 내용을 찾아 입력)

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        // setContentView(R.layout.activity_main)  
  
        val myButton = Button(this)  
        myButton.text = getString(R.string.button_string)  
        myButton.setBackgroundColor(Color.YELLOW)  
  
        val myLayout = ConstraintLayout(this)  
        myLayout.setBackgroundColor(Color.BLUE)  
  
        myLayout.addView(myButton)  
        setContentView(myLayout)  
    }  
}
```

### 5-2. MyLayout 실행

Q9. 실행 결과 버튼에 출력되는 문자는 대문자로 표시된다. 대소문자를 구분해서 출력되도록 코드를 수정하시오(수정 코드만 포함).

### 5-3. app > res > values > 오른쪽 버튼 > New > Values resource file

> File name: id.xml > Enter

> id.xml 을 아래와 같이 수정

```
<?xml version="1.0" encoding="utf-8"?>  
<resources>  
    <item name="myButton" type="id" />  
    <item name="myEditText" type="id" />  
</resources>
```

### 5-4. MainActivity.kt > onCreate 메소드

소스 코드를 다음과 같이 수정(새롭게 바뀐 내용을 찾아 입력)

```
val myButton = Button(this)  
myButton.text = getString(R.string.button_string)  
myButton.setBackgroundColor(Color.YELLOW)  
myButton.id = R.id.myButton
```

### 5-5. MyLayout 실행

## □ STEP 6: Button에 대한 constraint 속성 설정

### 6-1. MainActivity.kt > onCreate 메소드

소스 코드를 다음과 같이 수정(새롭게 바뀐 내용을 찾아 입력)

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        // setContentView(R.layout.activity_main)  
  
        val myButton = Button(this)  
        myButton.text = getString(R.string.button_string)  
        myButton.setBackgroundColor(Color.YELLOW)  
        myButton.id = R.id.myButton  
  
        val myLayout = ConstraintLayout(this)  
        myLayout.setBackgroundColor(Color.BLUE)  
  
        myLayout.addView(myButton)  
        setContentView(myLayout)  
  
        val set = ConstraintSet()  
        set.constrainHeight(myButton.id,  
            ConstraintSet.WRAP_CONTENT)  
        set.constrainWidth(myButton.id,  
            ConstraintSet.WRAP_CONTENT)  
  
        set.connect(myButton.id, ConstraintSet.START,  
            ConstraintSet.PARENT_ID, ConstraintSet.START, 0)  
        set.connect(myButton.id, ConstraintSet.END,  
            ConstraintSet.PARENT_ID, ConstraintSet.END, 0)  
        set.connect(myButton.id, ConstraintSet.TOP,  
            ConstraintSet.PARENT_ID, ConstraintSet.TOP, 0)  
        set.connect(myButton.id, ConstraintSet.BOTTOM,  
            ConstraintSet.PARENT_ID, ConstraintSet.BOTTOM, 0)  
  
        set.applyTo(myLayout)  
    }  
}
```

### 6-2. MyLayout 실행

Q10. 실행 화면 캡처(실습: 5의 실행 결과와 무엇이 달라졌는지 확인할 것)



## □ STEP 7: EditText 추가 및 constraint 속성 설정

### 7-1. MainActivity.kt > onCreate 메소드

소스 코드를 다음과 같이 수정(새롭게 바뀐 내용을 찾아 입력. 전체 코드 중 수정해야 할 부분만 나타낸 것임)

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    // setContentView(R.layout.activity_main)

    val myButton = Button(this)
    myButton.text = getString(R.string.button_string)
    myButton.setBackgroundColor(Color.YELLOW)
    myButton.id = R.id.myButton

    val myEditText = EditText(this)
    myEditText.setText("Name")
    myEditText.id = R.id.myEditText

    myLayout.addView(myButton)
    myLayout.addView(myEditText)
    setContentView(myLayout)

    set.connect(myButton.id, ConstraintSet.START,
        ConstraintSet.PARENT_ID, ConstraintSet.START, 0)
    set.connect(myButton.id, ConstraintSet.END,
        ConstraintSet.PARENT_ID, ConstraintSet.END, 0)
    set.connect(myButton.id, ConstraintSet.TOP,
        myEditText.id, ConstraintSet.BOTTOM, 0)
    set.connect(myButton.id, ConstraintSet.BOTTOM,
        ConstraintSet.PARENT_ID, ConstraintSet.BOTTOM, 0)

    set.constrainHeight(myEditText.id,
        ConstraintSet.WRAP_CONTENT)
    set.constrainWidth(myEditText.id,
        ConstraintSet.WRAP_CONTENT)

    set.connect(myEditText.id, ConstraintSet.LEFT,
        ConstraintSet.PARENT_ID, ConstraintSet.LEFT, 0)
    set.connect(myEditText.id, ConstraintSet.RIGHT,
        ConstraintSet.PARENT_ID, ConstraintSet.RIGHT, 0)
    set.connect(myEditText.id, ConstraintSet.TOP,
        ConstraintSet.PARENT_ID, ConstraintSet.TOP, 0)
    set.connect(myEditText.id, ConstraintSet.BOTTOM,
        myButton.id, ConstraintSet.TOP, 0)

    set.applyTo(myLayout)
```

### 7-2. MyLayout 실행

## □ STEP 8: EditText 객체의 너비(width) 값 지정

### 8-1. MainActivity.kt > onCreate 메소드

EditText 객체의 width 값을 아래와 같이 입력하고 실행시켜보자.

```
val myEditText = EditText(this)
myEditText.setText("Name")
myEditText.width = 350
myEditText.id = R.id.myEditText
```

### 8-2. 변환함수를 정의하고 실행시켜보자.

```
val myEditText = EditText(this)
myEditText.setText("Name")
myEditText.width = convertToPx(350)
myEditText.id = R.id.myEditText

private fun convertToPx(value: Int): Int {
    val px = TypedValue.applyDimension(
        TypedValue.COMPLEX_UNIT_DIP, value.toFloat(),
        resources.displayMetrics).toInt()

    return px
}
```

Q11. 위 함수 대신 아래 코드를 함수로 만들어 호출하는 코드를 작성하시오(함수 이름은 위와 똑같이 convertToPx로 할 것. 수정한 코드만 제출)

```
val edWidth = 350
myEditText.width = edWidth * resources.displayMetrics.density.toInt()
```

Q12 res/values/dimen.xml에 EditText 객체의 너비 350dp를 정의하고, 이 값을 width 속성에 지정하는 코드를 작성하시오(수정한 코드만 제출).