Implicit Intent

Mobile App Programming

Today's Contents

- Android project
- Intent
  - Implicit intent
  - Broadcast
- Lab practice

PA1

- Until this week Thursday 23:59
  - Late submission until Saturday 23:59.
  - 25%p penalty per day.

- Submit your project zip file (Export to Zip)

- Question? Google Spreadsheet.

Intent

- An Intent is a messaging object you can use to request an action from another app component.
- Three fundamental use cases:
  - Starting an activity
  - Starting a service
  - Delivering a broadcast
- Two types of intents:
  - Explicit intents
  - Implicit intents

https://developer.android.com/guide/components/intents-filters?hl=ko

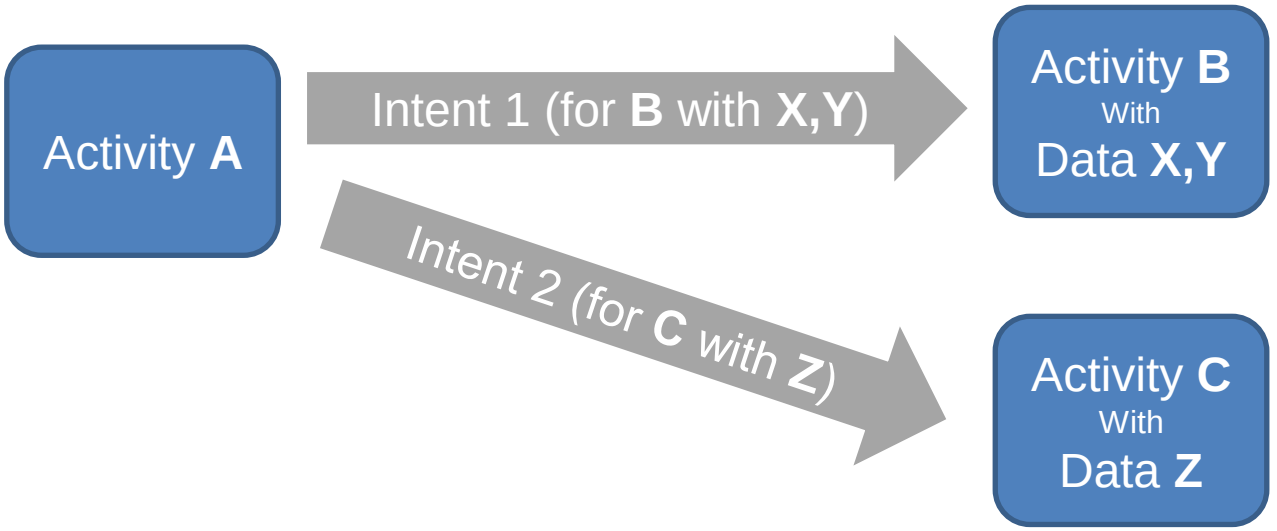
Implicit/Explicit Intent

- **Explicit** intents
  - **Specify** package name or component class name

- **Implicit** intents
  - Do **not specify above name**
  - Just general action
    - Call, Message, Location, …

Explicit Intent

- Explicit intents
  - Specify package name or component class name
  - Start an activity in Intent object
  - Data can be passed via Extras



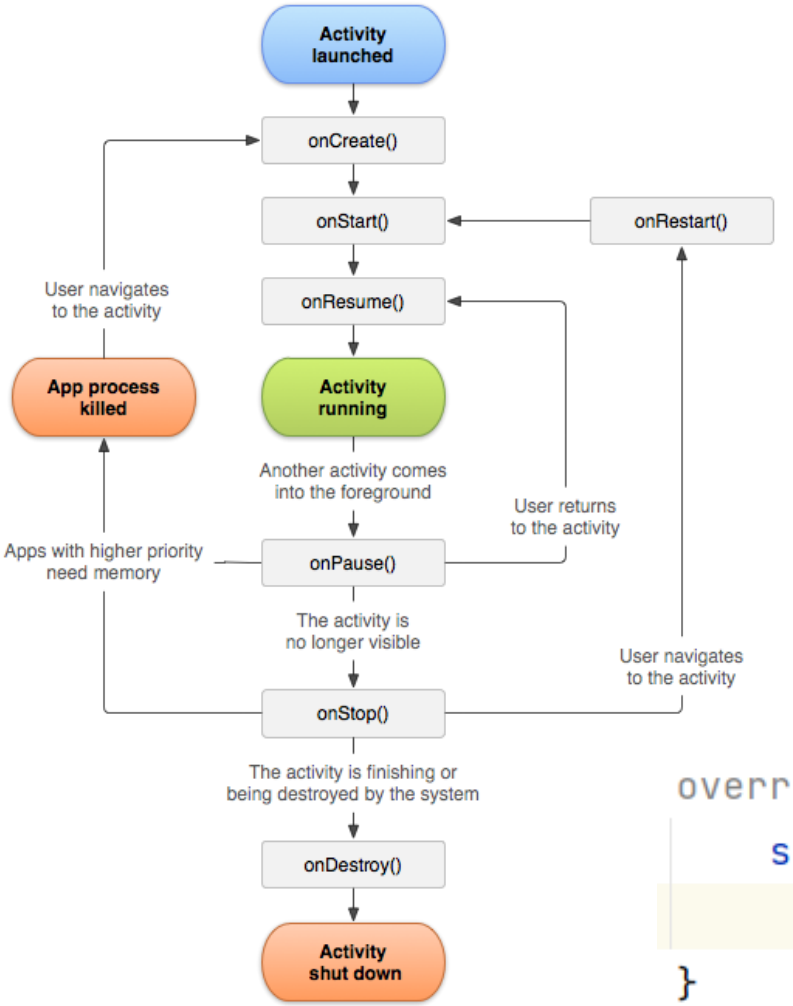
Explicit Intent

- startActivity, putExtra, get***Extra

```
val intent = Intent( packageContext: this, NothingActivity::class.java).apply{ this: Intent
    putExtra(EXT_NAME,  value: "Gildong Hong")
    putExtra(EXT_SID,  value: 2023524288)
}
startActivity(intent)
```

```
val name = intent.getStringExtra(MainActivity.EXT_NAME)
val sid = intent.getIntExtra(MainActivity.EXT_SID,  defaultValue: -1)
```

Activity Lifecycle



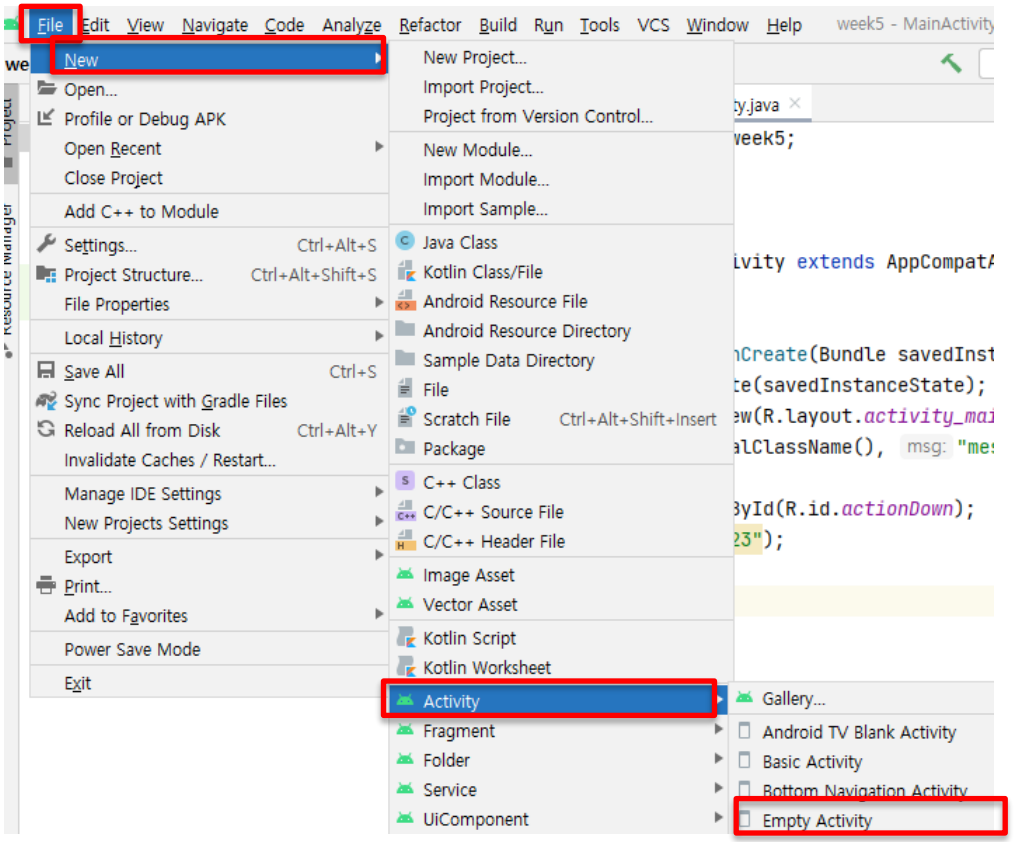

```
override fun onResume(){
    super.onResume()

}
```

Why do we use "New Activity"?

- **Two ways** to make a new activity

(1) Make another activity using "New Empty Activity"

  – File > New
    > Activity
    > Empty Activity

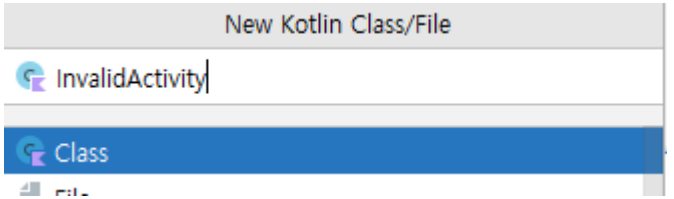


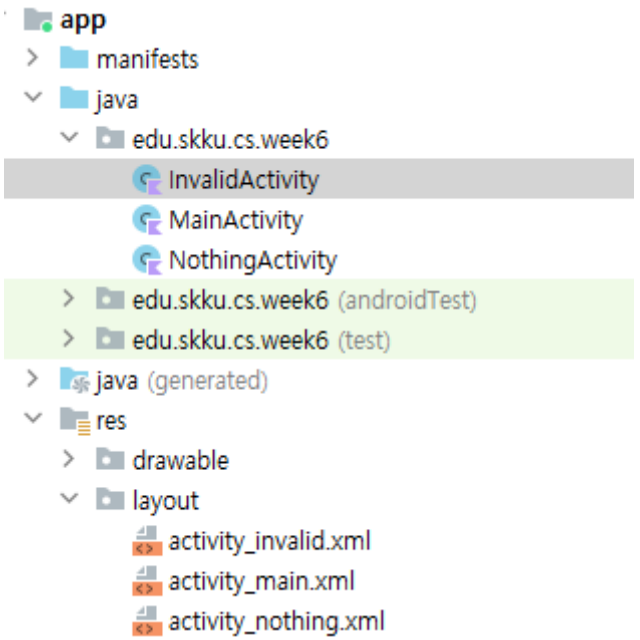
Why do we use "New Activity"?

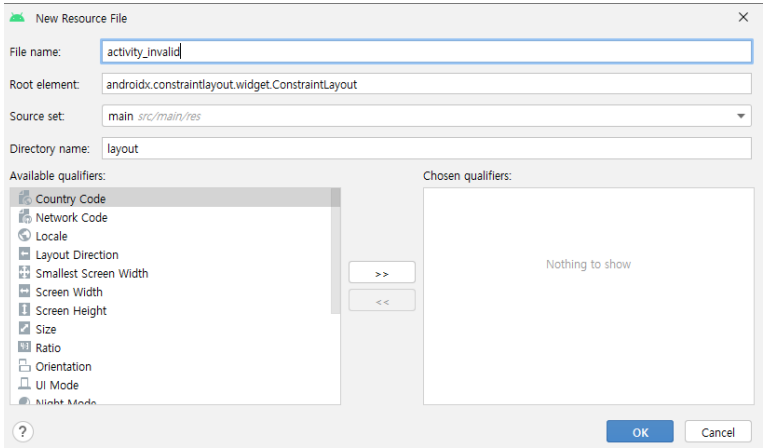
- **Two ways** to make a new activity

(2) Make another activity by creating new Kotlin and XML files

  - New Kotlin class
  - New layout file







Why do we use “New Activity”?

- Does (2) way work?
  - New Kotlin class
  - New layout file

```kotlin
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val intent23 = Intent(
            applicationContext,
            InvalidActivity::class.java
        )
        startActivity(intent23)

    }
}
```

```kotlin
class InvalidActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_invalid)
    }
}
```

```
FATAL EXCEPTION: main
Process: edu.skku.cs.week6, PID: 8784
java.lang.RuntimeException: Unable to start activity ComponentInfo{edu.skku.cs.week6/edu.skku.cs.week6.MainActivity}: android.content.ActivityNotFoundException: Unable to find explicit activity class {edu.skku.cs
    at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:3449)
    at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:3601)
    at android.app.servertransaction.LaunchActivityItem.execute(LaunchActivityItem.java:85)
    at android.app.servertransaction.TransactionExecutor.executeCallbacks(TransactionExecutor.java:135)
    at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:95)
    at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2066)
    at android.os.Handler.dispatchMessage(Handler.java:106)
    at android.os.Looper.loop(Looper.java:223)
    at android.app.ActivityThread.main(ActivityThread.java:7656) <1 internal line>
    at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:592)
    at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:947)
Caused by: android.content.ActivityNotFoundException: Unable to find explicit activity class {edu.skku.cs.week6/edu.skku.cs.week6.InvalidActivity}; have you declared this activity in your AndroidManifest.xml?
    at android.app.Instrumentation.checkStartActivityResult(Instrumentation.java:2065)
    at android.app.Instrumentation.execStartActivity(Instrumentation.java:1727)
    at android.app.Activity.startActivityForResult(Activity.java:5320)
    at androidx.activity.ComponentActivity.startActivityForResult(ComponentActivity.java:728)
    at android.app.Activity.startActivityForResult(Activity.java:5278)
    at androidx.activity.ComponentActivity.startActivityForResult(ComponentActivity.java:709)
    at android.app.Activity.startActivity(Activity.java:5664)
    at android.app.Activity.startActivity(Activity.java:5617)
    at edu.skku.cs.week6.MainActivity.onCreate(MainActivity.kt:19)
    at android.app.Activity.performCreate(Activity.java:8000)
    at android.app.Activity.performCreate(Activity.java:7984)
    at android.app.Instrumentation.callActivityOnCreate(Instrumentation.java:1309)
    at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:3422)<11 more...>
```

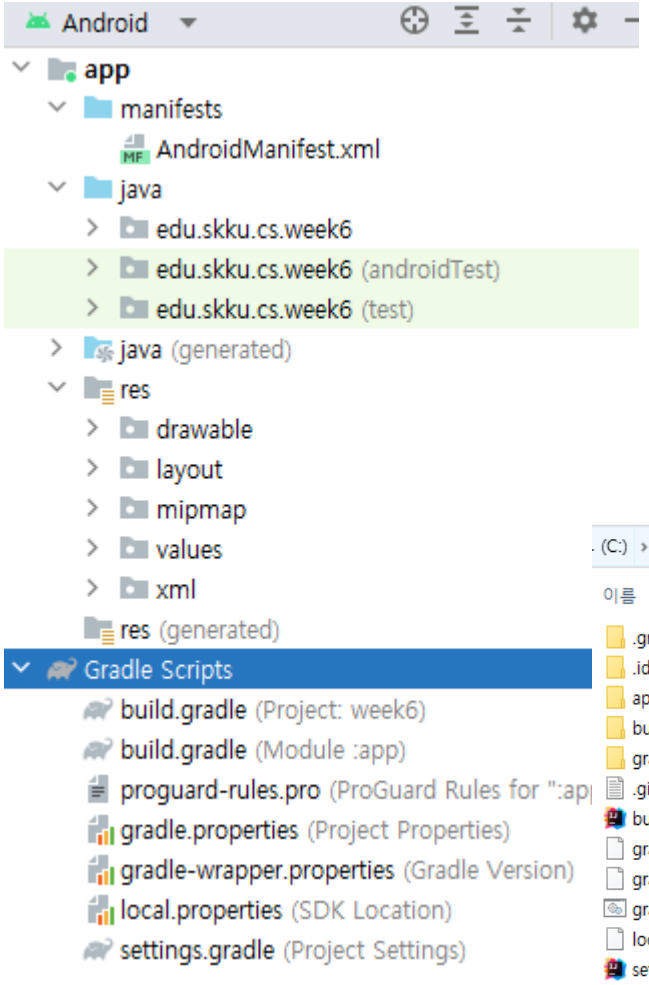
Why do we need “New Activity”?

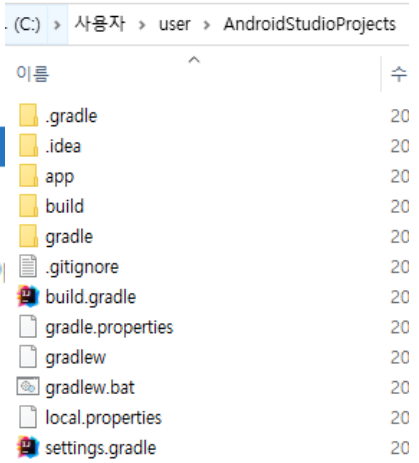
- When we generate with “New Activity”,
  - It will automatically modify **AndroidManifests.xml**

```
        android:theme="@style/Theme.Week6"
        tools:targetApi="31">
        <activity
            android:name=".NothingActivity"
            android:exported="false" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

Android project

- Android project has..
  - app
    - manifests
    - java
    - res
  - Gradle Scripts
    (actually not folder name)





Android project

- manifests
  - **AndroidManifest.xml** is in
  - Let "Android System" know what this app contains, after/while install into device
    - Which activities are in
    - Which permissions are needed
    - which actions will be accepted
    - What is app icon
    - …

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="week6"
        android:supportsRtl="true"
        android:theme="@style/Theme.Week6"
        tools:targetApi="31">
        <activity
            android:name=".NothingActivity"
            android:exported="false" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Android project

- java
  - package folder
    - java/kotlin code

- res = resources
  - drawable
    - image files
  - layout
    - layout files
  - …

Android project

- Gradle Scripts
  - What is gradle?
    - **Build tool** that automate compile, test, deploy, …
- build.gradle **(Project: …)**
  - **Project level** gradle file (Applied on whole project)
  - Basic plugins

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.
plugins {
    id 'com.android.application' version '7.4.2' apply false
    id 'com.android.library' version '7.4.2' apply false
    id 'org.jetbrains.kotlin.android' version '1.8.0' apply false
}
```

Android project : Gradle Scripts

- build.gradle **(Module: app)**

```
android {
    namespace 'edu.skku.cs.week6'
    compileSdk 33

    defaultConfig {
        applicationId "edu.skku.cs.week6"
        minSdk 29
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}
```

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
}
```

```
dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.8.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
}
```

Android project : Gradle Scripts

- build.gradle **(Module: app)**
  - **android {}** : All configuration related with Android…
    - e.g. SDK version
  - **plugins {}** : Configurations related with Android Build.
  - **dependencies {}** : Specify **the version** of libraries or add **external** libraries.

Android project: Gradle Scripts

- settings.gradle
  - **Where to download** the plugins(outside APIs)
  - and some other settings

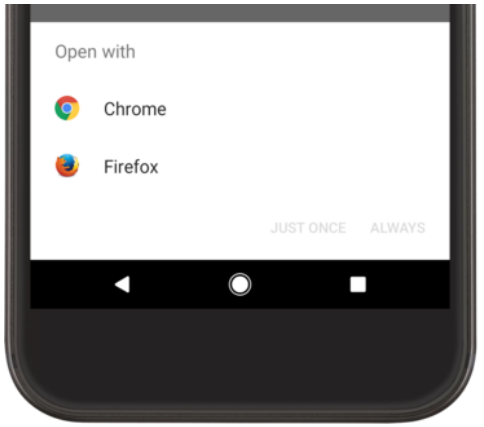
```
pluginManagement {
    repositories {
        google()
        mavenCentral()
        gradlePluginPortal()
    }
}
dependencyResolutionManagement {
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
    }
}
rootProject.name = "week6"
include ':app'
```

Android project: Summary

- Android Manifest
  - Tell the **android system** the configuration of the **application**
    - Package name of application
    - Description of app components(activity, service, broadcast receiver, contents provider)
    - Permission
    - …
- Gradle
  - Tell the IDE(and developer, compiler, …) the configuration of the **project**
    - Compile SDK version
    - Plugins to use
    - …

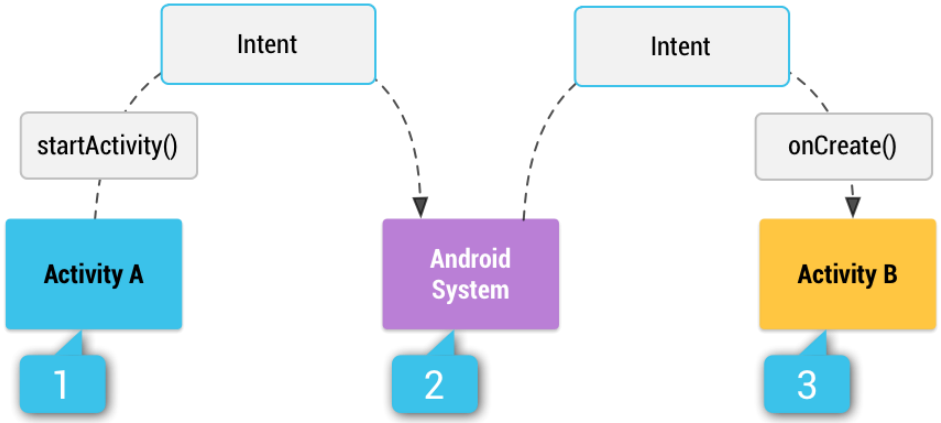
Implicit Intent

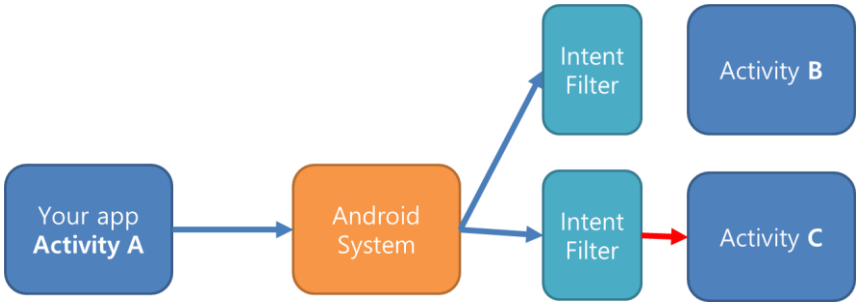
- Why do we need **implicit intent**?
  - There’re multiple applications that can
    - call / message
    - show map
    - send email
    - etc.
  - Then, which application do we use?



Implicit Intent

- How it works? By using **Intent Filter**
  - Declared general action: ACTION_VIEW, ACTION_CALL, ACTION_SEND…
  - Android System will choose candidate via Intent Filter
  - **If no candidate: Fail**
  - If multiple candidates: Let user choose (via system UI)





Implicit Intent

- If you want to **get** implicit intent, (NOT send)
  - Let android system know our app. can handle it.
    - Declare **Intent filter** on AndroidManifest.xml

```xml
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

Implicit Intent

[Intent Sender]

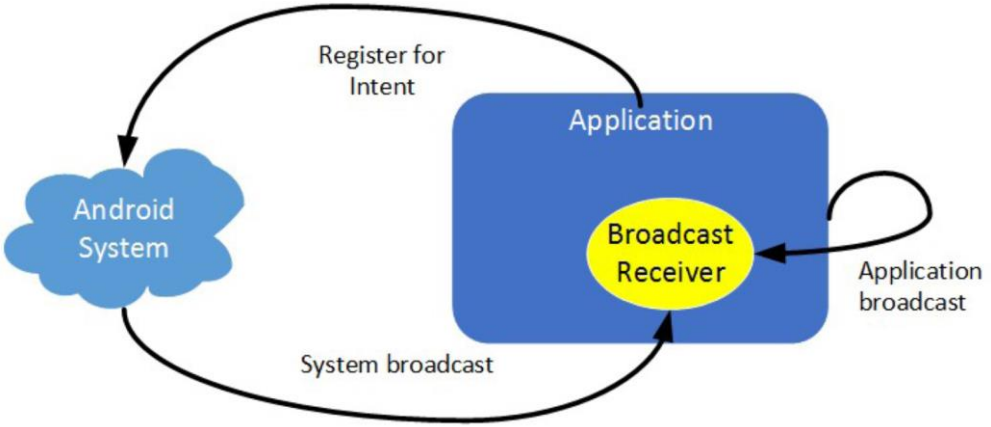
- Suppose that we want to open web browsing application!
  - Just make implicit intent and call *startActivity()*
  - Very similar to explicit intent

[Intent Receiver]

- My application have an activity that contains WebView.
  - Register an **intent filter** in AndroidManifest.xml
    - Android system is managing implicit activity
    - Let system know that it is a web browser app.

Broadcast

- Sending message protocol which declared by Android system. ~~or other application (not recommend since 8.0)~~
- **Implicit Intent** is exactly same with the broadcast message.



https://developer.android.com/guide/components/broadcasts?hl=ko

Implicit Intent

- **Two ways** to handle Implicit Intent.

(1) Receive with **Broadcast Receiver**

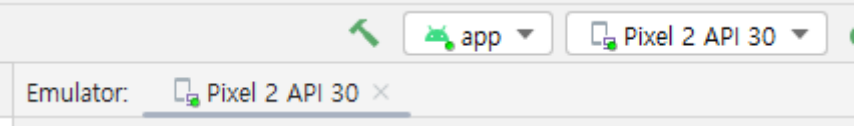
- Receive an Intent which is broadcasted from other places
- Usually used when application does not need to react the message.
- ACTION_BOOT_COMPLETED, ACTION_LOCALE_CHANGED, …
- Notify to all targeted applications when special event happens

(2) Receive with Activity

- Receive Intent which is passed with *startAcitivty()* method
- Used when Intent should be reacted via Activity
  - Usually used when application need to react

Exercise: Implicit Intent

- Please use Emulator(or device) API version 29~32
  - API 29(Android 10.0 Q) ~ API 32(Android 12L Sv2)
  - **Since API 33, there is additional rule for intent filter**
  - This will affect further exercise(Exercise: Broadcast)

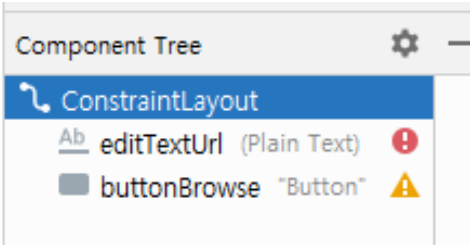


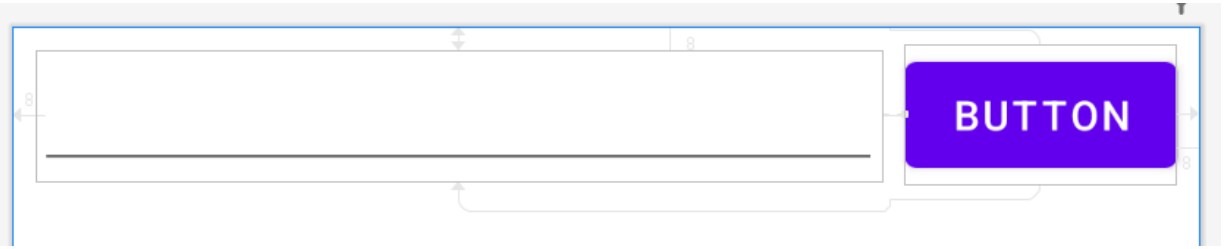
https://gist.github.com/devquint/b02c3afdf7683f9fec4414ac9b25288f

Exercise: Implicit Intent

- We want to open the web browser
  - Which browser to open? Let user to choose!
  - Android system will make a list that have proper intent filters
  - What we need to do is just make intent and call *startActivity()*

(1) Make user interface first!





https://gist.github.com/devquint/b02c3afdf7683f9fec4414ac9b25288f

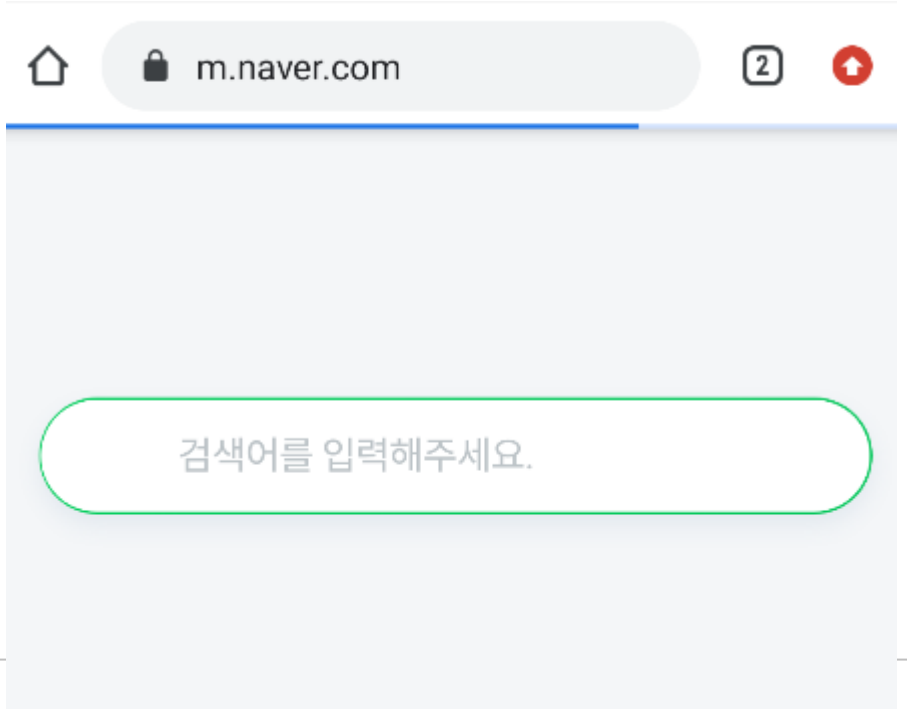
Exercise: Implicit Intent

- Make an implicit intent and call *startActivity()* with it
- Use URI(Uniform Resource Identifier)
  - URL(Uniform Resource Locator) is a subset of URI.

```
val btnStart = findViewById<Button>(R.id.buttonBrowse)
btnStart.setOnClickListener { it: View!
    val urlEditText = findViewById<EditText>(R.id.editTextUrl)
    val uri = Uri.parse( uriString: "https://" + urlEditText.text.toString())
    val webIntent = Intent(Intent.ACTION_VIEW, uri)
    startActivity(webIntent)
}
```

https://gist.github.com/devquint/b02c3afdf7683f9fec4414ac9b25288f

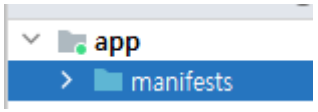
Exercise: Implicit Intent



https://gist.github.com/devquint/d7b5d27a24fc2b8d63dee860d7293eaf

Exercise: Broadcast

- We want to print toast message when the language changed
  - Add intent filter on **AndroidManifest.xml**
  - Android system will send broadcast to us
  - android.intent.action.LOCALE_CHANGED The action that we will handle



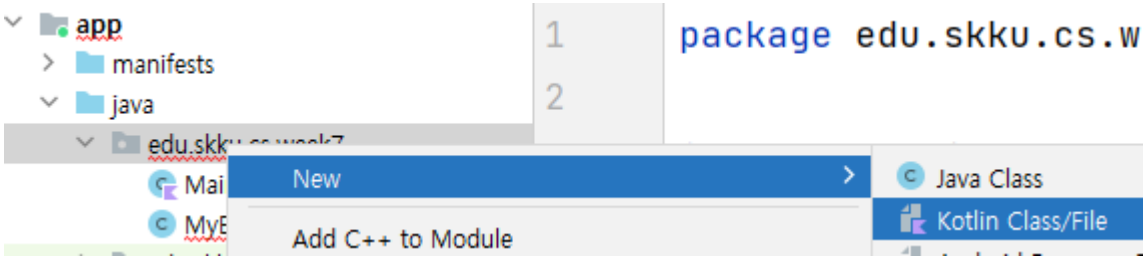
Receiver class package location

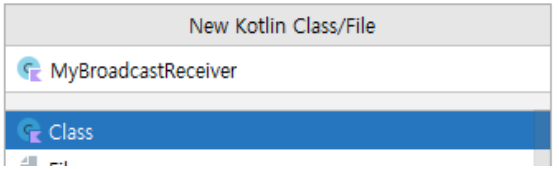
```
    </activity>

    <receiver android:name=".MyBroadcastReceiver" android:exported="true">
        <intent-filter>
            <action android:name="android.intent.action.LOCALE_CHANGED"/>
        </intent-filter>
    </receiver>
</application>
```

https://gist.github.com/devquint/d7b5d27a24fc2b8d63dee860d7293eaf

Exercise: Broadcast

- Create **BroadcastReceiver class** that prints a toast message when **LOCALE_CHANGED** action happens.



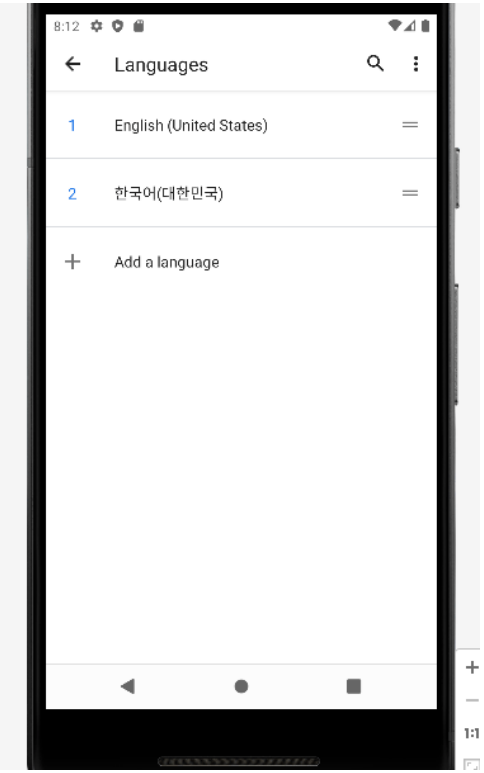


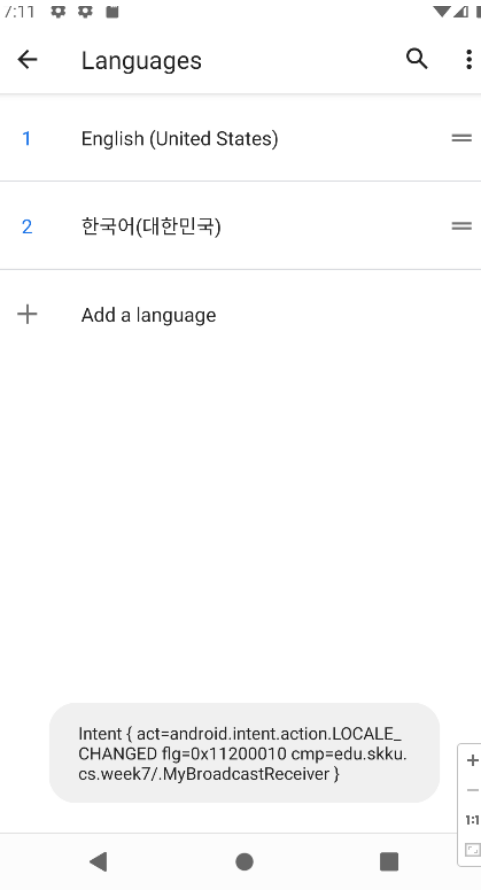

```
class MyBroadcastReceiver: BroadcastReceiver() {
    override fun onReceive(p0: Context?, p1: Intent?) {
        Toast.makeText(p0, p1.toString(), Toast.LENGTH_LONG).show()
    }
}
```

https://gist.github.com/devquint/d7b5d27a24fc2b8d63dee860d7293eaf

Exercise: Broadcast

- Run app once to install it -> Change system locale





Implicit Intent Examples

- If time of android system changes,
  - Alarm apps might change their scheduled timer
  - Android system will broadcast `android.intent.action.TIME_SET`
  - Additional user(owner of smartphone) action is not required
  - Application will: receive broadcast and do background task

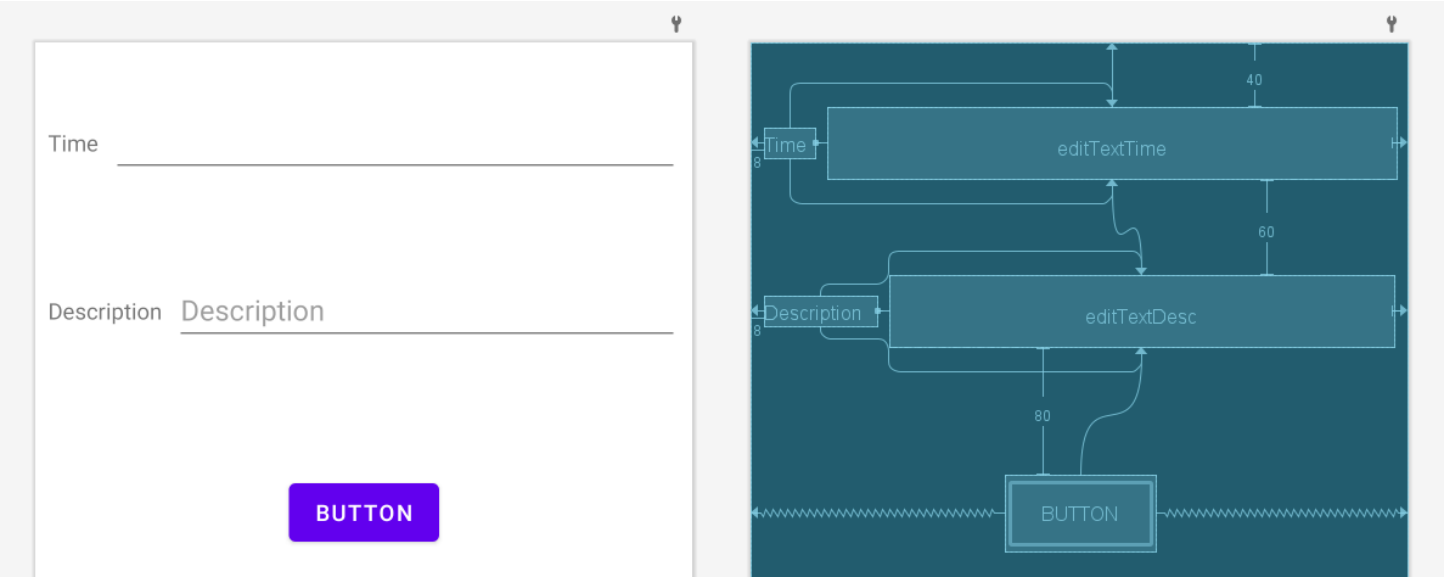
- If user tries to send email,
  - **Caller** application will send implicit intent by calling startActivity()
  - **Callee** application will: receive implicit intent and start activity
- If your application is email application (**callee**)
  - Register intent filter in AndroidManifest.xml

[Lab-Practice #7] Alarm set!

- Make an alarm setup application
- FirstActivity
  - EditText: Input hour, minute, and description
  - Button: Press to go on Second activity (Explicit intent)
- SecondActivity
  - User can check information
  - Button: Press OK button to call another alarm app. (Implicit Intent)

[Lab-Practice #7] Alarm set!

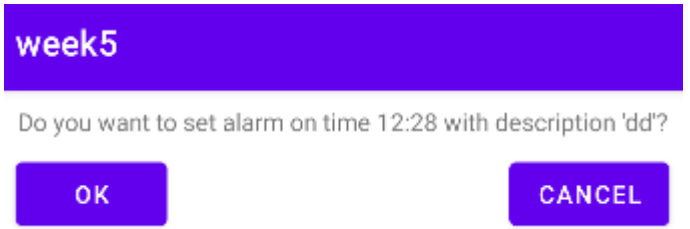
- FirstActivity

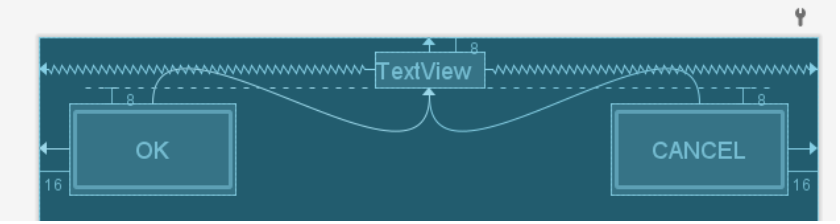


  - Textview to label, EditText to get input.
  - android:inputType="time" / "textPersonName"

[Lab-Practice #7] Alarm set!

- SecondActivity



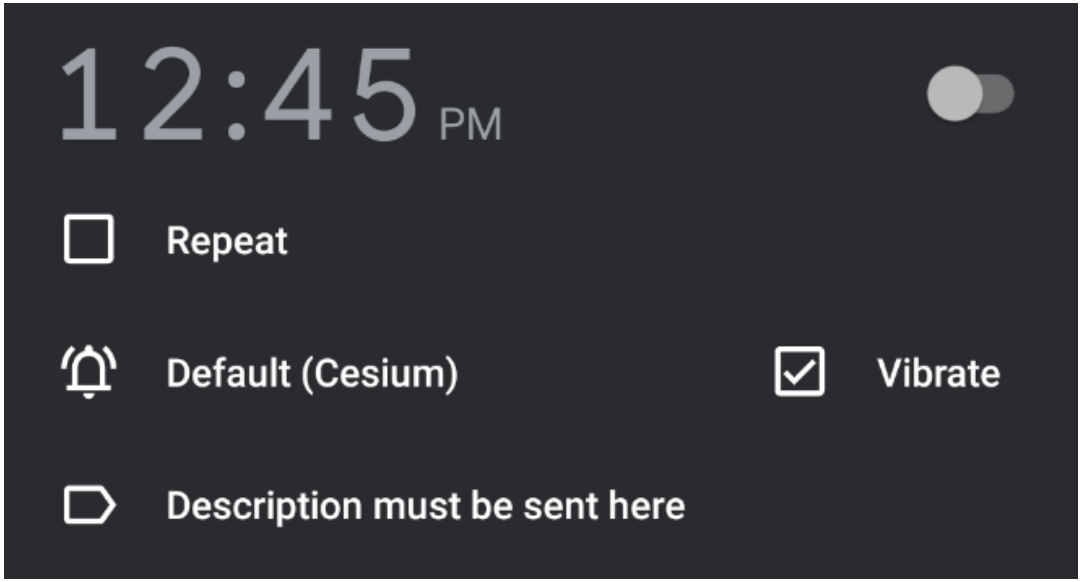


  - Get data from First activity and set TextView to
    "Do you want to set alarm on time <Time> with description '<Description>'?"

  - Cancel button: close activity
  - OK button: Send implicit intent to alarm app & close activity.

[Lab-Practice #7] Alarm set!

- Then, an alarm application will open



  - Time and description must be sent here
  - Above is emulator default alarm app for Pixel 2 API 30

[Lab-Practice #7] Alarm set!

- **TIPS**
  - Alarm app intent
    - https://developer.android.com/guide/components/intents-common?hl=ko
    - AndroidManifest.xml (**cannot autocomplete**)

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.skku.cs.week5">
    <uses-permission android:name="com.android.alarm.permission.SET_ALARM" />
    <queries>
        <intent>
            <action android:name="android.intent.action.SET_ALARM" />
        </intent>
    </queries>
    <application
```

      - <uses-permission android:name="com.android.alarm.permission.SET_ALARM" />
      - <action android:name="android.intent.action.SET_ALARM" />
      - These are queries, **NOT** intent filter

[Lab-Practice #7] Alarm set!

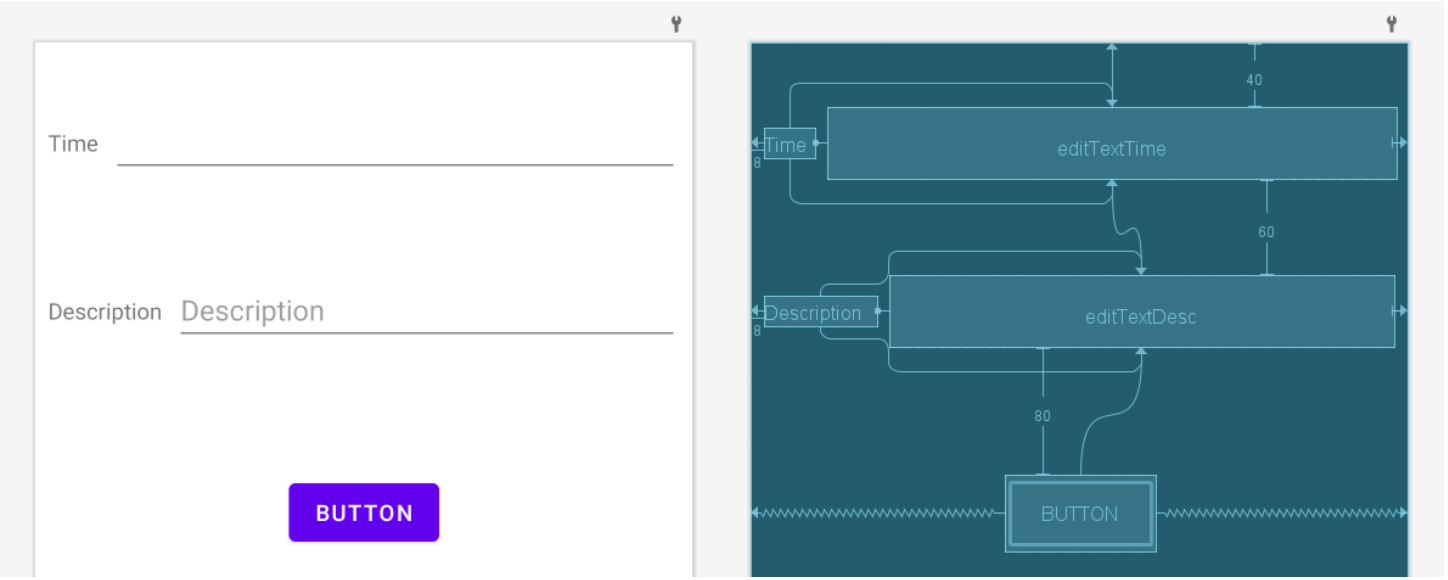
- **TIPS**
  - **ACTION_SET_ALARM** action details
    - Intent(AlarmClock.ACTION_SET_ALARM)
    - Pass **Extra** values! Check key and value below.
      - » **KEY** : AlarmClock.EXTRA_MESSAGE
        VALUE : Message, String
      - » **KEY** : AlarmClock.EXTRA_HOUR
        VALUE : Alarm hour, int, 0~23
      - » **KEY** : AlarmClock.EXTRA_MINUTES
        VALUE : Alarm minute, int, 0~59

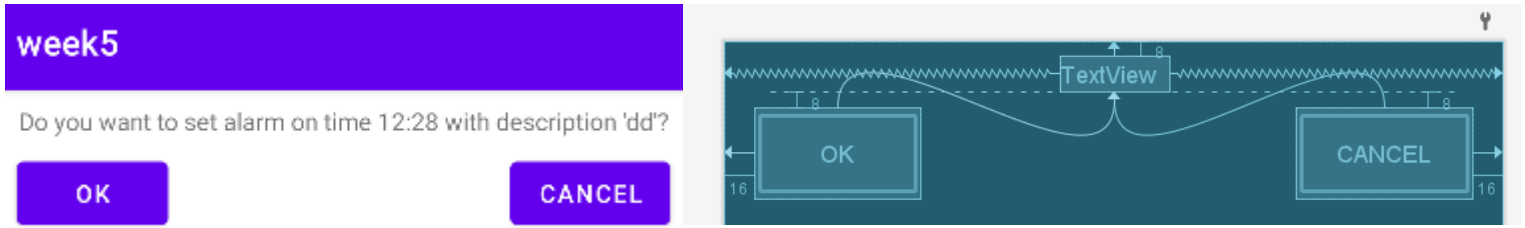
[Lab-Practice #7] Alarm set!

- **TIPS**
  - Close activity
    - this.finish()
  - Slice (hour:minute) to (hour) and (minute)
    - .split(“:”)[0]
    - .split(“:”)[1]
  - Parse String to int
    - “string_to_parse”. toInt()
    - “123”.toInt() // 123

[Lab-Practice #7] Alarm set!

- **TIPS**
  - We do NOT care the detail UI setup today.





[Lab-Practice #7] Alarm set!

- Criteria
  - UI is following the given guideline.
  - Time and description data must pass to second activity
  - Time and description data must pass to alarm application.
  - When pressing **back button** on alarm app, and re-entering to first activity, **EditText should be cleared.**
  - Input error checking is not needed (form is always HH:MM and valid)
  - Execution
    - **Write time and description, and press button: Check second activity opens and textview have proper data**
    - **Press ok: Check alarm application opens and alarm set**
    - **Press back button: Check go back to FIRST activity**
    - **Write time and description, and press button again**
    - **Press cancel button: Check go back to first activity**