**TASK NO.1**

Create Simple Hello World! App Run on android virtual device and on your android phone.

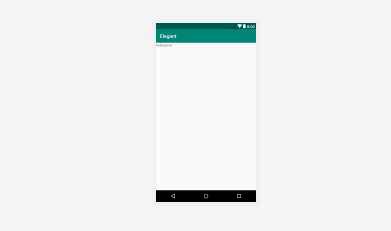
**Activitymain.xml**

<?**xml version="1.0" encoding="utf-8"**?>  
<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"**>  
  
 <**TextView  
 android:id="@+id/hw"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="helloworld"** />  
  
</**LinearLayout**>

**Mainativity.java**

**package** com.example.elegant;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.**activity\_main**);  
 }  
}

**OUTPUT:**



**TASK NO.2**

Why we need separate folder for Assets and Resources if both folder are for same purpose?

**SOLUTION:**

Both are pretty similar. The real main difference between the two is that in the res directory each file is given a pre-compiled ID which can be accessed easily through R.id.[res id]. This is useful to quickly and easily access images, sounds, icons.

The assets directory is more like a filesystem and provides more freedom to put any file you would like in there. You then can access each of the files in that system as you would when accessing any file in any file system through Java. This directory is good for things such as game details, dictionaries,...etc. Hope that helps.

**assets/**

This is empty. You can use it to store raw asset files. Files that you save here are compiled into an .apk file as-is, and the original filename is preserved. You can navigate this directory in the same way as a typical file system using URIs and read files as a stream of bytes using the AssetManager. For example, this is a good location for textures and game data.

**res/raw/**

For arbitrary raw asset files. Saving asset files here instead of in the assets/ directory only differs in the way that you access them. These files are processed by aapt and must be referenced from the application using a resource identifier in the R class. For example, this is a good place for media, such as MP3 or Ogg files.

**Conclusion**

* Android API includes a very comfortable Resources framework that is also optimized for most typical use cases for various mobile apps. You should master Resources and try to use them wherever possible.
* However, if you need more flexibility for your special case, Assets are there to give you a lower level API that allows organizing and processing your resources with a higher degree of freedom.

**TASK NO.3**

Explain the directory structure of simple Hello World app created in task 1 . **SOLUTION:**

|  |  |
| --- | --- |
| **Sr.No.** | **Folder, File & Description** |
| 1 | **Java**  This contains the **.java** source files for your project. By default, it includes an *MainActivity.java* source file having an activity class that runs when your app is launched using the app icon. |
|  |  |
| 2 | **res/drawable-hdpi**  This is a directory for drawable objects that are designed for high-density screens. |
| 3 | **res/layout**  This is a directory for files that define your app's user interface. |
| 4 | **res/values**  This is a directory for other various XML files that contain a collection of resources, such as strings and colours definitions. |
| 5 | **AndroidManifest.xml**  This is the manifest file which describes the fundamental characteristics of the app and defines each of its components. |
| 6 | **Build.gradle**  This is an auto generated file which contains compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionCode and versionName |