Android App Development Basics

App Fundamentals

- Some of the programming languages used in Android development are the following:
  - **C++** is close to C# and Java, it makes it easy for programmers to switch to C++ or vice versa.
  - **Swift** is the official programming language for iOS.
  - **Kotlin** combines object-oriented and functional programming features. Kotlin uses two different keywords to declare variables: **val** (never changes) and **var** (can change).
  - Java
  - C#
- The app components are the essential building blocks of an Android app.
  - **Activity** is the entry point for interacting with the user. It represents a single screen with a user interface. An example is an activity that shows a list of new emails.
  - **Service** runs in the background to perform long-running operations or to perform work for remote processes. An example is playing music in the background while using a different app.
  - **Broadcast receiver** is a component that enables the system to deliver events to the app outside of a regular user flow, allowing the app to respond to system-wide broadcast announcements. An example is a broadcast indicating that the battery is low.
  - **Content provider** manages a shared set of app data that you can store which your app can access. An example is a content provider that manages the user's contact information and is requested by another app to read information about a particular person.
- App activities, services, and broadcast receivers are activated by intents.
- An **intent** is a message defined by an Intent object that describes an action to perform, including the data to be acted upon, the category of component that should perform the action, and other instructions.

The Manifest File

- The **manifest file** (AndroidManifest.xml) details the components that exists in an app.
- The **`<manifest>`** element is the root element of the AndroidManifest.xml file.
- The **`<application>`** element contains sub-elements that declare each of the application's components and has attributes that can affect all the components.
- The required elements in the manifest file are `<manifest>` and `<application>`, which should occur only once.
- The `<application>` element must be the last element inside the `<manifest>` element.
- The icon and label that are set in the **`<application>`** element are the default icon and label for each of the app's components.

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

    <dist:module dist:instant="true" />

    <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>
```

Figure 1. The Default AndroidManifest.xml File

- The following must be declared in a manifest file:
    - The app's package name
      Ex. **`package="com.example.myapplication"`**
    - The app's components:
        - **`<activity>`** for each subclass of `Activity`
        - **`<service>`** for each subclass of `Service`
        - **`<receiver>`** for each subclass of `BroadcastReceiver`
        - **`<provider>`** for each subclass of `ContentProvider`

      Ex. **`<activity android:name=".MainActivity">`**
    - The permissions that the app needs in order to access protected parts of the system or other apps
      Ex. **`<uses-permission android:name="android.permission.SEND_SMS"/>`**
    - The hardware and software features used or required by the app
      Ex. **`<uses-feature android:name="android.hardware.sensor.compass" android:required="true" />`**
- The **`<intent-filter>`** element specifies the types of intents that an activity, service, or broadcast receiver can respond to.
- The **`<action>`** element adds an action to an intent filter while **`<category>`** adds a category name to an intent filter.

References:

DiMarzio, J. (2017). *Beginning Android programming with Android Studio*. Indiana: John Wiley & Sons, Inc.

Google Developers Training Team. (2018). *Android developer fundamentals (version 2)*. Retrieved from https://google-developer-training.github.io

Android Developers (n.d.). Citing sources. Retrieved from https://developer.android.com/