**Android Activity Lifecycle**

**Android Activity Lifecycle** is controlled by 7 methods of android.app.Activity class. The android Activity is the subclass of ContextThemeWrapper class.

An activity is the single screen in android. It is like window or frame of Java.

By the help of activity, you can place all your UI components or widgets in a single screen.

The 7 lifecycle method of Activity describes how activity will behave at different states.

**Android Activity Lifecycle methods**

Let's see the 7 lifecycle methods of android activity.

|  |  |
| --- | --- |
| **Method** | **Description** |
| **onCreate** | called when activity is first created. |
| **onStart** | called when activity is becoming visible to the user. |
| **onResume** | called when activity will start interacting with the user. |
| **onPause** | called when activity is not visible to the user. |
| **onStop** | called when activity is no longer visible to the user. |
| **onRestart** | called after your activity is stopped, prior to start. |
|  |  |
| **onDestroy** | called before the activity is destroyed. |



package example.javatpoint.com.activitylifecycle;

import android.app.Activity;

import android.os.Bundle;

import android.util.Log;

public class MainActivity extends Activity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        Log.d("lifecycle","onCreate invoked");

    }

    @Override

    protected void onStart() {

        super.onStart();

        Log.d("lifecycle","onStart invoked");

    }

    @Override

    protected void onResume() {

        super.onResume();

        Log.d("lifecycle","onResume invoked");

    }

   @Override

    protected void onPause() {

     super.onPause();

        Log.d("lifecycle","onPause invoked");

    }

    @Override

    protected void onStop() {

        super.onStop();

        Log.d("lifecycle","onStop invoked");

    }

    @Override

    protected void onRestart() {

        super.onRestart();

        Log.d("lifecycle","onRestart invoked");

    }

   @Override

    protected void onDestroy() {

        super.onDestroy();

        Log.d("lifecycle","onDestroy invoked");

    }

}

**Android Intent Tutorial**

Android Intent is the message that is passed between components such as activities, content providers, broadcast receivers, services etc.

It is generally used with startActivity() method to invoke activity, broadcast receivers etc.

The dictionary meaning of intent is intention or purpose. So, it can be described as the intention to do action.

The LabeledIntent is the subclass of android.content.Intent class.

Android intents are mainly used to:

* Start the service
* Launch an activity
* Display a web page
* Display a list of contacts
* Broadcast a message
* Dial a phone call etc.

**Types of Android Intents**

There are two types of intents in android: implicit and explicit.

1) **Implicit Intent**

Implicit Intent doesn't specifiy the component. In such case, intent provides information of available components provided by the system that is to be invoked.

For example, you may write the following code to view the webpage.

Intent intent=new Intent(Intent.ACTION\_VIEW);

intent.setData(Uri.parse("http://www.javatpoint.com"));

startActivity(intent);

2) **Explicit Intent**

Explicit Intent specifies the component. In such case, intent provides the external class to be invoked.

Intent i = new Intent(getApplicationContext(), ActivityTwo.class);

startActivity(i);

To get the full code of explicit intent, visit the next page.

**Android Implicit Intent Example**

Let's see the simple example of implicit intent that displays a web page.

**activity\_main.xml**

File: **activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context="example.javatpoint.com.implicitintent.MainActivity">

    <EditText

        android:id="@+id/editText"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_marginEnd="8dp"

        android:layout\_marginStart="8dp"

        android:layout\_marginTop="60dp"

        android:ems="10"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintHorizontal\_bias="0.575"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintTop\_toTopOf="parent" />

    <Button

        android:id="@+id/button"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_marginRight="8dp"

        android:layout\_marginLeft="156dp"

        android:layout\_marginTop="172dp"

        android:text="Visit"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintHorizontal\_bias="0.0"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintTop\_toBottomOf="@+id/editText" />

</android.support.constraint.ConstraintLayout>

**Activity class**

File: **MainActivity.java**

package example.javatpoint.com.implicitintent;

import android.content.Intent;

import android.net.Uri;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

public class MainActivity extends AppCompatActivity {

    Button button;

    EditText editText;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        button = findViewById(R.id.button);

        editText =  findViewById(R.id.editText);

        button.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View view) {

                String url=editText.getText().toString();

                Intent intent=new Intent(Intent.ACTION\_VIEW, Uri.parse(url));

                startActivity(intent);

            }

        });

    }

}

**Android Service Tutorial**

**Android service** is a component that is *used to perform operations on the background* such as playing music, handle network transactions, interacting content providers etc. It doesn't has any UI (user interface).

The service runs in the background indefinitely even if application is destroyed.

Moreover, service can be bounded by a component to perform interactivity and inter process communication (IPC).

The android.app.Service is subclass of ContextWrapper class.

**Life Cycle of Android Service**

There can be two forms of a service.The lifecycle of service can follow two different paths: started or bound.

1. Started
2. Bound

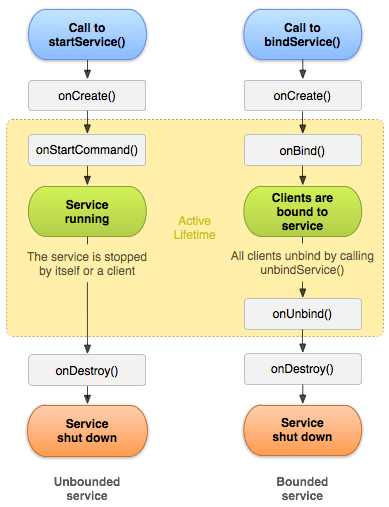
**1) Started Service**

A service is started when component (like activity) calls **startService()** method, now it runs in the background indefinitely. It is stopped by **stopService()** method. The service can stop itself by calling the **stopSelf()** method.

**2) Bound Service**

A service is bound when another component (e.g. client) calls **bindService()** method. The client can unbind the service by calling the **unbindService()** method.

The service cannot be stopped until all clients unbind the service.

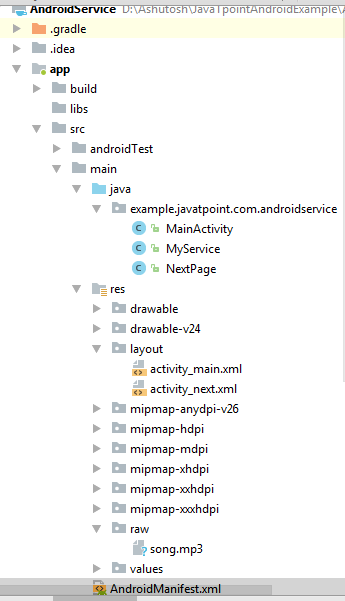


**Understanding Started and Bound Service by background music example**

Suppose, I want to play music in the background, so call startService() method. But I want to get information of the current song being played, I will bind the service that provides information about the current song.

**Android Service Example**

Let's see the example of service in android that plays an audio in the background. Audio will not be stopped even if you switch to another activity. To stop the audio, you need to stop the service.



**activity\_main.xml**

Drag the 3 buttons from the pallete, now the activity\_main.xml will look like this:

File: activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context="example.javatpoint.com.androidservice.MainActivity">

    <Button

        android:id="@+id/buttonStart"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_alignParentTop="true"

        android:layout\_centerHorizontal="true"

        android:layout\_marginTop="74dp"

        android:text="Start Service" />

    <Button

        android:id="@+id/buttonStop"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_centerHorizontal="true"

        android:layout\_centerVertical="true"

        android:text="Stop Service" />

    <Button

        android:id="@+id/buttonNext"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_alignParentBottom="true"

        android:layout\_centerHorizontal="true"

        android:layout\_marginBottom="63dp"

        android:text="Next Page" />

</RelativeLayout>

activity\_next.xml

It is the layout file of next activity.

File: activity\_next.xml

It contains only one textview displaying the message Next Page

<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context="example.javatpoint.com.androidservice.NextPage">

    <TextView

        android:id="@+id/textView"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:layout\_marginEnd="8dp"

        android:layout\_marginStart="8dp"

        android:layout\_marginTop="200dp"

        android:text="Next Page"

        app:layout\_constraintEnd\_toEndOf="parent"

        app:layout\_constraintStart\_toStartOf="parent"

        app:layout\_constraintTop\_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>

Service class

Now create the service implemenation class by inheriting the Service class and overridding its callback methods.

File: MyService.java

package example.javatpoint.com.androidservice;

import android.app.Service;

import android.content.Intent;

import android.media.MediaPlayer;

import android.os.IBinder;

import android.support.annotation.Nullable;

import android.widget.Toast;

public class MyService extends Service {

    MediaPlayer myPlayer;

    @Nullable

    @Override

    public IBinder onBind(Intent intent) {

        return null;

    }

    @Override

    public void onCreate() {

        Toast.makeText(this, "Service Created", Toast.LENGTH\_LONG).show();

        myPlayer = MediaPlayer.create(this, R.raw.sun);

        myPlayer.setLooping(false); // Set looping

    }

    @Override

    public void onStart(Intent intent, int startid) {

        Toast.makeText(this, "Service Started", Toast.LENGTH\_LONG).show();

        myPlayer.start();

    }

    @Override

    public void onDestroy() {

        Toast.makeText(this, "Service Stopped", Toast.LENGTH\_LONG).show();

        myPlayer.stop();

    }

}

Activity class

Now create the MainActivity class to perform event handling. Here, we are writing the code to start and stop service. Additionally, calling the second activity on buttonNext.

File: MainActivity.java

package example.javatpoint.com.androidservice;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

public class MainActivity extends AppCompatActivity implements View.OnClickListener{

    Button buttonStart, buttonStop,buttonNext;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        buttonStart = findViewById(R.id.buttonStart);

        buttonStop = findViewById(R.id.buttonStop);

        buttonNext =  findViewById(R.id.buttonNext);

        buttonStart.setOnClickListener(this);

        buttonStop.setOnClickListener(this);

        buttonNext.setOnClickListener(this);

    }

    public void onClick(View src) {

        switch (src.getId()) {

            case R.id.buttonStart:

                startService(new Intent(this, MyService.class));

                break;

            case R.id.buttonStop:

                stopService(new Intent(this, MyService.class));

                break;

            case R.id.buttonNext:

                Intent intent=new Intent(this,NextPage.class);

                startActivity(intent);

                break;

        }

    }

}

NextPage class

Now, create another activity.

File: NextPage.java

package example.javatpoint.com.androidservice;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

public class NextPage extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_next);

    }

}

Declare the Service in the AndroidManifest.xml file

Finally, declare the service in the manifest file.

File: AndroidManifest.xml

Let's see the complete AndroidManifest.xml file

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

    package="example.javatpoint.com.androidservice">

    <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>

        <activity android:name=".NextPage"></activity>

        <service

            android:name=".MyService"

            android:enabled="true" />

    </application>

</manifest>