

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

## UNIT – 4

### JSON AND XML PARSING

JSON stands for JavaScript Object Notation. It is an independent data exchange format and is the best alternative for XML. JSON stands for JavaScript Object Notation. It is an independent data exchange format and is the best alternative for XML.

Android provides four different classes to manipulate JSON data. These classes are **JSONArray,JSONObject,JSONStringer and JSNTokenizer.**

The first step is to identify the fields in the JSON data in which you are interested in.

#### **JSON - Elements**

An JSON file consist of many components. Here is the table defining the components of an JSON file and their description –

| Sr.No | Component & description  |
|-------|--|
| 1     | <b>Array([])</b><br>In a JSON file , square bracket ([]) represents a JSON array                           |
| 2     | <b>Objects({})</b><br>In a JSON file, curly bracket ({} ) represents a JSON object                         |
| 3     | <b>Key</b><br>A JSON object contains a key that is just a string. Pairs of key/value make up a JSON object |
| 4     | <b>Value</b><br>Each key has a value that could be string , integer or double e.t.c                        |

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

For example. In the JSON given below we interested in getting temperature only.

```
{
  "sys":
  {
    "country":"GB",
    "sunrise":1381107633,
    "sunset":1381149604
  },
  "weather":[
    {
      "id":711,
      "main":"Smoke",
      "description":"smoke",
      "icon":"50n"
    }
  ],
  "main":
  {
    "temp":304.15,
    "pressure":1009,
  }
}
```

## ***JSON - Parsing***

For parsing a JSON object, we will create an object of class JSONObject and specify a string containing JSON data to it. Its syntax is –

```
String in;
JSONObject reader = new JSONObject(in);
```

The last step is to parse the JSON. A JSON file consist of different object with different key/value pair e.t.c. So JSONObject has a separate function for parsing each of the component of JSON file. Its syntax is given below –

```
JSONObject sys = reader.getJSONObject("sys");
country = sys.getString("country");
```

```
JSONObject main = reader.getJSONObject("main");
temperature = main.getString("temp");
```

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

The method **getJSONObject** returns the JSON object. The method **getString** returns the string value of the specified key.

| Sr.No | Method & description   |
|-------|--|
| 1     | <b>get(String name)</b><br>This method just Returns the value but in the form of Object type |
| 2     | <b>getBoolean(String name)</b><br>This method returns the boolean value specified by the key |
| 3     | <b>getDouble(String name)</b><br>This method returns the double value specified by the key   |
| 4     | <b>getInt(String name)</b><br>This method returns the integer value specified by the key     |
| 5     | <b>getLong(String name)</b><br>This method returns the long value specified by the key       |
| 6     | <b>length()</b><br>This method returns the number of name/value mappings in this object..    |
| 7     | <b>names()</b><br>This method returns an array containing the string names in this object.   |

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

## services

Services in Android is a component that runs in the background without any user interaction or interface. Unlike Activities services does not have any graphical interface. Services run invisibly performing long running tasks - doing Internet look ups, playing music, triggering notifications etc.,

Services run with a higher priority than inactive or invisible activities and therefore it is less likely that the Android system terminates them for resource management. The only reason Android will stop a Service prematurely is to provide additional resources for a foreground component usually an Activity. When this happens, your Service can be configured to restart automatically.

### Creating a Service

Create a new class that extends Service. Below is the code snippet.

```
import android.app.Service;
```

```
import android.content.Intent;
```

```
import android.os.IBinder;
```

```
public class MyService extends Service {  
public void onCreate() {  
    super.onCreate();  
    // This method is invoked when the service is called.  
}  
public IBinder onBind() {  
    super.onCreate();  
    // A client or activity is binded to service  
}  
}
```

### **Services takes Two Forms**

#### **Started**

A Service is started when a application component such as an activity invokes it by calling `startService()`. Once started, a service can run in the background indefinitely, even if the component that started it is destroyed

#### **Bound**

A service is "bound" when an application component binds to it by calling `bindService()`. A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with inter-process communication.

# SHREE H. N. SHUKLA GROUP OF COLLEGES

(Affiliated to Saurashtra University & Gujarat Technological University)

## AndroidManifest.xml

```
<service  
    android:name=".MyService"  
    android:enabled="true" />
```

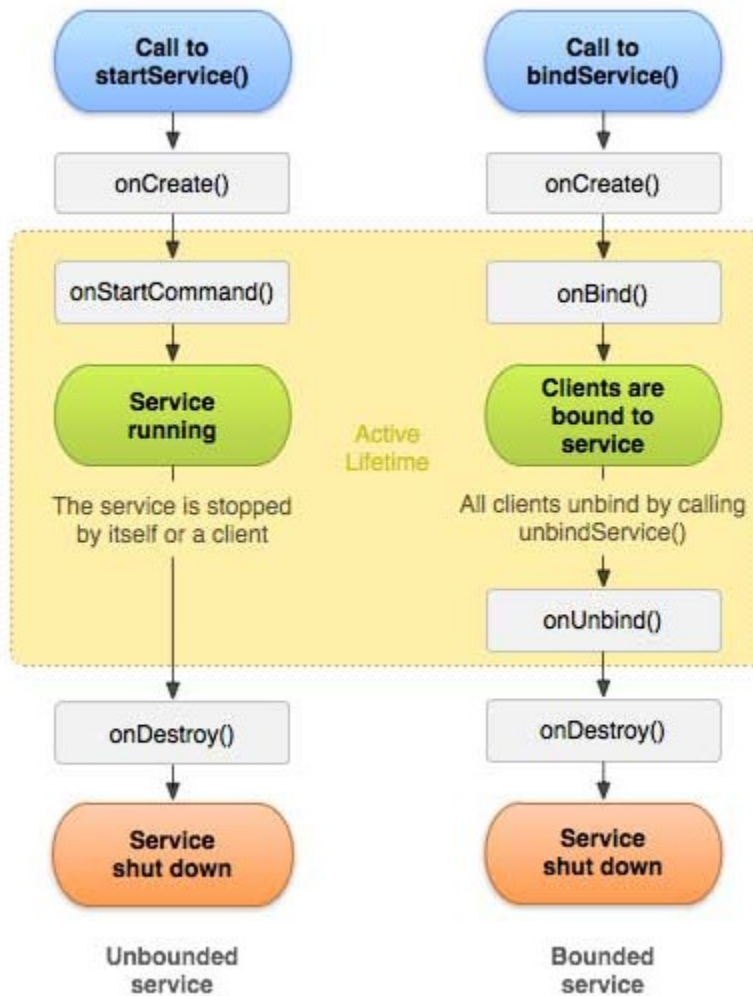
A service is a component that runs in the background to perform long-running operations without needing to interact with the user. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity. A service can essentially take two states:

| State   | Description  |
|---------|--|
| Started | A service is <b>started</b> when an application component, such as an activity, starts it by calling <code>startService()</code> . Once started, a service can run in the background indefinitely, even if the component that started it is destroyed.   |
| Bound   | A service is <b>bound</b> when an application component binds to it by calling <code>bindService()</code> . A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with interprocess communication (IPC). |

A service has lifecycle callback methods that you can implement to monitor changes in the service's state and you can perform work at the appropriate stage. The following diagram on the left shows the lifecycle when the service is created with `startService()` and the diagram on the right shows the lifecycle when the service is created with `bindService()`: (image courtesy : android.com )

# SHREE H. N. SHUKLA GROUP OF COLLEGES

(Affiliated to Saurashtra University & Gujarat Technological University)



To create an service, you create a Java class that extends the Service base class or one of its existing subclasses. The **Service** base class defines various callback methods and the most important are given below. You don't need to implement all the callbacks methods. However, it's important that you understand each one and implement those that ensure your app behaves the way users expect.

| Callback         | Description   |
|------------------|---|
| onStartCommand() | The system calls this method when another component, such as an activity, requests that the service be started, by calling startService(). If you implement this method, it is your responsibility to stop the service when its work is done, by calling stopSelf() or stopService() methods.   |
| onBind()         | The system calls this method when another component wants to bind with the service by calling bindService(). If you implement this method, you must provide an interface that clients use to communicate with the service, by returning an <i>IBinder</i> object. You must always implement this method, but if you don't want to allow binding, then you should return null. |

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

|             |   |
|-------------|---|
| onUnbind()  | The system calls this method when all clients have disconnected from a particular interface published by the service.   |
| onRebind()  | The system calls this method when new clients have connected to the service, after it had previously been notified that all had disconnected in its onUnbind(Intent).                                       |
| onCreate()  | The system calls this method when the service is first created using onStartCommand() or onBind(). This call is required to perform one-time setup.   |
| onDestroy() | The system calls this method when the service is no longer used and is being destroyed. Your service should implement this to clean up any resources such as threads, registered listeners, receivers, etc. |

---

## Service Overview

We already know that Android Activity can be started, stopped, destroyed if the system resources become too low and maybe can be recreated, a **Service are designed to have a longer life. A Service in Android can be started from an Activity, from a Broadcast receiver and other services too.**

We have to notice that using Service we don't create automatically new threads, so if we implement a simple logic inside our Service, that doesn't require long time processing we don't need to run it a separate thread, but if we have to implement complex logic, with long time processing, we have to take care of creating a new thread, otherwise the service runs on the main thread and it could cause ANR problem.

In Android a Service is used for two main reason:

- Implement multi-task
- Enable Inter-Process-Communication (IPC)

A typical example of the first case is an app that required to download data from a remote server, in this case we can have Activity that interacts with a user and starts a service that accomplishes the work in background while the user can use the app and maybe when the service finishes sends a message to the user.

In the second case, we want to "share" some common functions so that different app can re-use them. For example, we can suppose we have a service that sends an email and we want to share this service among several apps so that we don't have to rewrite the same code. In this case we can use IPC so that the service exposes a "remote" interface that can be called by other app.

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

## JSON

JSON stands for JavaScript Object Notation. It is an independent data exchange format and is the best alternative for XML. This chapter explains how to parse the JSON file and extract necessary information from it.

Android provides four different classes to manipulate JSON data. These classes are **JSONArray, JSONObject, JSONStringer and JSNTokenizer.**

In the JSON given below we are interested in getting temperature only.

```
{
  "sys":
    {
      "country": "GB",
      "sunrise": 1381107633,
      "sunset": 1381149604
    },
  "weather": [
    {
      "id": 711,
      "main": "Smoke",
      "description": "smoke",
      "icon": "50n"
    }
  ],
  "main":
    {
      "temp": 304.15,
      "pressure": 1009,
    }
}
```

### JSON - Elements

An JSON file consists of many components. Here is the table defining the components of an JSON file and their description:

| Sr.No | Component & description   |
|-------|---|
| 1     | <b>Array([])</b><br>In a JSON file, square bracket ([]) represents a JSON array |
| 2     | <b>Objects({})</b>  |



# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

|   |  |
|---|--|
|   | In a JSON file, curly bracket ({} represents a JSON object   |
| 3 | <b>Key</b><br>A JSON object contains a key that is just a string. Pairs of key/value make up a JSON object |
| 4 | <b>Value</b><br>Each key has a value that could be string , integer or double e.t.c                        |

## JSON - Parsing

For parsing a JSON object, we will create an object of class JSONObject and specify a string containing JSON data to it. Its syntax is:

```
String in;  
JSONObject reader = new JSONObject(in);
```

The last step is to parse the JSON. An JSON file consist of different object with different key/value pair e.t.c. So JSONObject has a seperate function for parsing each of the component of JSON file. Its syntax is given below:

```
JSONObject sys = reader.getJSONObject("sys");  
country = sys.getString("country");
```

```
JSONObject main = reader.getJSONObject("main");  
temperature = main.getString("temp");
```

The method **getJSONObject** returns the JSON object. The method **getString** returns the string value of the specified key.

## The difference between [ and { - (Square brackets and Curly brackets)

In general all the JSON nodes will start with a square bracket or with a curly bracket. The difference between [ and { is, the square bracket ([]) represents starting of an **JSONArray** node whereas curly bracket ({} represents **JSONObject**. So while accessing these nodes we need to call appropriate method to access the data.

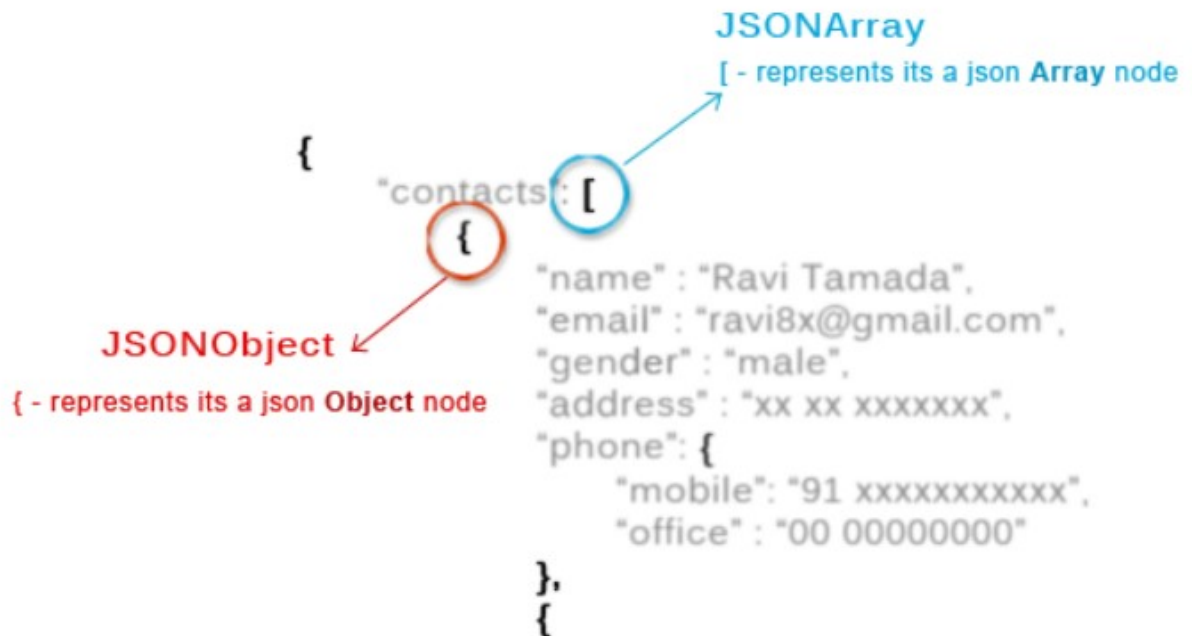
If your JSON node starts with [, then we should use getJSONArray() method. Same as if the node starts with {, then we should use JSONObject() method.

# SHREE H. N. SHUKLA GROUP OF COLLEGES

(Affiliated to Saurashtra University & Gujarat Technological University)

JSON Structure

AndroidHiv



```
<uses-permission
android:name="android.permission.INTERNET" />
```

**JSON** (Javascript Object Notation) is a programming language . It is minimal, textual, and a subset of JavaScript. It is an alternative to XML.

Android provides support to parse the JSON object and array.

## Advantage of JSON over XML

- 1) JSON is faster and easier than xml for AJAX applications.
- 2) Unlike XML, it is shorter and quicker to read and write.
- 3) It uses array.

## json object

A JSON object contains key/value pairs like map. The keys are strings and the values are the JSON types. Keys and values are separated by comma. The { (curly brace) represents the json object.

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

```
{
  "employee": {
    "name": "sachin",
    "salary": 56000,
    "married": true
  }
}
```

## **json array**

The [ (square bracket) represents the json array.

```
["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
{ "Employee" :
  [
    {"id": "101", "name": "Sonoo Jaiswal", "salary": "50000"},
    {"id": "102", "name": "Vimal Jaiswal", "salary": "60000"}
  ]
}
```

## **activity main.xml**

```
<RelativeLayout 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" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="75dp"
        android:layout_marginTop="46dp"
        android:text="TextView" />
```

```
</RelativeLayout>
```

## **Activity.java**

```
package com.javatpoint.jsonparsing;
```

```
import org.json.JSONException;
import org.json.JSONObject;
import android.app.Activity;
import android.os.Bundle;
```

# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

```
import android.widget.TextView;

public class MainActivity extends Activity {
    public static final String JSON_STRING="{\"employee\":{\"name\":\"Sachin\",\"salary\":56000}}";

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

        TextView textView1=(TextView)findViewById(R.id.textView1);

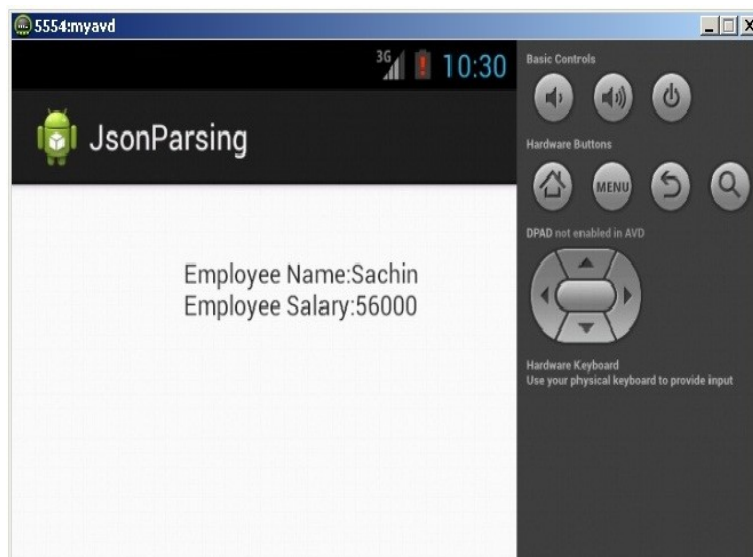
        try{
            JSONObject emp=(new JSONObject(JSON_STRING)).getJSONObject("employee");

            String empname=emp.getString("name");
            int empsalary=emp.getInt("salary");

            String str="Employee Name:"+empname+"\n"+"Employee Salary:"+empsalary;
            textView1.setText(str);

        }catch (Exception e) {e.printStackTrace();}

    }
}
```



# SHREE H. N. SHUKLA GROUP OF COLLEGES

*(Affiliated to Saurashtra University & Gujarat Technological University)*

## Android XML Parsing using SAX Parser

Android provides the facility to parse the xml file using SAX, DOM etc. parsers. The SAX parser cannot be used to create the XML file, It can be used to parse the xml file only.

### Advantage of SAX Parser over DOM

It consumes less memory than DOM.

---

## Android XML Parsing using DOM Parser

**We can parse the xml document by dom parser also. It can be used to create and parse the xml file.**

### Advantage of DOM Parser over SAX

**It can be used to create and parse the xml file both but SAX parser can only be used to parse the xml file.**

### Disadvantage of DOM Parser over SAX

**It consumes more memory than SAX.**