**Lab 29: JSON Parsing using Kotlin**

# **Introduction**

We will Learn XML/DOM parsing in Android using Kotlin

**Let’s get Started**

In this experiment we will develop an Android App to demonstrate the use of Android DOM/XML Parsing.

**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**

Graphical user interface, application, shape

Description automatically generated

**Step 3: Update MainActivity.kt as per the code given below**

**package** com.example.domparser  
**import** android.os.Bundle  
**import** android.widget.ListAdapter  
**import** android.widget.ListView  
**import** android.widget.SimpleAdapter  
**import** androidx.appcompat.app.AppCompatActivity  
**import** org.w3c.dom.Document  
**import** org.w3c.dom.Element  
**import** org.w3c.dom.Node  
**import** org.w3c.dom.NodeList  
**import** org.xml.sax.SAXException  
**import** java.io.IOException  
**import** java.io.InputStream  
**import** javax.xml.parsers.DocumentBuilder  
**import** javax.xml.parsers.DocumentBuilderFactory  
**import** javax.xml.parsers.ParserConfigurationException  
  
**open class** MainActivity : AppCompatActivity() {  
 **override fun** onCreate(savedInstanceState: Bundle?) {  
 **super**.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 *// Try and Catch for avoiding the application to crash* **try** {  
  
 *// This list will contain the data from the information.xml file* **val** userList: ArrayList<HashMap<String, String?>> = ArrayList()  
  
 *// This listView will display the data from the information.xml file* **val** lv = findViewById<ListView>(R.id.*user\_list*)  
  
 *// The information.xml file will be taken in the form of input stream* **val** istream: InputStream = *assets*.open(**"information.xml"**)  
  
 *// Steps to convert this input stream into a list* **val** builderFactory: DocumentBuilderFactory = DocumentBuilderFactory.newInstance()  
 **val** docBuilder: DocumentBuilder = builderFactory.newDocumentBuilder()  
 **val** doc: Document = docBuilder.parse(istream)  
 **val** nList: NodeList = doc.getElementsByTagName(**"user"**)  
  
 *// Iterating through this list* **for** (i **in** 0 *until* nList.*length*) {  
 **if** (nList.item(0).*nodeType* === Node.*ELEMENT\_NODE*) {  
 **val** user: HashMap<String, String?> = HashMap()  
 **val** elm: Element = nList.item(i) **as** Element  
 user[**"name"**] = getNodeValue(**"name"**, elm)  
 user[**"designation"**] = getNodeValue(**"designation"**, elm)  
 userList.add(user)  
 }  
 }  
  
 *// Using Adapter to broadcast the information extracted* **val** adapter: ListAdapter = SimpleAdapter(  
 **this**,  
 userList,  
 R.layout.*list*,  
 *arrayOf*(**"name"**, **"designation"**),  
 *intArrayOf*(R.id.*name*, R.id.*designation*)  
 )  
 lv.*adapter* = adapter  
 } **catch** (e: IOException) {  
 e.printStackTrace()  
 } **catch** (e: ParserConfigurationException) {  
 e.printStackTrace()  
 } **catch** (e: SAXException) {  
 e.printStackTrace()  
 }  
 }  
  
 *// A function to get the node value while parsing* **private fun** getNodeValue(tag: String?, element: Element): String? {  
 **val** nodeList = element.getElementsByTagName(tag)  
 **val** node = nodeList.item(0)  
 **if** (node != **null**) {  
 **if** (node.hasChildNodes()) {  
 **val** child = node.*firstChild* **while** (child != **null**) {  
 **if** (child.*nodeType* == Node.*TEXT\_NODE*) {  
 **return** child.*nodeValue* }  
 }  
 }  
 }  
 *// Returns nothing if nothing was found* **return ""** }  
}

**Step 4: Update activity\_main.xml as per the code given below**

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:orientation="vertical"**>  
  
 *<!--ListView to display the list-->* <**ListView  
 android:id="@+id/user\_list"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:dividerHeight="1dp"** />  
</**LinearLayout**>

**Step 6: Create list.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:padding="5dip"**>  
  
 *<!--textView to show the name node-->* <**TextView  
 android:id="@+id/name"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textSize="17dp"  
 android:textStyle="bold"** />  
  
 *<!--textView to show the designation node-->* <**TextView  
 android:id="@+id/designation"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/name"  
 android:layout\_marginTop="7dp"  
 android:textColor="#343434"  
 android:textSize="14dp"** />  
  
</**RelativeLayout**>

**Step 7: create following information.xml in asset folder**

*<?***xml version="1.0" encoding="utf-8"***?>*<**users**>  
 <**user**>  
 <**name**>Steve</**name**>  
 <**designation**>Apple</**designation**>  
 </**user**>  
 <**user**>  
 <**name**>Sundar</**name**>  
 <**designation**>Google</**designation**>  
 </**user**>  
 <**user**>  
 <**name**>Jeff</**name**>  
 <**designation**>Amazon</**designation**>  
 </**user**>  
</**users**>

**Step 7: Check Output on Android Emulator.**

