

항목 선택(1)

Mobile Software
2022 Fall

All rights reserved, 2022, Copyright by Youn-Sik Hong (편집, 배포 불허)

What to do next?

- 항목 선택
 - **RadioButton**
 - CheckBox
 - Spinner
 - AlertDialog
- 강의 노트에 포함된 코드 제공(7-2.소스코드. hwp)

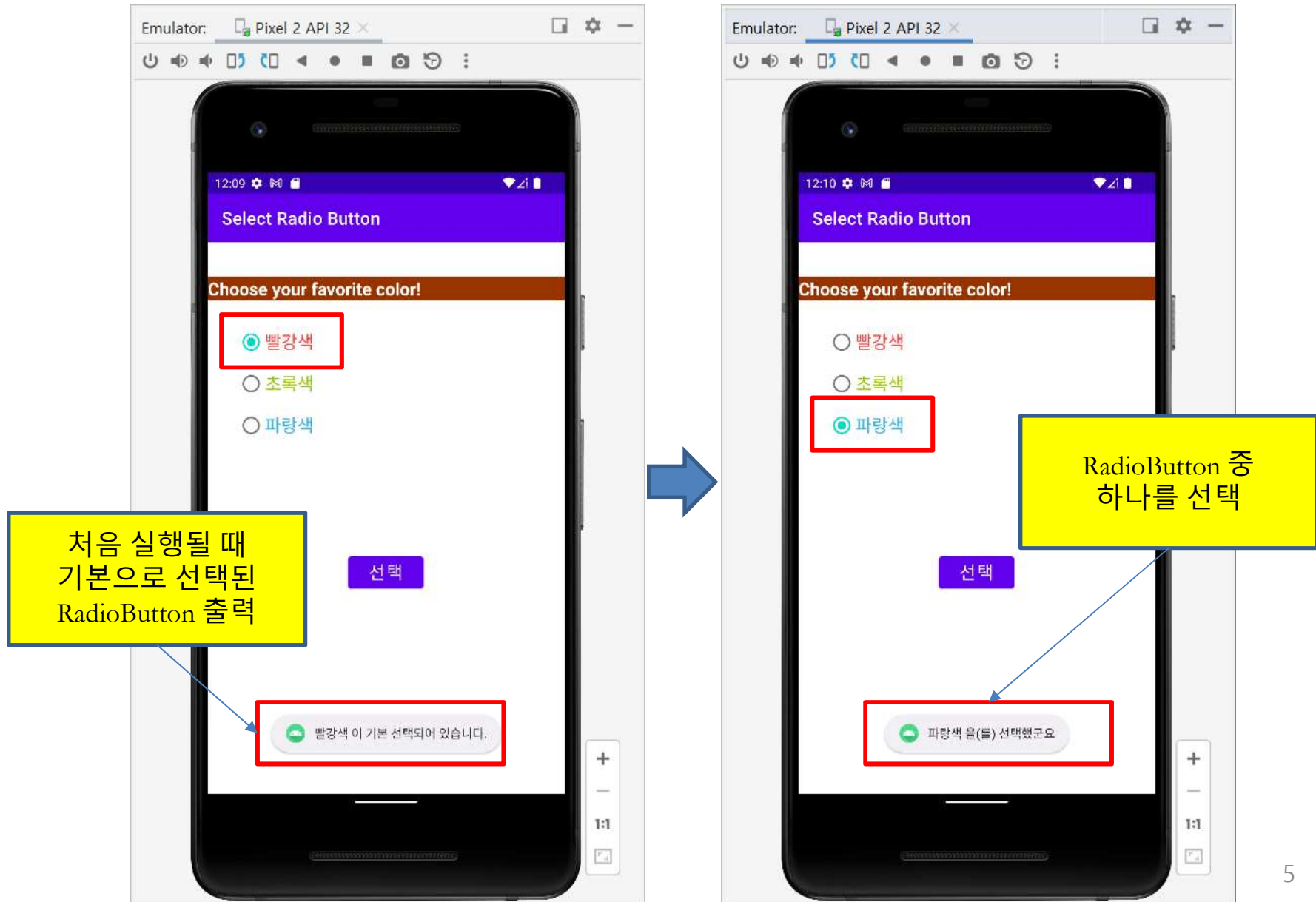
RadioGroup과 RadioButton

- A radio button is a two-states button that can be either checked or unchecked.
 - checked (**true**) or unchecked (**false**) → **isChecked** ()
- Radio buttons are normally used together in a **RadioGroup**.
 - 그룹에 속한 RadioButton 중 한 개만 선택 가능(**mutually exclusive**)
- RadioButton inherits from ... **TextView**.
 - All the standard TextView properties for font face, style, color, ... are available for controlling the look of radio buttons.

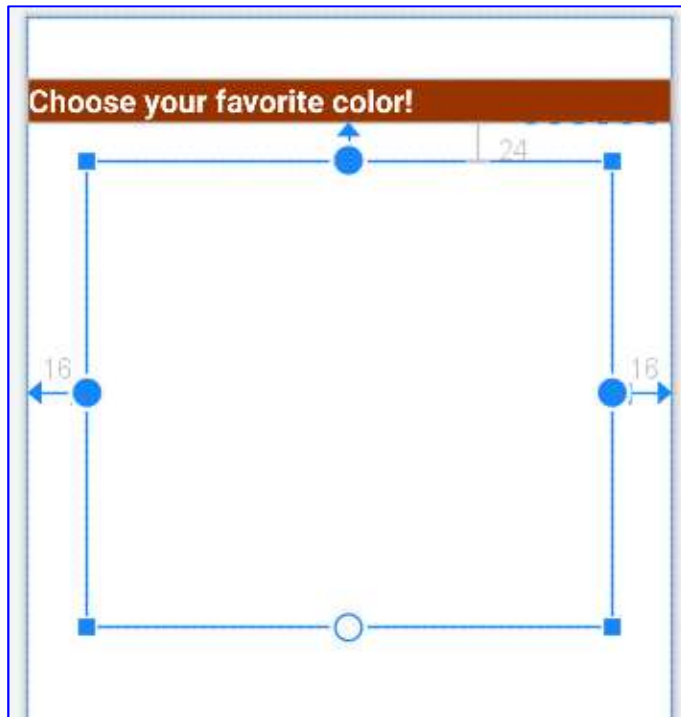
실습 프로젝트 생성

- 새 프로젝트 생성
 - Project name : **Select Radio Button**
 - Package name : **com.example.selectradiobutton**
 - Activity : **Empty Activity**
 - Activity name : **MainActivity.kt**
 - Layout name : **activity_main.xml**
- 자동 생성된 XML 파일의 root view는 **ConstraintLayout**

실습 1: RadioButton

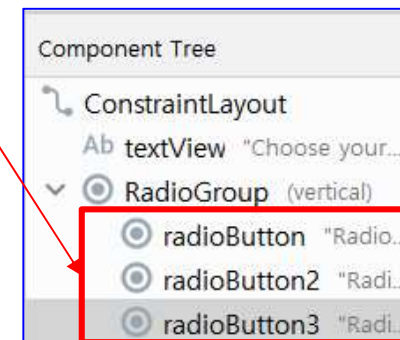
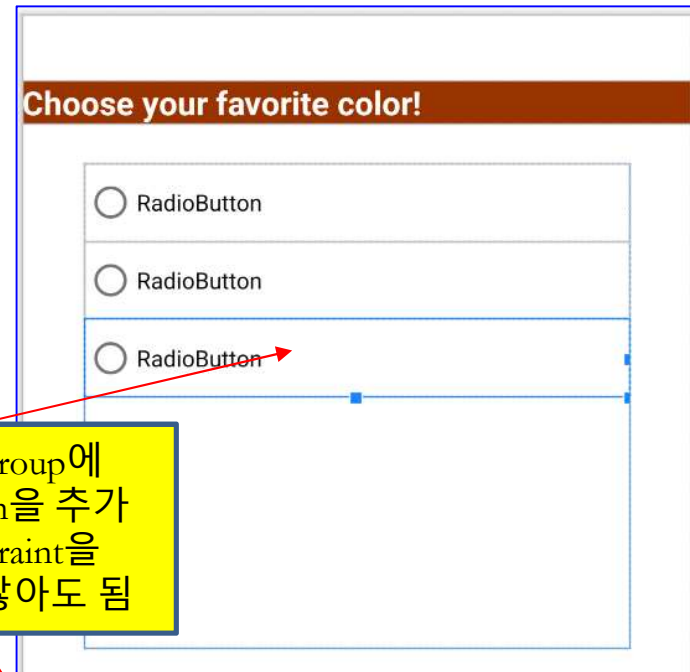


RadioGroup에 RadioButton 추가 (1/2)

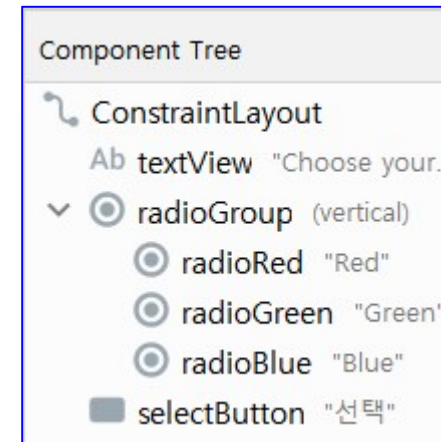
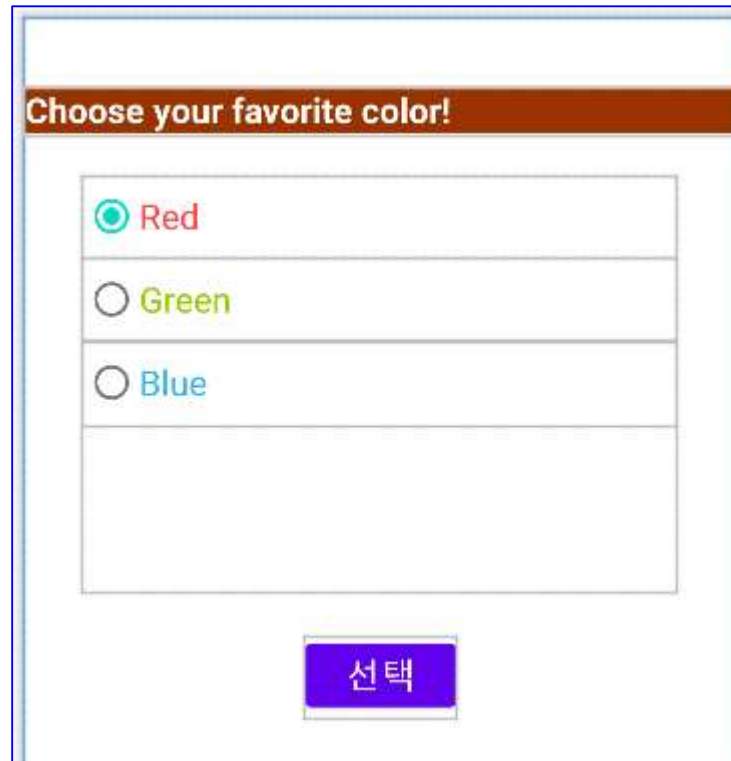


1. RadioGroup이 차지하는 공간을 만들고 constraints를 설정

2. RadioGroup에 RadioButton을 추가
→ Constraint를 설정하지 않아도 됨



RadioGroup에 RadioButton 추가 (2/2)



문자열 배열 리소스

res/values/strings.xml

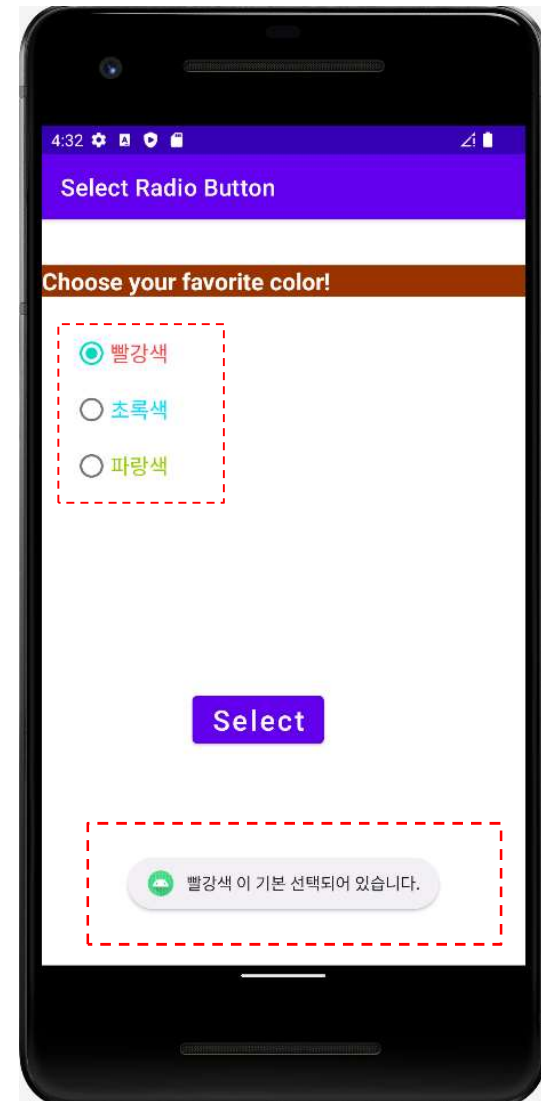
```
<resources>
    <string name="app_name">Select Radio Button</string>
    <string-array name="colors">
        <item>빨강색</item>
        <item>초록색</item>
        <item>파랑색</item>
    </string-array>
</resources>
```


앱 초기 동작: 기본 설정 확인

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val radioRed: RadioButton = findViewById(R.id.radioRed)  
        val radioBlue: RadioButton = findViewById(R.id.radioBlue)  
        val radioGreen: RadioButton = findViewById(R.id.radioGreen)  
  
        val colorArray: Array<String>  
            = resources.getStringArray(R.array.colors)  
        radioRed.text = colorArray[0]  
        radioBlue.text = colorArray[1]  
        radioGreen.text = colorArray[2]  
  
        if (radioRed.isChecked)  
            showToast(radioRed.text.toString())  
        else if (radioBlue.isChecked)  
            showToast(radioBlue.text.toString())  
        else if (radioGreen.isChecked)  
            showToast(radioGreen.text.toString())  
        else  
            showToast(str: "")  
    }  
  
    private fun showToast(str: String) {...}  
}
```

초기 화면에서
기본으로 선택된
버튼을 확인
(이벤트와 무관)

소스코드 - 3쪽



2가지 방법으로 구현

- **RadioButton을 독립된 버튼으로 다룸**
 - RadioButton 3개에 대해
 - 체크 상태를 각각 조사
 - event listener도 각각 등록
 - **View.OnClickListener** 인터페이스 상속
 - callback 메소드: **onClick**
- **RadioButton을 RadioGroup에 종속된 버튼으로 다룸**
 - RadioButton 중 1개만 선택
 - **checkedRadioButtonId** 속성
 - event listener는 1개만 등록
 - **RadioGroup.OnCheckedChangeListener** 인터페이스 상속
 - callback 메소드: **onCheckedChanged**

실습 1A: Activity(1/2) – listener 클래스

```
inner class RadioListener : View.OnClickListener {  
    override fun onClick(v: View?) {  
        if (v == null) return  
        var sb = StringBuilder()  
        when (v.id) {  
            R.id.radioRed -> sb.append((v as RadioButton).text)  
            R.id.radioBlue -> sb.append((v as RadioButton).text)  
            R.id.radioGreen -> sb.append((v as RadioButton).text)  
        }  
        sb.append(" 을(를) 선택했군요")  
        Toast.makeText(applicationContext, sb.toString(),  
            Toast.LENGTH_SHORT).show()  
    }  
}
```

Safe call : v가 null일 수 있기 때문

RadioGroup에 속한 버튼 중 하나는 반드시 선택.
→ 버튼의 checked 상태를 확인할 필요가 없음.
→ 어느 버튼에서 이벤트가 발생했는지 확인.

실습 1A: Activity(2/2)

```
val colorArray:Array<String>
    = resources.getStringArray(R.array.colors)
radioRed.text = colorArray[0]
radioBlue.text = colorArray[1]
radioGreen.text = colorArray[2]

if (radioRed.isChecked)
    showToast(radioRed.text.toString())
else if (radioBlue.isChecked)
    showToast(radioBlue.text.toString())
else if (radioGreen.isChecked)
    showToast(radioGreen.text.toString())
else
    showToast( str: "" )
```

```
val lis = RadioListener()
radioRed.setOnClickListener(lis)
radioBlue.setOnClickListener(lis)
radioGreen.setOnClickListener(lis)
```

3개의 RadioButton이
모두 같은 listener를
이벤트 handler로 등록

실습 1B: RadioGroup을 활용

초기 화면에서
기본으로 선택된
버튼을 확인

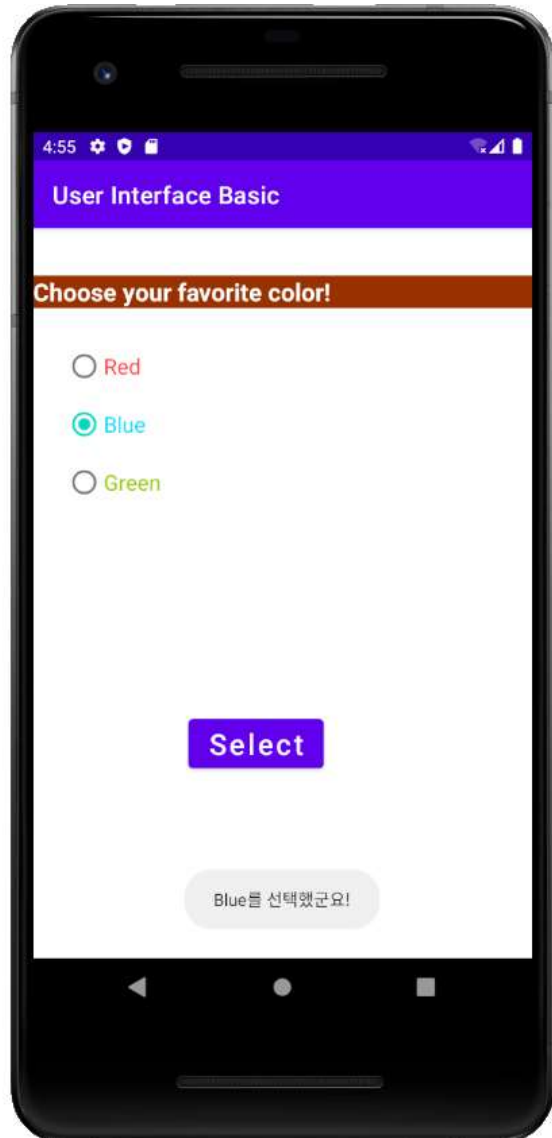
```
val radioRed: RadioButton = findViewById(R.id.radioRed)
val radioBlue: RadioButton = findViewById(R.id.radioBlue)
val radioGreen: RadioButton = findViewById(R.id.radioGreen)

val radioGroup: RadioGroup = findViewById(R.id.radioGroup)
when (radioGroup.checkedRadioButtonId) {
    R.id.radioRed -> showToast(radioRed.text.toString())
    R.id.radioBlue -> showToast(radioBlue.text.toString())
    R.id.radioGreen -> showToast(radioGreen.text.toString())
    else -> showToast(str. "")
}
```

RadioGroup만
이벤트 listener를
등록

```
radioGroup.setOnCheckedChangeListener { _, checkedId ->
    var sb = StringBuilder()
    var rdButton: RadioButton = findViewById(checkedId)
    sb.append(rdButton.text)
    sb.append(" 을(를) 선택했군요")
    Toast.makeText(applicationContext, sb.toString(),
        Toast.LENGTH_SHORT).show()
}
```

Do it yourself : Select 버튼은 왜 사용하지 않나요?



- **Missions to be completed**

- 1. RadioButton에 대한 이벤트 listener는 없앴
 - RadioButton을 선택하거나 변경해도 아무 것도 출력하지 않음.
 - 단, 초기 상태에서 기본으로 선택된 RadioButton을 출력하는 코드는 그대로 유지
- 2. Select 버튼을 클릭했을 때 앞의 코드와 동일한 결과가 나오도록 코드를 수정
- 3. 리스너를 구현하는 코드를 람다 식으로 표현

What to do next?

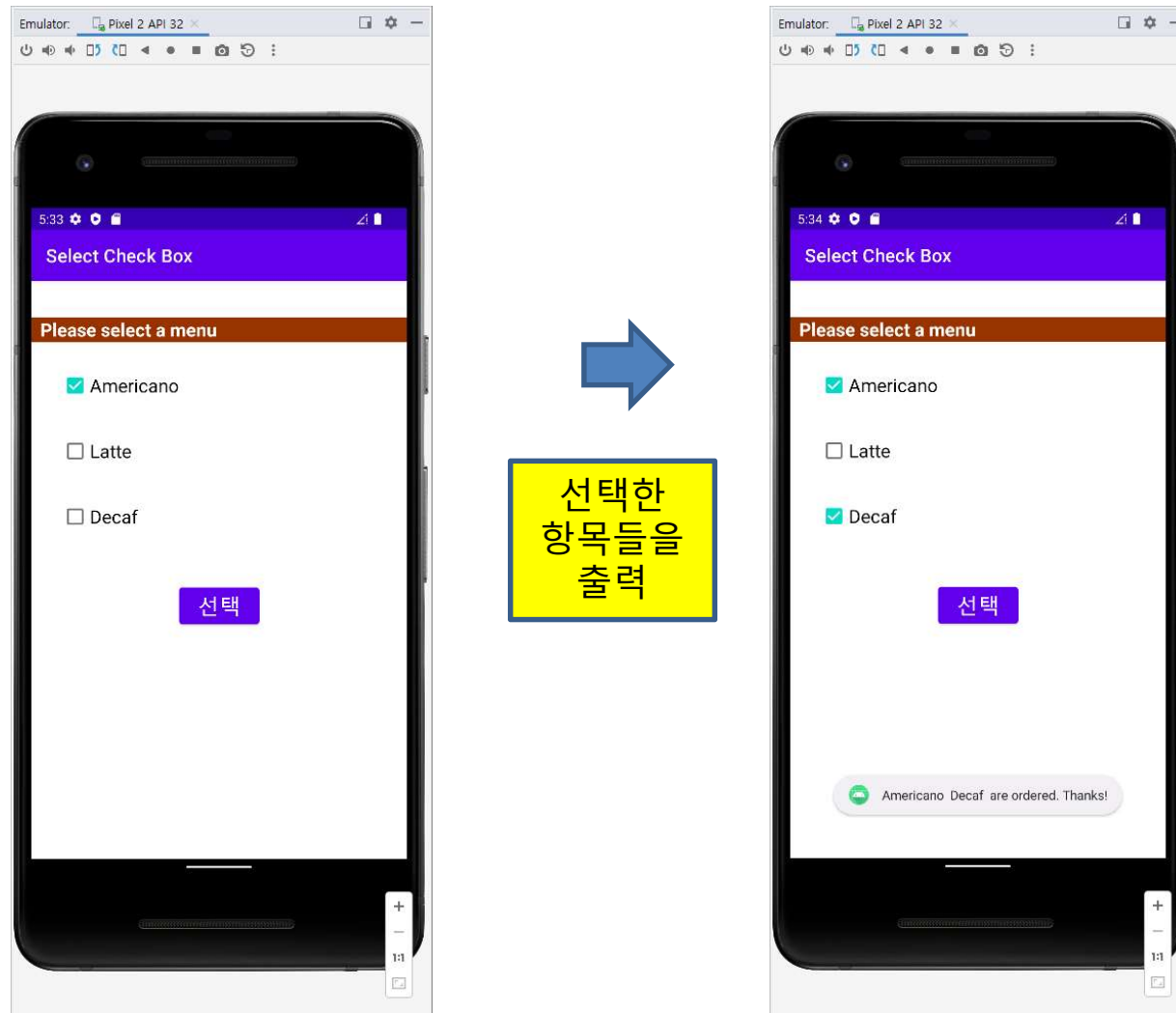
- 항목 선택
 - RadioButton
 - **CheckBox**
 - Spinner
 - AlertDialog

실습 프로젝트 생성

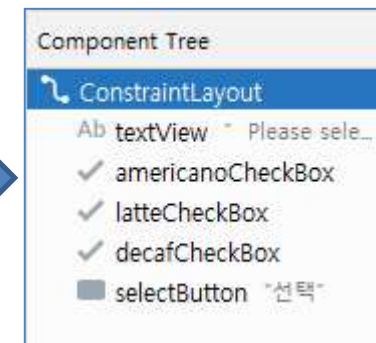
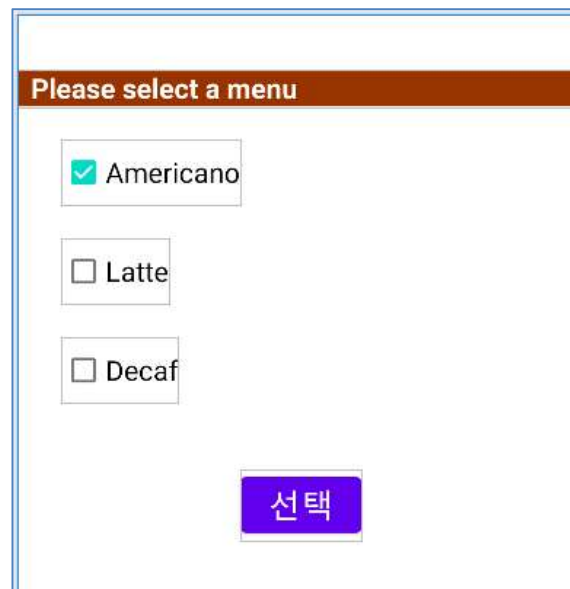
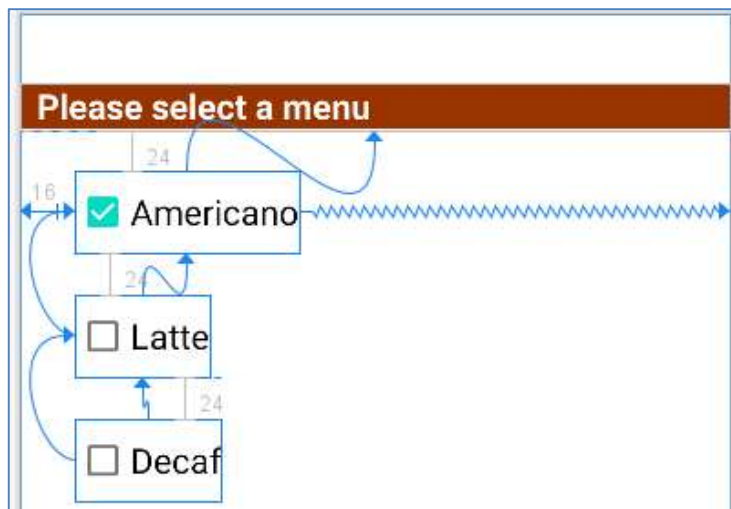
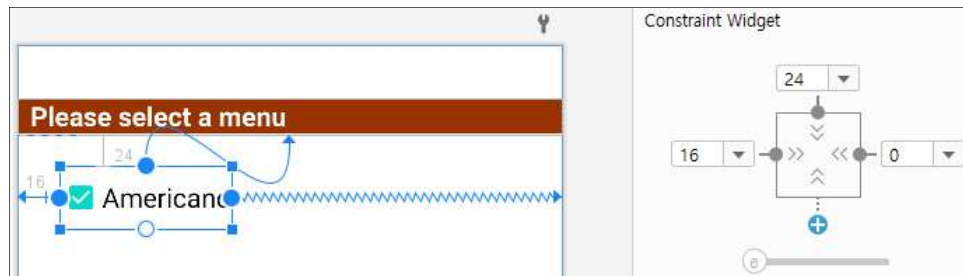
- 새 프로젝트 생성
 - Project name : **Select Check Box**
 - Package name : **com.example.selectcheckbox**
 - Activity : **Empty Activity**
 - Activity name : **MainActivity.kt**
 - Layout name : **activity_main.xml**
- 자동 생성된 XML 파일의 root view는 **ConstraintLayout**

CheckBox

- Two-states button : 다중 선택
 - checked (**true**) 또는 unchecked (**false**) → **isChecked** ()



CheckBox 화면 레이아웃



문자열 배열 리소스

res/values/strings.xml

```
<resources>
  <string name="app_name">Select Check Box</string>
  <string-array name="coffee_menus">
    <item>Americano</item>
    <item>Latte</item>
    <item>Decaf</item>
  </string-array>
</resources>
```

실습 2: CheckBox

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val checkAmericano: CheckBox = findViewById(R.id.americanoCheckBox)  
        val checkLatte: CheckBox = findViewById(R.id.latteCheckBox)  
        val checkDecaf: CheckBox = findViewById(R.id.decafCheckBox)  
  
        val coffeeArr: Array<String> = resources.getStringArray(R.array.coffee_menus)  
        checkAmericano.text = coffeeArr[0]  
        checkLatte.text = coffeeArr[1]  
        checkDecaf.text = coffeeArr[2]  
  
        val sel: Button = findViewById(R.id.selectButton)  
        sel.setOnClickListener { it: View! -> {  
            var sb = StringBuilder()  
            if (checkAmericano.isChecked) sb.append(" ${checkAmericano.text} ")  
            if (checkLatte.isChecked) sb.append(" ${checkLatte.text} ")  
            if (checkDecaf.isChecked) sb.append(" ${checkDecaf.text} ")  
            sb.append(" are ordered. Thanks!")  
  
            Toast.makeText(applicationContext,  
                sb.toString(), Toast.LENGTH_LONG).show()  
        }}  
    }  
}
```

CheckBox 항목
선택 여부 확인

개별 CheckBox에 대한 이벤트 처리

```
val checkAmericano: CheckBox = findViewById(R.id.americanoCheckBox)
val checkLatte:CheckBox = findViewById(R.id.latteCheckBox)
val checkDecaf:CheckBox = findViewById(R.id.decafCheckBox)

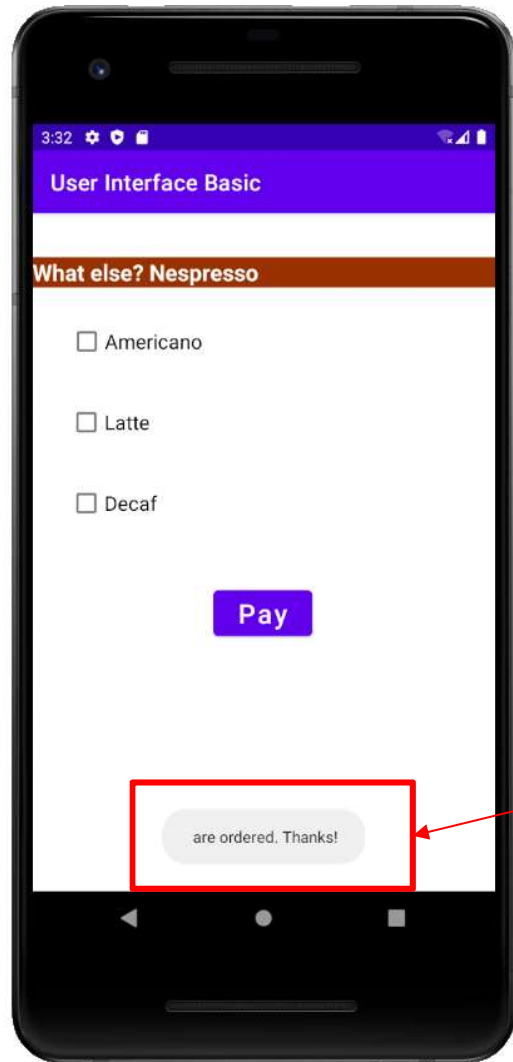
val coffeeArr:Array<String> = resources.getStringArray(R.array.coffee_menus)
checkAmericano.text = coffeeArr[0]
checkLatte.text = coffeeArr[1]
checkDecaf.text = coffeeArr[2]

checkAmericano.setOnCheckedChangeListener(
    object : CompoundButton.OnCheckedChangeListener {
        override fun onCheckedChanged(view: CompoundButton?,
                                       checked: Boolean) {
            showToast(check, str: " ${checkAmericano.text} ")
        }
    })
```

```
checkAmericano.setOnCheckedChangeListener { view, checked ->
    showToast(check, str: " ${checkAmericano.text} ")
}
```

람다 식 변환

Do it yourself : 소스 코드의 bug를 해결



- Missions to be completed
 - 1. Bug 해결
 - 아무 것도 선택하지 않았으면,
 - 아래와 같은 안내 메시지 출력
 - "Nothing selected. Please order again."
 - 2. 리스너를 구현하는 코드를 람다 식으로 표현

아무 것도
check하지 않고
Pay 버튼을 클릭하면

What to do next?

- 항목 선택
 - RadioButton
 - CheckBox
 - **Spinner**
 - AlertDialog

Spin Control

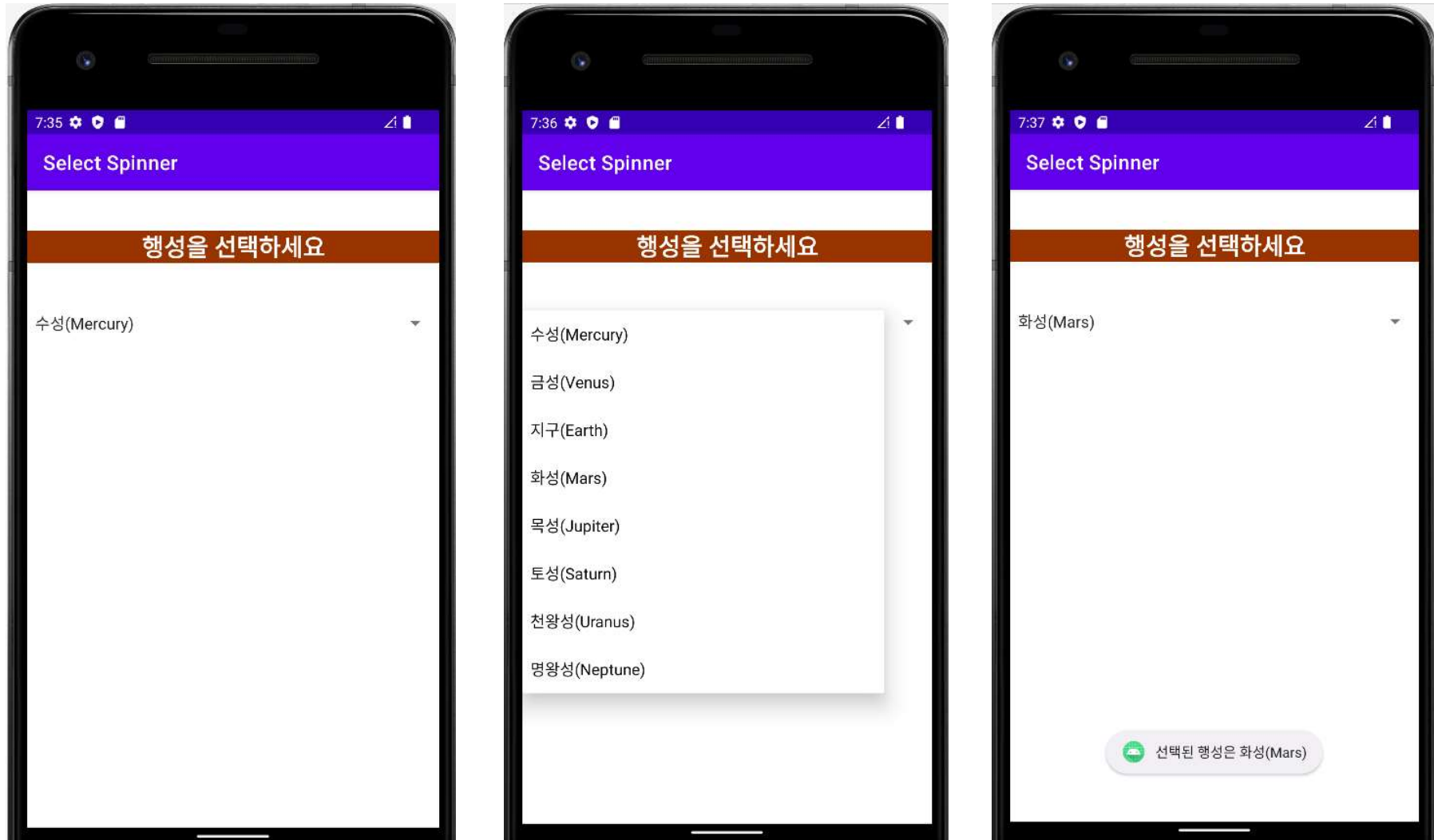
- **Spinner** is the equivalent of the drop-down selector.
- Spinners have the *same functionality of a RecyclerView* but **take less space**.
- As with **RecyclerView**, you provide the adapter for data and child views via **Adapter** and hook in a listener object for selections with **OnItemSelectedListener()**.
- Use the **setDropDownViewResource (int resource)** method
 - to supply the resource ID of the view to use.



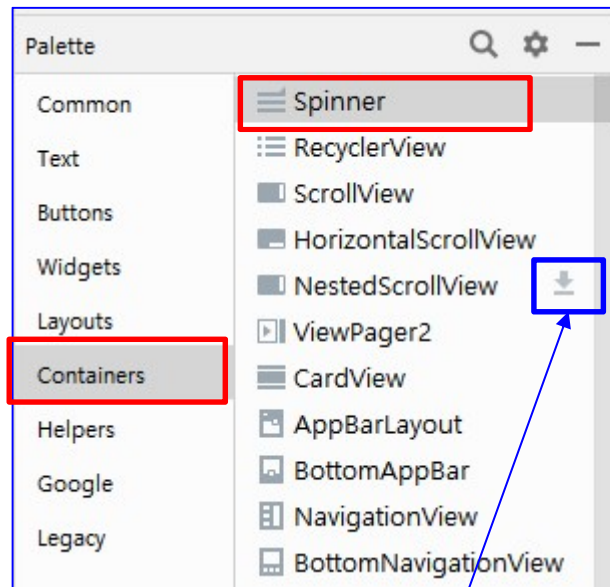
실습 프로젝트 생성

- 새 프로젝트 생성
 - Project name : **Select Spinner**
 - Package name : **com.example.selectspinner**
 - Activity : **Empty Activity**
 - Activity name : **MainActivity.kt**
 - Layout name : **activity_main.xml**
- 자동 생성된 XML 파일의 root view는 **ConstraintLayout**

실습 3: Spinner

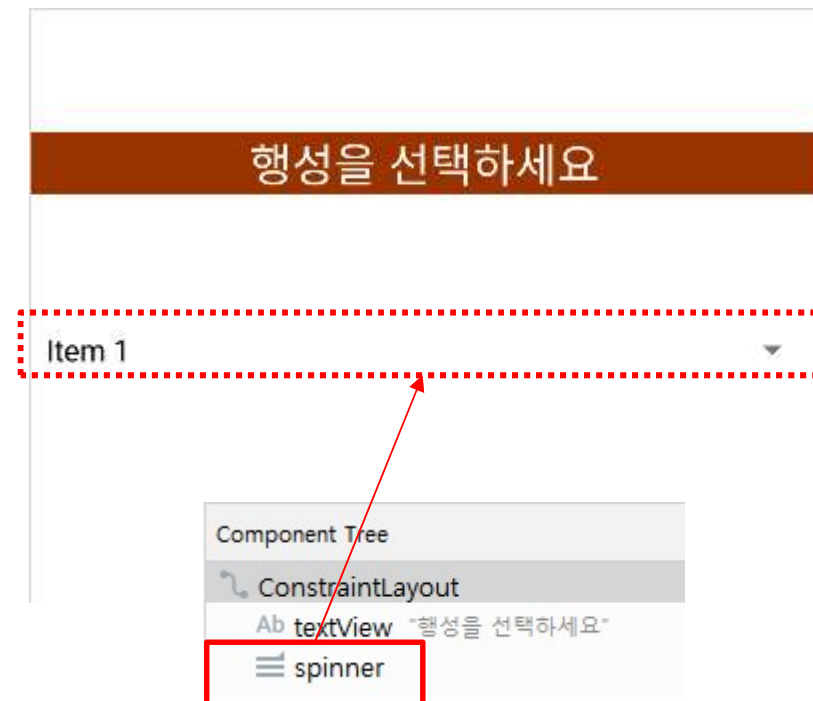


실습 3: Layout



따로 download해서
설치가 필요!

activity_main.xml



문자열 배열 리소스

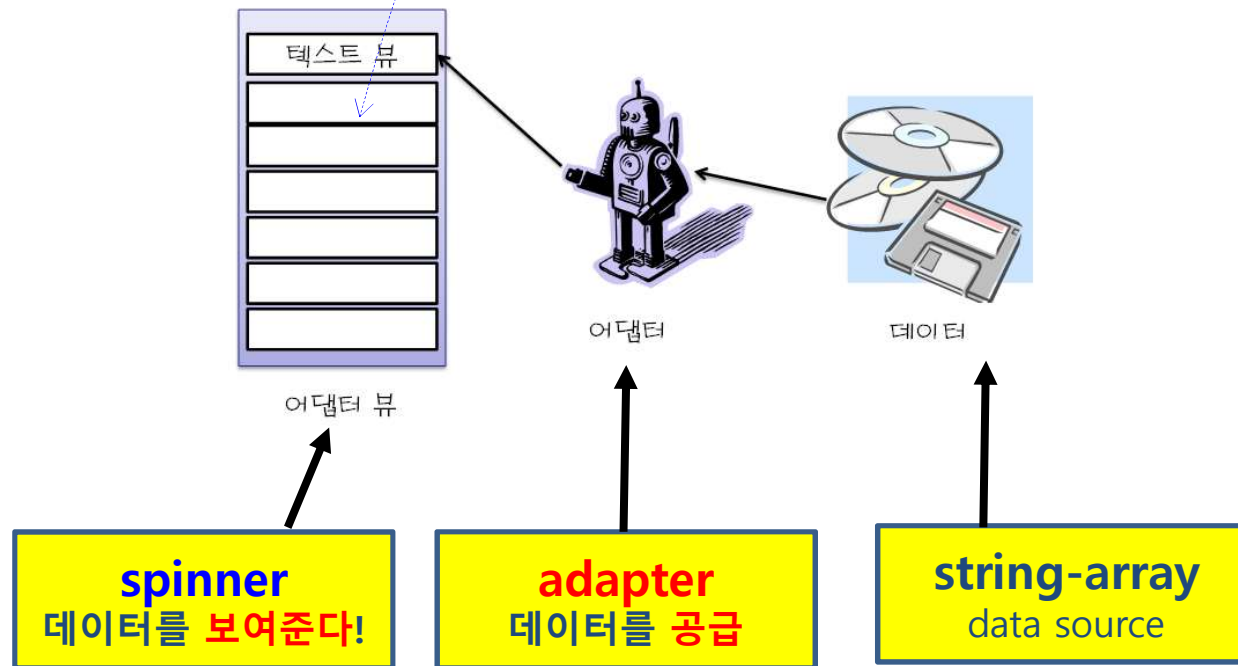
res/values/strings.xml

```
<resources>
    <string name="app_name">Select Spinner</string>
    <string-array name="planets_array">
        <item>수성(Mercury)</item>
        <item>금성(Venus)</item>
        <item>지구(Earth)</item>
        <item>화성(Mars)</item>
        <item>목성(Jupiter)</item>
        <item>토성(Saturn)</item>
        <item>천왕성(Uranus)</item>
        <item>명왕성(Neptune)</item>
    </string-array>
</resources>
```

AdapterView 와 Adapter

```
val planets = resources.getStringArray(R.array.planets_array)
val adapter = ArrayAdapter(this,
    android.R.layout.simple_spinner_item, planets)
adapter.setDropDownViewResource(
    android.R.layout.simple_spinner_dropdown_item)

val spinner:Spinner = findViewById(R.id.spinner)
spinner.adapter = adapter
```



실습 3: Activity

```
val planets = resources.getStringArray(R.array.planets_array)
val adapter = ArrayAdapter(this,
    android.R.layout.simple_spinner_item, planets)
val spinner = findViewById<Spinner>(R.id.spinner)
spinner.adapter = adapter
```

```
adapter.setDropDownViewResource(
    android.R.layout.simple_spinner_dropdown_item)
```

Spinner의 drop down
항목을 보여줌

```
spinner.onItemSelectedListener =
    object : AdapterView.OnItemSelectedListener {
        override fun onItemSelected(parent: AdapterView<*>?,
            view: View?, position: Int, id: Long) {
            var planet = parent?.getItemAtPosition(position).toString()
            Toast.makeText(parent?.context,
                "선택된 행성은 $planet",
                Toast.LENGTH_SHORT).show()
        }

        override fun onNothingSelected(parent: AdapterView<*>?) {
        }
    }
```

Spinner의 drop down 항목을 선택했을 때
발생하는 이벤트를 처리하는 listener 인터페이스
→ 2개의 abstract method를 구현해야 함

잠깐! ArrayAdapter

```
val adapter = ArrayAdapter(this,  
    android.R.layout.simple_spinner_item, planets)
```

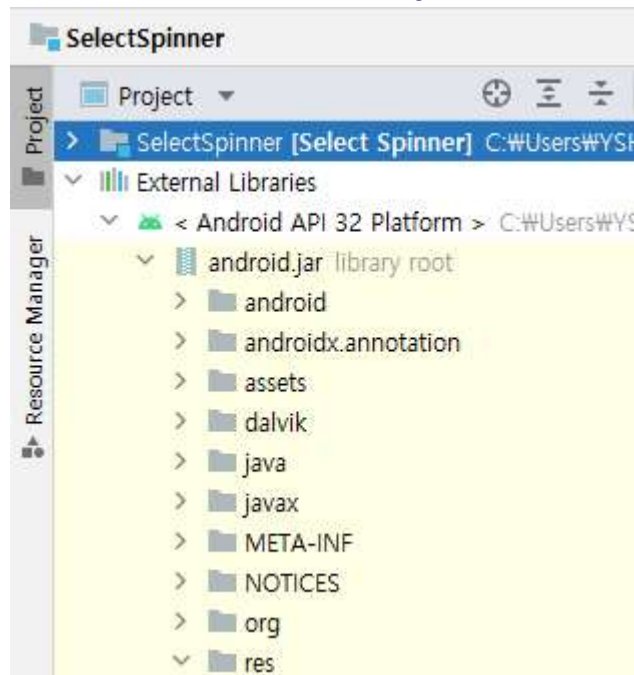
- The **ArrayAdapter** constructor takes *three parameters*:
 - The **Context** to use
 - 어디에 보여 줄 것인가? → UI를 갖고 있는 Activity 객체
 - The **resource ID** of a view to use
 - item의 layout style
 - **android.R.layout.simple_list_item_1**
 - The **actual (source)** array or list of items to show

잠깐! android.R.layout.simple_spinner_item

1. Android SDK 설치 폴더 : c:\Program Files (x86) \Android

« Android > android-sdk > platforms > android-31 > data > res > layout

2. Android Studio > Project 모드



simple_list_item_1.xml
simple_list_item_2.xml
simple_list_item_2_single_choice.xml
simple_list_item_activated_1.xml
simple_list_item_activated_2.xml
simple_list_item_checked.xml
simple_list_item_multiple_choice.xml
simple_list_item_single_choice.xml
simple_selectable_list_item.xml
simple_spinner_dropdown_item.xml
simple_spinner_item.xml

External Libraries

- > API Platform > android.jar
- > res > layout

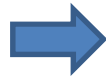
잠깐! **ArrayAdapter<String>**

- **Generic class**(범용 클래스)는 **Collection**에서 주로 사용
 - 컬렉션(Collection)이란?
 - (여러 개의 항목들을 관리하기 위한) 자료 구조
 - **ArrayList, LinkedList, HashSet, TreeSet**
- *T-parameter : Type-parameter* **T**
 - 어떤 type의 객체도 저장할 수 있지만,
 - 엉뚱한 type의 객체가 저장되는 걸 막기 위해
 - 저장할 객체의 type을 지정
- 범용 클래스인 **ArrayAdapter** 객체를 생성할 때
 - *type parameter* **T**를 **String**으로 지정
 - 배열에 속한 항목들의 type을 **String**으로 지정

잠깐! 람다식 확장 함수 also()

```
spinner.onItemSelectedListener =  
    object : AdapterView.OnItemSelectedListener {  
        override fun onItemSelected(parent: AdapterView<*>?,  
            view: View?, position: Int, id: Long) {  
            var planet = parent?.getItemAtPosition(position).toString()  
            Toast.makeText(parent?.context,  
                "선택된 행성은 $planet",  
                Toast.LENGTH_SHORT).show()  
        }  
  
        override fun onNothingSelected(parent: AdapterView<*>?) {  
        }  
    }
```

람다 식과 연결해서
사용 가능한 확장 함수 also()
→ also() 함수를
호출한 객체 T가 it
→ 여기서는 리스트너



```
object : AdapterView.OnItemSelectedListener {  
    override fun onItemSelected(parent: AdapterView<*>?,  
        view: View?, position: Int, id: Long) {  
        var planet = parent?.getItemAtPosition(position).toString()  
        Toast.makeText(parent?.context,  
            "선택된 행성은 $planet",  
            Toast.LENGTH_SHORT).show()  
        }  
  
    override fun onNothingSelected(parent: AdapterView<*>?) {  
    }  
}.also { spinner.onItemSelectedListener = it }
```

잠깐! 리소스를 인자에 직접 지정 (1/2)

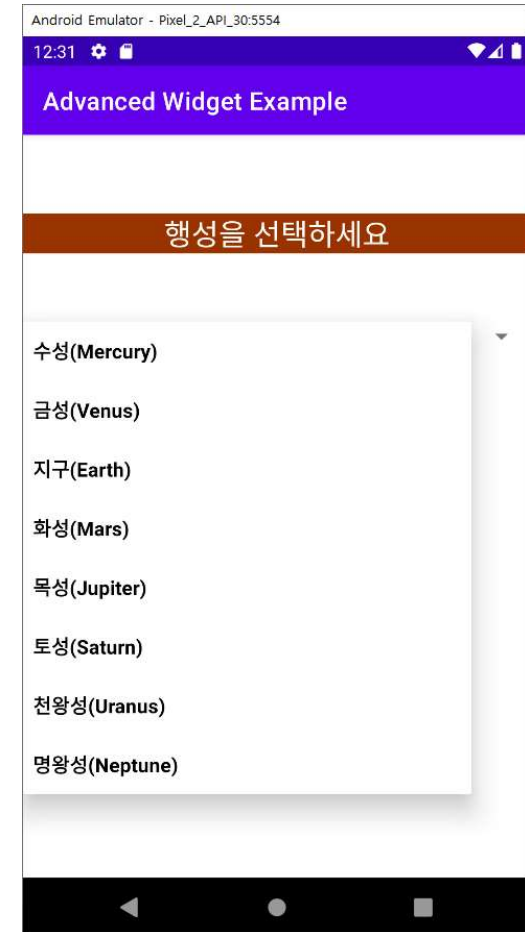
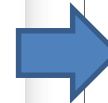
```
val planets = resources.getStringArray(R.array.planets_array)
val adapter = ArrayAdapter(this,
    android.R.layout.simple_spinner_item, planets)
val spinner = findViewById<Spinner>(R.id.spinner)
spinner.adapter = adapter
```



```
val adapter = ArrayAdapter.createFromResource(this,
    R.array.planets_array, android.R.layout.simple_spinner_item)
```

잠깐! 리소스를 인자에 직접 지정 (2/2)

```
<string-array name="planets_array">
  <item><b>수성(Mercury)</b></item>
  <item><b>금성(Venus)</b></item>
  <item><b>지구(Earth)</b></item>
  <item><b>화성(Mars)</b></item>
  <item><b>목성(Jupiter)</b></item>
  <item><b>토성(Saturn)</b></item>
  <item><b>천왕성(Uranus)</b></item>
  <item><b>명왕성(Neptune)</b></item>
</string-array>
```



잠깐! createFromResource

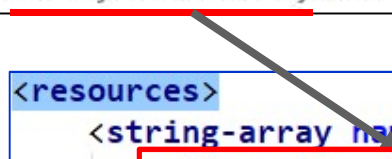
```
ArrayAdapter<CharSequence> createFromResource (Context context,  
                                              int textArrayResId,  
                                              int textViewResId)
```

Creates a new ArrayAdapter from external resources.
The **content of the array** is obtained through **getTextArray** (int).

getTextArray

```
CharSequence[] getTextArray (int id)
```

Return the styled text array associated with a particular resource ID.



```
<resources>  
  <string-array name="fruits">  
    <item><b>apple</b></item>  
    <item><i>banana</i></item>  
    <item>grape</item>  
    <item>orange</item>
```