**Lab 27: Android WebView using Kotlin**

# **Introduction**

The WebView view in your programme shows web pages. Additionally, you can specify an HTML string and use WebView to display it inside your programme. Your programme becomes a web application when you use WebView. You must include the WebView> element in your xml layout file in order to add WebView to your application.

**Let’s get Started:**

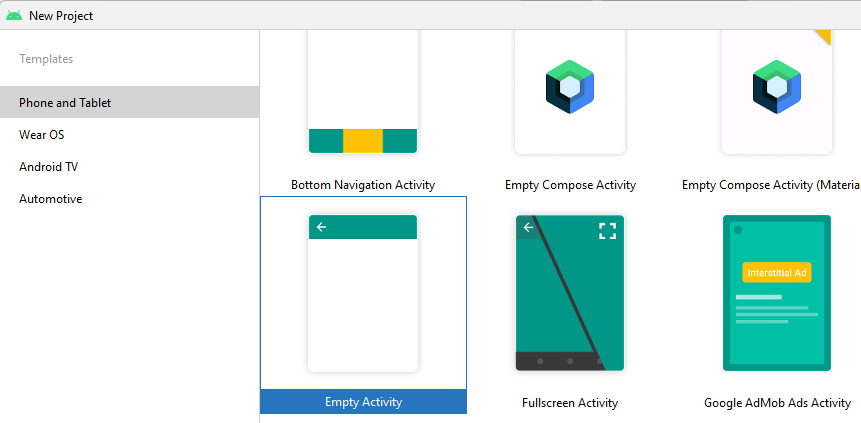
In this experiment we will develop an Android App to demonstrate the use of Android WebView.

**Step 1: Create a New Project in Android Studio as shown below**

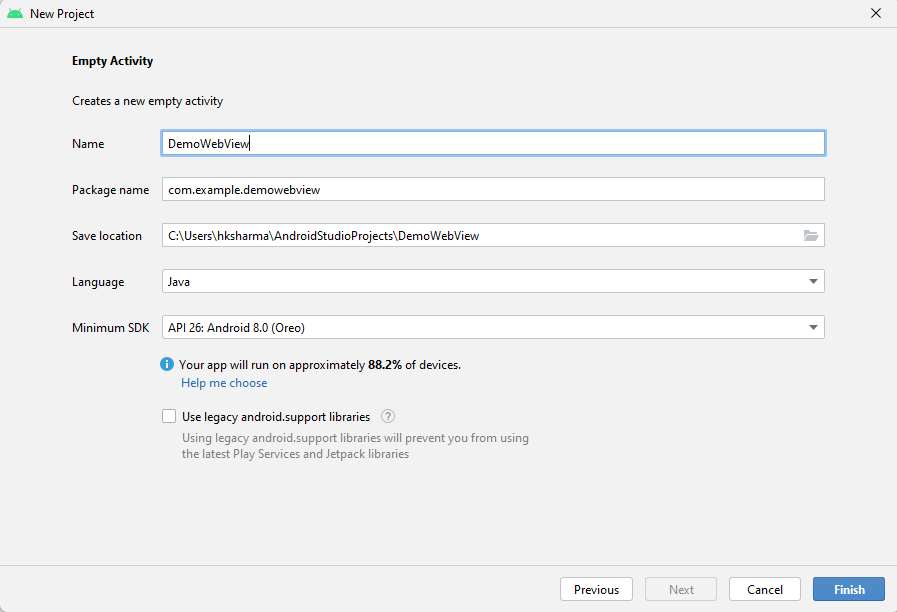
Graphical user interface, text, application

Description automatically generated

**Step 2: Select Empty Activity as shown below**



**Step 3: Provide a Project Name as shown below**

****

**Step 4: Update MainActivity.kt as per the code given below**

**package** com.example.kotlinwebview  
**import** android.os.Bundle  
**import** android.webkit.WebViewClient  
**import** androidx.appcompat.app.AppCompatActivity  
**import** kotlinx.android.synthetic.main.activity\_main.\*  
  
**class** MainActivity : AppCompatActivity() {  
  
 **override fun** onCreate(savedInstanceState: Bundle?) {  
 **super**.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 *// WebViewClient allows you to handle  
 // onPageFinished and override Url loading.* webView.*webViewClient* = WebViewClient()  
  
 *// this will load the url of the website* webView.loadUrl(**"https://www.snap.com/"**)  
  
 *// this will enable the javascript settings, it can also allow xss vulnerabilities* webView.*settings*.*javaScriptEnabled* = **true** *// if you want to enable zoom feature* webView.*settings*.setSupportZoom(**true**)  
 }  
  
 *// if you press Back button this code will work* **override fun** onBackPressed() {  
 *// if your webview can go back it will go back* **if** (webView.canGoBack())  
 webView.goBack()  
 *// if your webview cannot go back  
 // it will exit the application* **else  
 super**.onBackPressed()  
 }  
}

**Step 5: activity\_main.xml**

<**androidx.constraintlayout.widget.ConstraintLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"**>  
  
 *<!-- Place Web-view on the Screen -->* <**WebView  
 android:id="@+id/webView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**/>  
  
</**androidx.constraintlayout.widget.ConstraintLayout**>

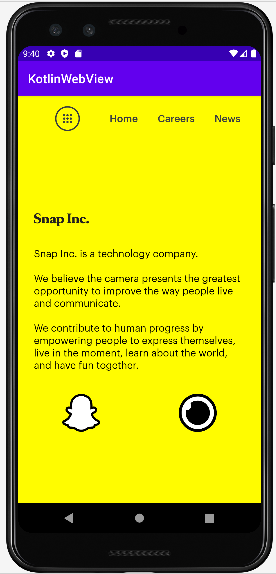
**Step 6: Add these Lines in gradle files**

plugins **{** id **'com.android.application'** id **'kotlin-android'** id **'kotlin-android-extensions'  
}**

**Step 7: Add permission in AndroidManifest file**

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"**>  
 <**uses-permission android:name="android.permission.INTERNET"**/>  
  
 <**application  
 android:allowBackup="true"  
 android:dataExtractionRules="@xml/data\_extraction\_rules"  
 android:fullBackupContent="@xml/backup\_rules"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.KotlinWebView"  
 tools:targetApi="31"**>  
 <**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**>  
  
 <**meta-data  
 android:name="android.app.lib\_name"  
 android:value=""** />  
 </**activity**>  
 </**application**>  
  
</**manifest**>

**Step 8: Check Output on Android Emulator and it should look like as given below**



**Voila!!** We have successfully completed this lab.