Mobile Phone Security

Dr. Digvijaysinh Rathod
Associate Professor
(Cyber Security and Digital Forensics)
Institute of Forensic Science
Gujarat Forensic Sciences University

digvijay.rathod@gfsu.edu.in

Session – II

Android Application framework

Android Application Folder Structure

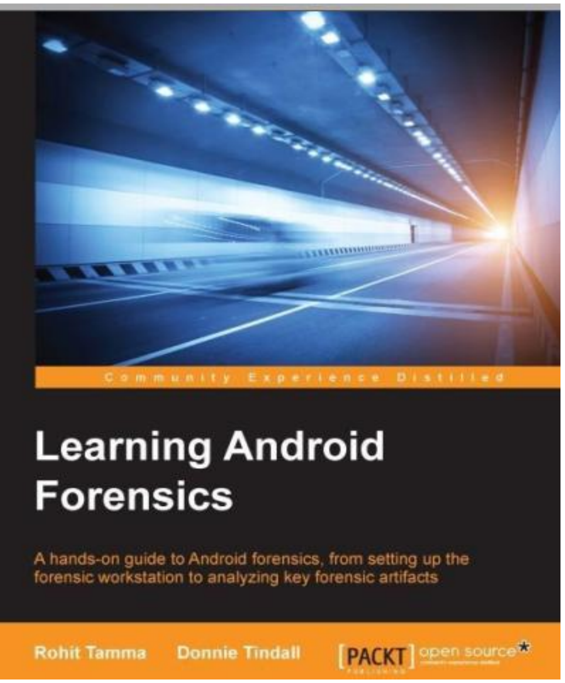
AndroidManifest.xml

Resource

Activity

Intent

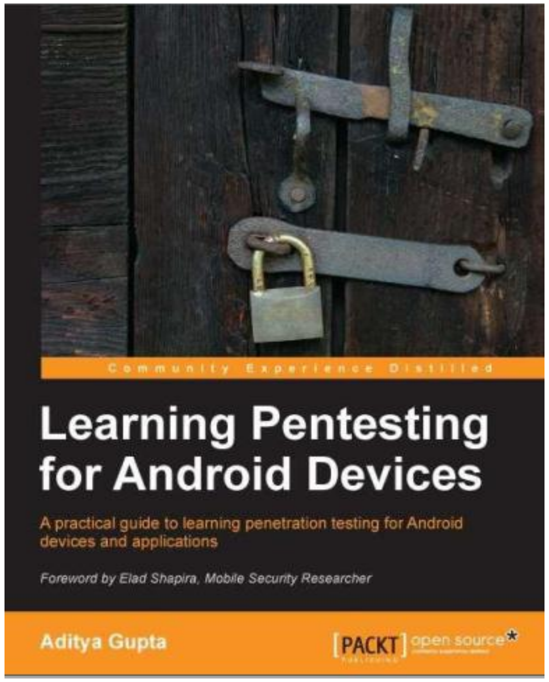
Reference


Community Experience Distilled

Learning Android Forensics

A hands-on guide to Android forensics, from setting up the forensic workstation to analyzing key forensic artifacts

Rohit Tamma Donnie Tindall [PACKT] open source*


Community Experience Distilled

Learning Pentesting for Android Devices

A practical guide to learning penetration testing for Android devices and applications

Foreword by Elad Shapira, Mobile Security Researcher

Aditya Gupta [PACKT] open source*

www.developer.google.com

Android Application Development Lab Confi.

- ✓Install JDK – Java Development Kit : Latest is Java SE Development Kit 8 for 32 bit / 64 bit download – download from oracle side.
- ✓Download Android Studio – www.developer.android.com. This is also useful to refer and documentation related to android cocepts.

Android Application Components

- ✓Four main component that can be used within an Android Application
  - ✓**Activity** – UI and handle user integration with mobile phone.
  - ✓**Service** – Background processing associated with android application.
  - ✓**Broadcast Receivers** – Handle communication between Android OS and Applicaton.
  - ✓**ContentProviders-** handle data and database managements operations.

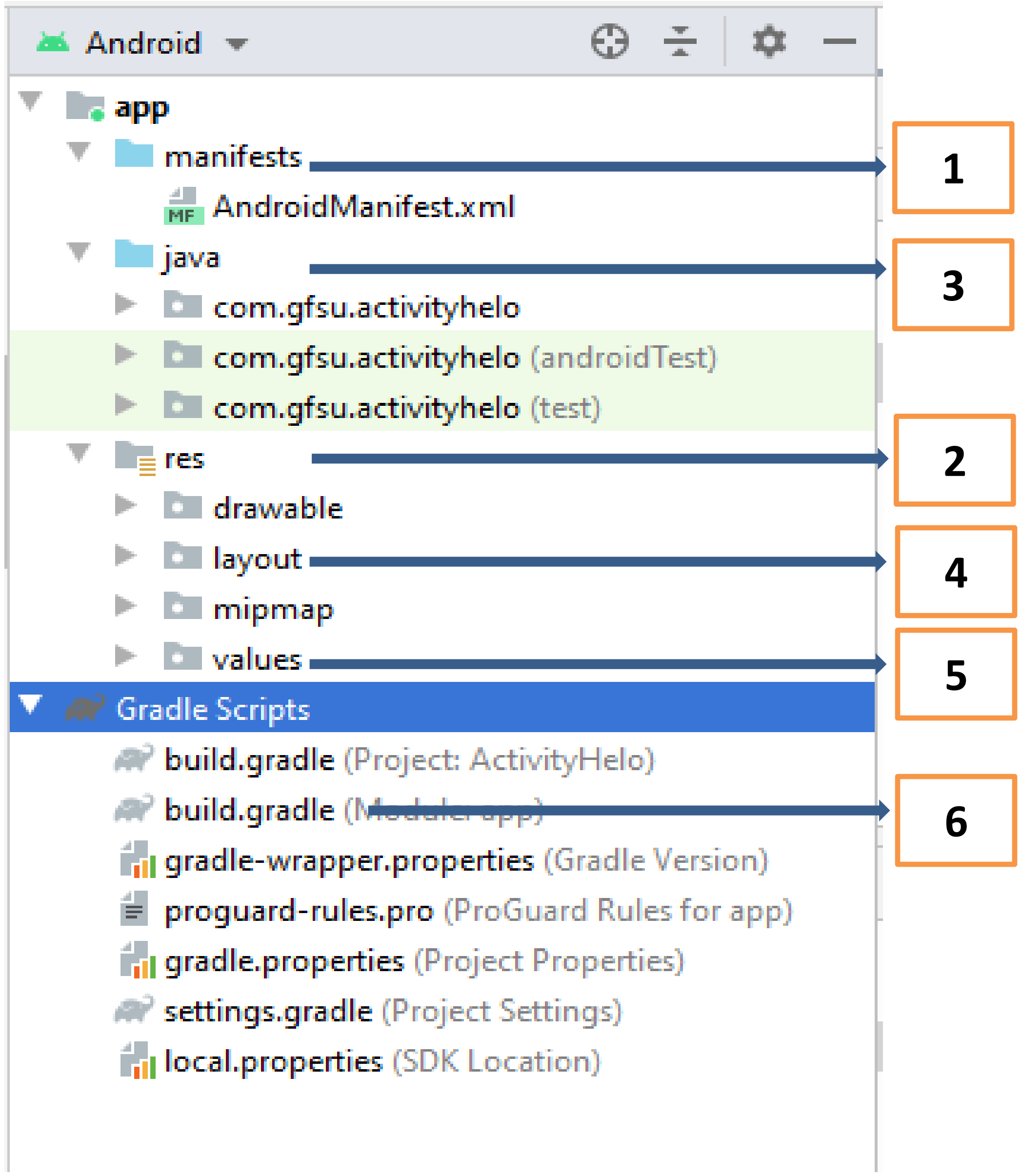
Android Application Components Example

- ✓ Alarm Application
  - ✓You open the alarm application and set the alarm using UI – **Activity**
  - ✓Data will be saved – **ContentProviders**
  - ✓Service which continuously looking in the background for set time – **Service**
  - ✓Once set alarm time matched, alarm event will ring the alarm – Event handling – **Broadcast Receiver**

Android Application- Demo

- ✓Create new android application.
- ✓Project
- ✓AndroidManifest.xml

Android Application Framework



```
Android
▼ app
  ▼ manifests
      AndroidManifest.xml
  ▼ java
    ▶ com.gfsu.activityhelo
    ▶ com.gfsu.activityhelo (androidTest)
    ▶ com.gfsu.activityhelo (test)
  ▼ res
    ▶ drawable
    ▶ layout
    ▶ mipmap
    ▶ values
▼ Gradle Scripts
    build.gradle (Project: ActivityHelo)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for app)
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

Android Application Framework
App Manifest Overview

- ✓Every app project must have an AndroidManifest.xml file (with precisely that name) at the **root of the project** source set.
- ✓The manifest file describes essential information about your app to the **Android build tools, the Android operating system, and Google Play.**
- ✓**Manifest file is required to declare the following:**
  - ✓The components of the app, which include all **activities, services, broadcast receivers, and content providers.**

Android Application Framework
App Manifest Overview

✓**Manifest file is required to declare the following:**

✓The app's package name, which usually matches your **code's namespace**. The **Android build** tools use this to determine the location of code entities when building your project.

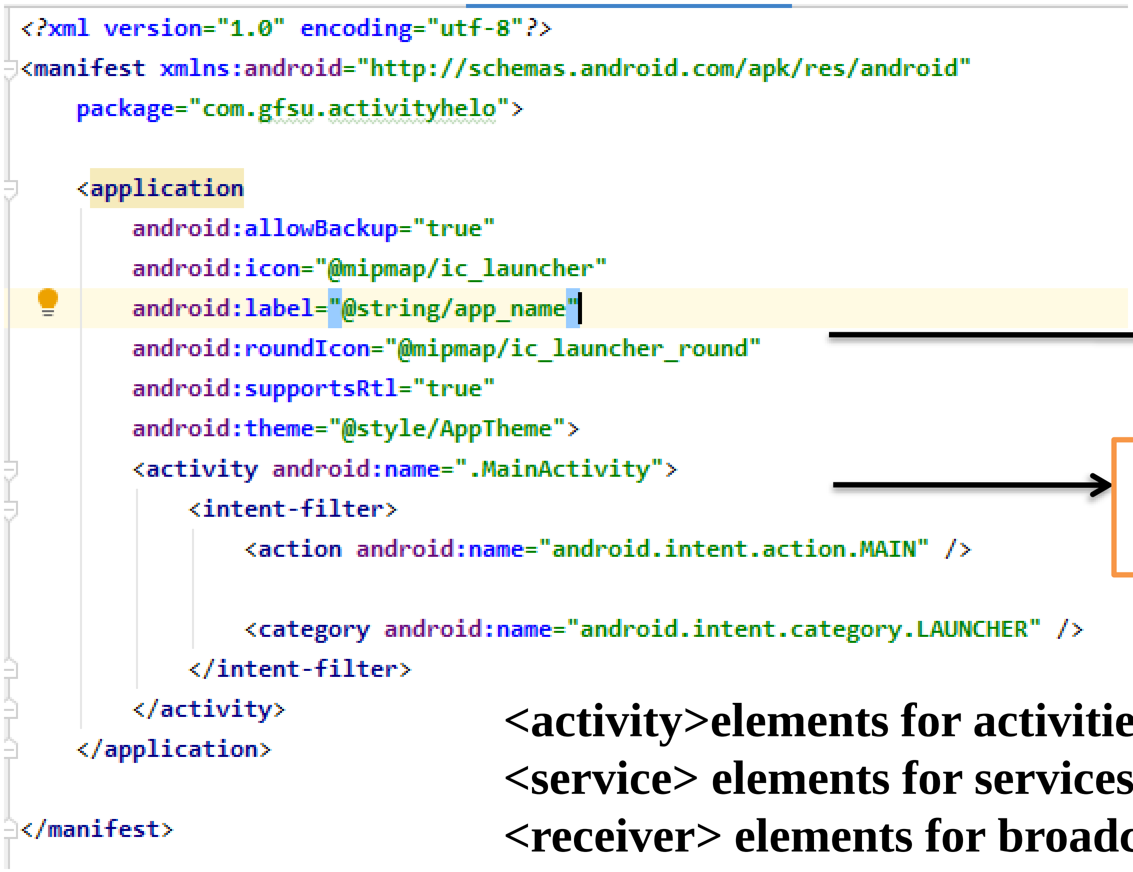
✓The **permissions** that the app needs in order to access protected parts of the system or other apps.

✓The manifest file is also where you can declare what **types of hardware or software features your app requires**, and thus, which types of devices your app is compatible with.

Android Application Framework - Example

✓Points to be discussed – with empty activity

✓AndroidManifest.xml



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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

1

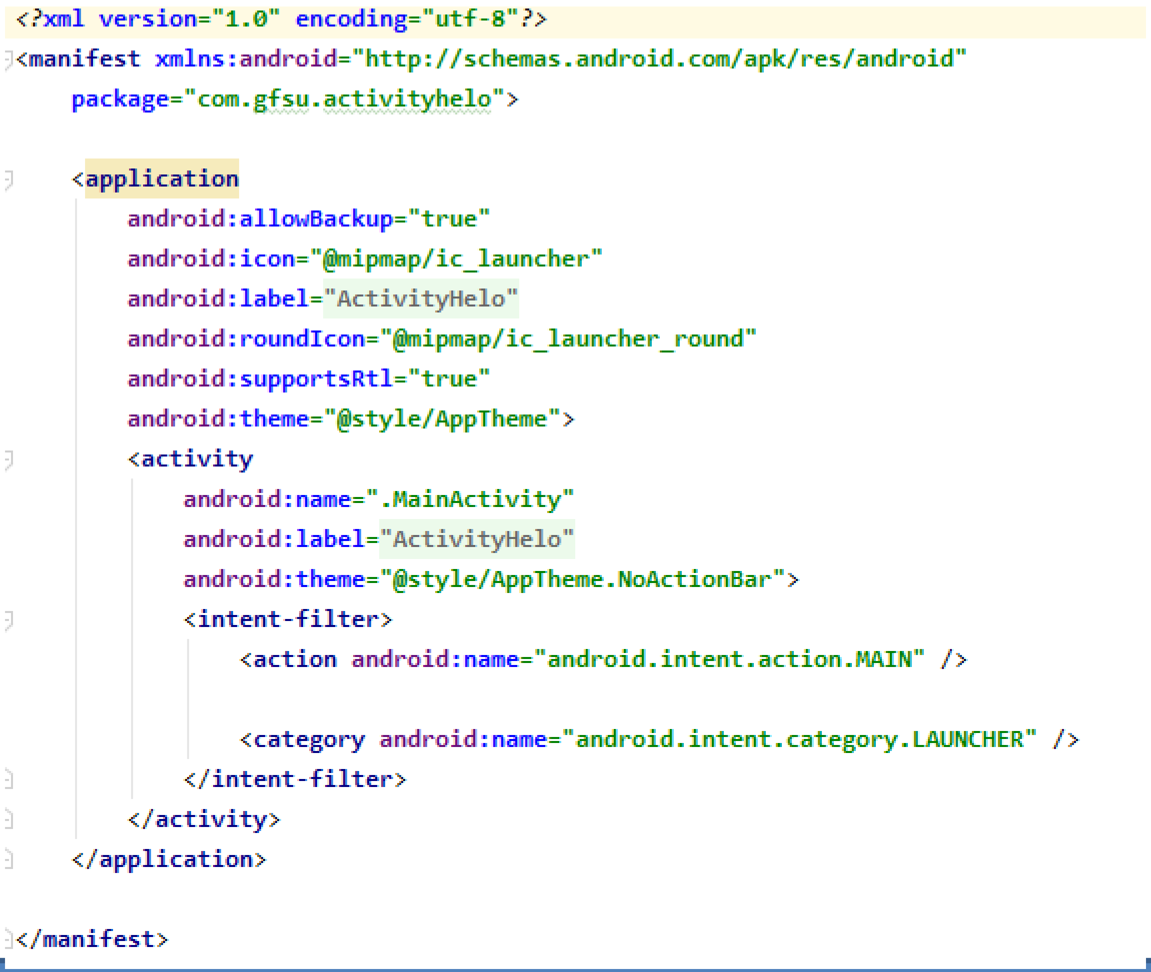
2 - Which other tag we can use here?

<activity>elements for activities
<service> elements for services
<receiver> elements for broadcast receivers
<provider> elements for content providers

Android Application Framework - Example

✓Points to be discussed – with one activity

✓AndroidManifest.xml

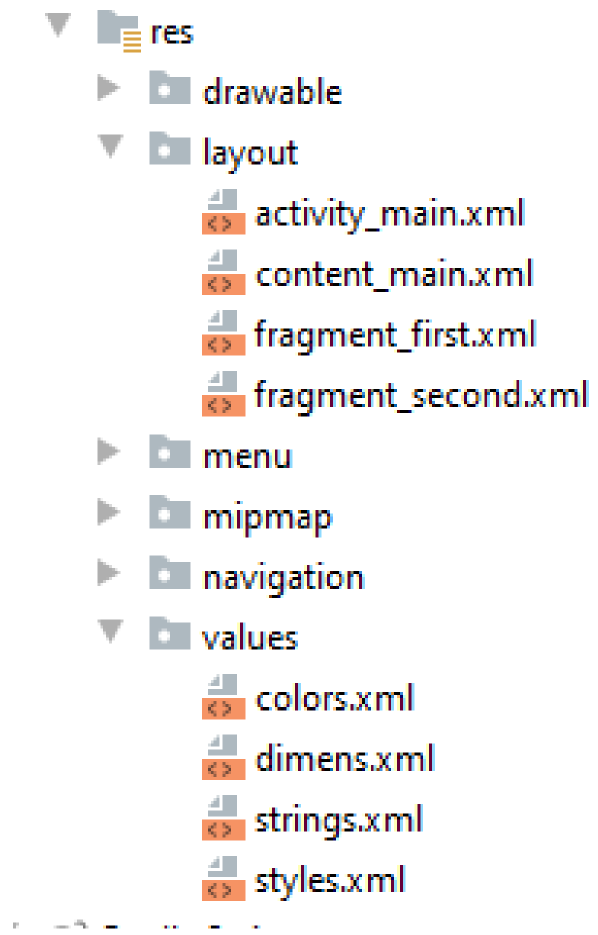


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.gfsu.activityhelo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="ActivityHelo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="ActivityHelo"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Android Application Framework - Android Application Component

- ✓Points to be discussed – with one activity
  - ✓AndroidManifest.xml
    - ✓Following is the list of tags which you will use in your manifest file to specify different Android application components –
    - ✓<activity>elements for activities
    - ✓<service> elements for services
    - ✓<receiver> elements for broadcast receivers
    - ✓<provider> elements for content providers

Android Application Framework - App resources overview



```
res
  drawable
  layout
    activity_main.xml
    content_main.xml
    fragment_first.xml
    fragment_second.xml
  menu
  mipmap
  navigation
  values
    colors.xml
    dimens.xml
    strings.xml
    styles.xml
```

✓Resources are the **additional files and static content** that your code uses, such as **bitmaps, layout definitions, user interface strings,** animation instructions, and more.

✓drawable/Bitmap files (.png, .9.png, .jpg, .gif)

✓layout/XML files that define a user interface layout.

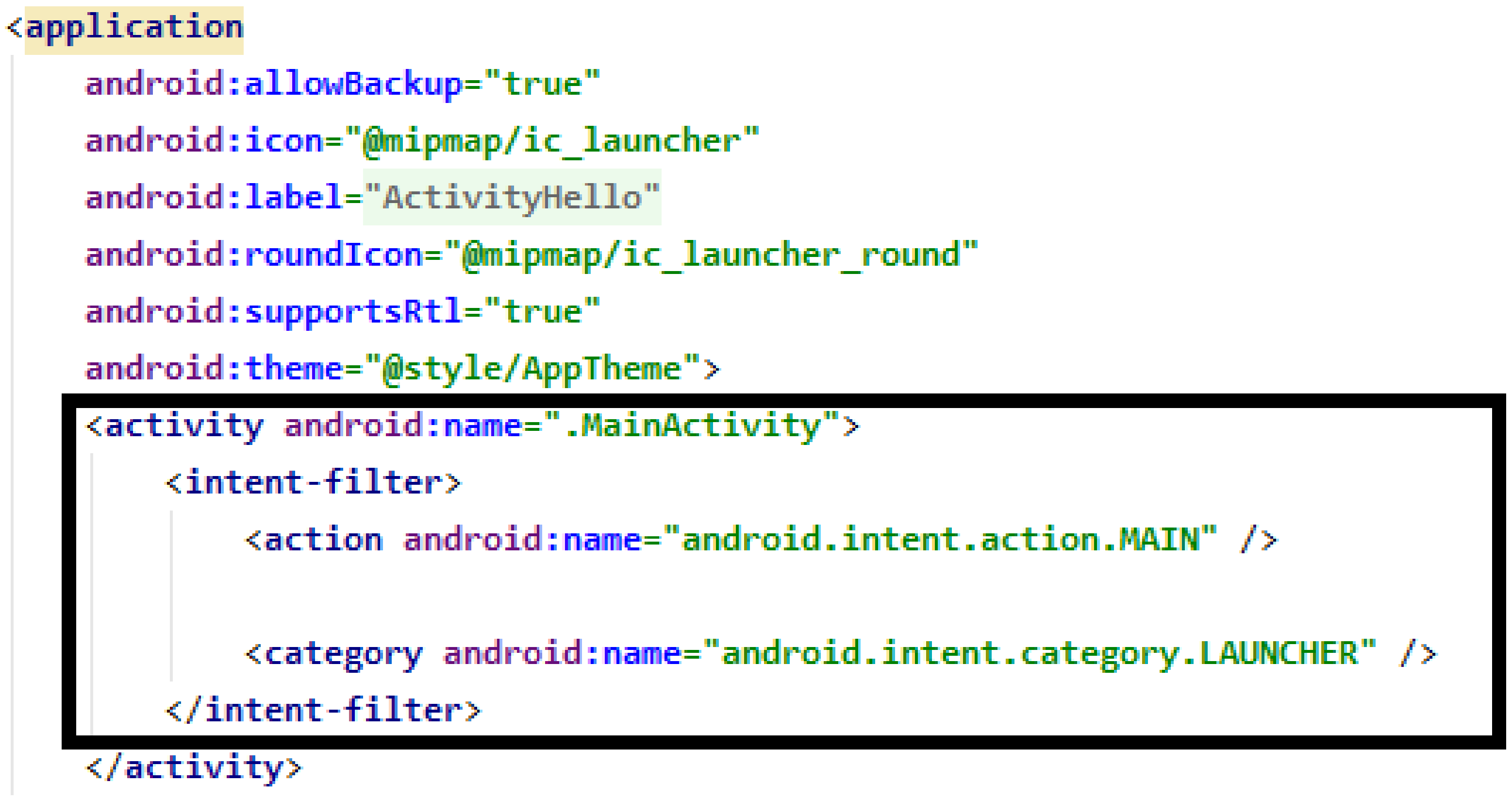
✓menu/XML files that define app menus, such as an Options Menu, Context Menu, or Sub Menu.

✓values/XML files that contain simple values, such as strings, integers, and colors.

Ref: https://developer.android.com/guide/topics/resources/providing-resources

Android Application Framework - Activity

✓An activity represents a single screen with a user interface just like window or frame.



```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="ActivityHello"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

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

Android Application Framework - Activity

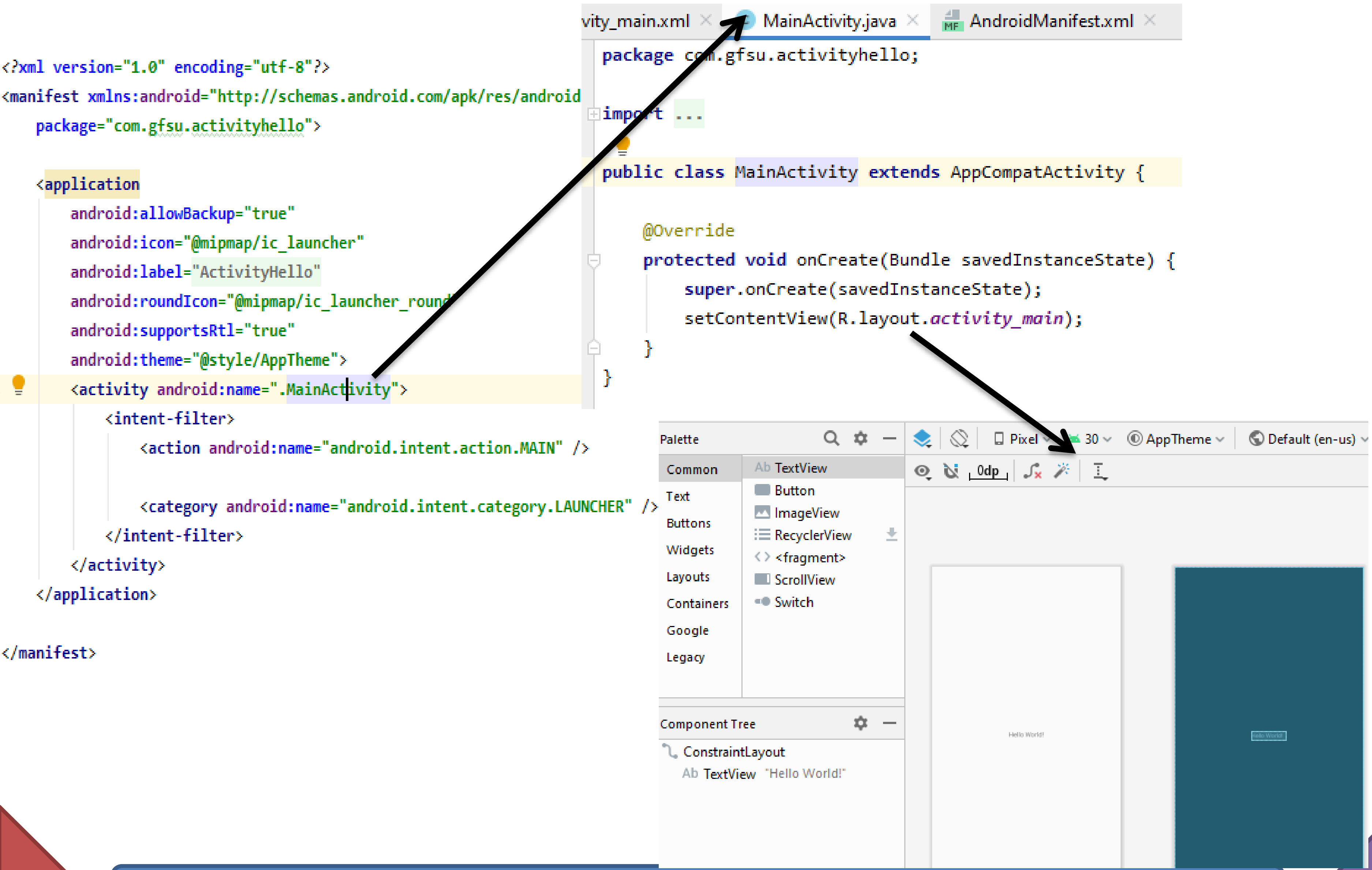
✓An activity represents a **single screen** with a **user interface** just like window or frame.

✓An application can have **one or more activities** without any restrictions.

✓Every activity you **define** for your application must be declared in your **AndroidManifest.xml file** and

✓ the main activity for your app must be declared in the manifest with an **<intent-filter>** that includes the **MAIN action and LAUNCHER** category.

Android Application Framework - Activity



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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="ActivityHello"
        android:roundIcon="@mipmap/ic_launcher_round
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

```java
package com.gfsu.activityhello;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

The application framework : Intent

✓An Intent is a **messaging object** you can use to request an action from another app component.

✓Although intents facilitate **communication** between components in several ways, there are three fundamental use cases:

✓It can be used with **startActivity to launch an Activity, broadcastIntent** to send it to any interested **BroadcastReceiver** components, and **startService(Intent)** or **bindService(Intent, ServiceConnection, int)** to communicate with a background Service.

The application framework : Intent

✓ An Intent object can contain the following components

1. **Action:** This is mandatory part of the Intent object and is a string naming the action to be performed.
2. **Category :** The category is an optional part of Intent object and it's a string containing additional information about the kind of component that should handle the intent.

The application framework : Intent

✓There are two types of intents:

1. **Explicit** intents specify which application will satisfy the intent, by supplying either the target app's package name or a fully-qualified component class name.

   You'll typically use an **explicit intent** to start a component in your **own app,** because you know the **class name** of the activity or service you want to start. For example, you might start a new activity within your app in response to a user action, or start a service to download a file in the background.

The application framework : Intent

- ✓There are two types of intents:

  2. Implicit intents do not name a specific component, but instead declare a **general action to perform**, which allows a component from **another app** to handle it. For example, if you want to show the **user a location on a map,** you can use an **implicit intent** to request that another capable app show a specified **location on a map.**

Mobile Phone Security

Dr. Digvijaysinh Rathod
Associate Professor
(Cyber Security and Digital Forensics)
Institute of Forensic Science
Gujarat Forensic Sciences University

digvijay.rathod@gfsu.edu.in