[**안드로이드 서비스(Service)**](http://treejsh.tistory.com/entry/%EC%95%88%EB%93%9C%EB%A1%9C%EC%9D%B4%EB%93%9C-service)

[서비스](http://treejsh.tistory.com/category/%EC%84%9C%EB%B9%84%EC%8A%A4) 2012/04/12 10:45

안드로이드 응용프로그램은 Activity, Content Provider, Broadcast Receiver, Service 이렇게 4개로 구성되어있습니다.

이번 포스트는 이중 마지막 **Service** 에 대해서 알아보겠습니다.

Service 는 사용자가 보여지는 부분이 아닌 백그라운드에서 동작합니다.

뮤직앱에서 다른 화면으로 넘어가도 소리가 들려야 되기때문에 실제로 서비스에서 재생이 됩니다. 다운로드 서비스도 마찬가지 입니다.

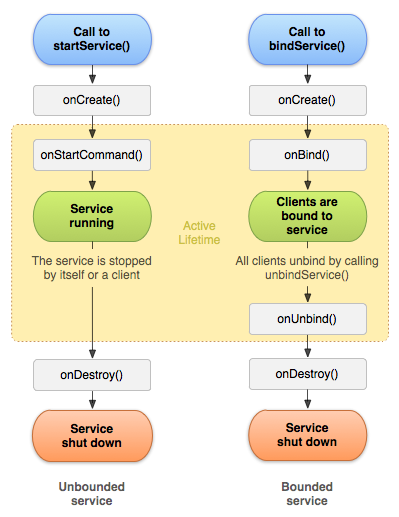
또한 Broadcast Receiver은 7초 제한이 있기 때문에 보통 시간이 소요될만한 작업은 Service를 이용하여 구현합니다.

안드로이드 서비스는 2가지 형태로 되어있습니다.

주기적으로 백그라운드에서 실행이 되는 데몬형태와 메소드를 노출시켜 화면에서 실행할수 있도록 하는 원격 호출 인터페이스가 있습니다.

원격 호출 인터페이스는 aidl 을 사용하며 다음 포스트에서 알아보겠습니다.

다음은 서비스의 lifecycle과 callback method 입니다.



첫번째는 데몬형식의 서비스이고 두번째는 원격호출인터페이스 형식입니다.

위의 콜백 메소드를 재정의 하므로서 서비스를 실행합니다.

onBind()가 추상메소드 이므로 문법적으로 반드시 재정의 해주어야 하고 모든 서비스는 onCreate()에서 시작하고 onDestroy()에서 종료됩니다.

client 호출방식에 따라서  startService()가 호출되면 onStartCommand()가 bindService()가 호출되면 onBind()가 실행되면서 서비스가 실행이 됩니다.

또한 startService()로 시작한 서비스를 다시 bindService()로 호출하여 onBind()를 실행할수 있습니다.

그럼 이제 코드를 작성해 보겠습니다.

다음은 Service class 입니다.

편의상 sdcard의 root에 test.mp3 파일을 넣고 그 mp3파일을 실행하는 예제입니다.

[view source](http://treejsh.tistory.com/12#viewSource)

[print](http://treejsh.tistory.com/12#printSource)[?](http://treejsh.tistory.com/12#about)

01.**public** **class** MyService **extends** Service{

02.MediaPlayer mMediaPlayer;

03.String mMp3Path = "";

04.

05.@Override

06.**public** **void** onCreate() {

07.**super**.onCreate();

08.//미디어플레이어를 초기화

09.mMediaPlayer = **new** MediaPlayer();

10.//곡 재생이 완료하면 서비스를 종료시킨다

11.mMediaPlayer.setOnCompletionListener(**new** OnCompletionListener() {

12.**public** **void** onCompletion(MediaPlayer arg0) {

13.stopSelf(); //서비스 종료

14.}

15.});

16.}

17.

18.@Override

19.**public** **int** onStartCommand(Intent intent, **int** flags, **int** startId) {

20.Toast.makeText(**this**, "service start..", Toast.LENGTH\_SHORT).show();

21.//sdcard에 있는 test.mp3 을 찾아검사

22.String ext = Environment.getExternalStorageState();

23.**if** (ext.equals(Environment.MEDIA\_MOUNTED)) {

24.mMp3Path = Environment.getExternalStorageDirectory().getAbsolutePath()+"/test.mp3";

25.File mp3file = **new** File(mMp3Path);

26.**if**(mp3file.exists()){

27.//mp3파일이 있으면 쓰레드실행

28.**new** Thread(mRun).start();

29.}

30.}

31.**return** START\_STICKY;

32.}

33.

34.Runnable mRun = **new** Runnable() {

35.**public** **void** run() {

36.**try**{

37.//미디어플레이어 재생

38.mMediaPlayer.setDataSource(mMp3Path);

39.mMediaPlayer.prepare();

40.mMediaPlayer.start();

41.}**catch**(Exception e){

42.e.printStackTrace();

43.}

44.}

45.};

46.

47.@Override

48.**public** **void** onDestroy() {

49.Toast.makeText(**this**, "service destroy..", Toast.LENGTH\_SHORT).show();

50.**if**(mMediaPlayer!= **null** && mMediaPlayer.isPlaying()){

51.mMediaPlayer.stop();

52.mMediaPlayer.release();

53.mMediaPlayer=**null**;

54.}

55.**super**.onDestroy();

56.}

57.

58.@Override

59.**public** IBinder onBind(Intent arg0) {

60.**return** **null**;

61.}

62.}

onCreate()에 MediaPlayer 객체를 초기화 했으며 onStartCommand() 에서 실행합니다.

onStartCommand()는 시스템이 좀더 잘 관리할수 있도록 리턴값을 반환합니다.

* START\_STICTY

메모리부족이나 기타 상황에서 시스템이 강제로 service를 종료된후 service가 재시작될때 null Intent가 담긴 onStartCommand() 콜백함수가 실행된다. 이 경우 null Intent로 호출때의 경우를 처리해줘야 합니다.

* START\_NOT\_STICTY

이 경우는 프로세스가 강제로 종료되었을 경우 재시작하지 않고 종료된 상태로 남게 됩니다. 예를 들면 매 15분마다 네트워크 체크를 하는 service가 강제로 종료되었을경우 15분후에 자동적으로 다시 service가 실행되므로 재시작하지 않아도 되는 경우입니다.

* START\_REDELIVER\_INTENT

이 경우에는 프로세스가 강제로 종료되었을 경우 Intent가 다시 전달되어 재시작합니다. 단, 여러차레 시도후 작업이 종료되지 않으면 service는 재시작 되지 않습니다. 반드시 실행되어야 하는 service에 해당이 됩니다.

service는 메인 thread에서 실행이 되고 백그라운드에서 사용자가 알수 없도록 실행되어야 하기 때문에 별도의 thread를 통해서 MediaPlayer객체를 실행합니다. 그렇지 않으면 MediaPlayer가 실행하는 동안에 사용자가 화면상에서의 액션을 취하려면 delay이가 발생할 것입니다. thread를 사용하지 않아도 재대로 작동할 것입니다.

MediaPlayer가 재생을 마치면 콜백메서드에서 자동으로 서비스가 종료되게끔 하였습니다.

서비스를 사용하려면 AndroidManifest.xml 에 서비스를 등록해야합니다.

[view source](http://treejsh.tistory.com/12#viewSource)

[print](http://treejsh.tistory.com/12#printSource)[?](http://treejsh.tistory.com/12#about)

1.<**service** android:name=".MyService" android:enabled="true">

2.<**intent-filter**>

3.<**action** android:name="com.test.SERVICETEST">

4.</**action**></**intent-filter**>

5.</**service**>

암시적 Intent를 사용할수도 있기 때문에 intent filter에 action명을 등록합니다.

layout xml을 작성해보록 하겠습니다.

[view source](http://treejsh.tistory.com/12#viewSource)

[print](http://treejsh.tistory.com/12#printSource)[?](http://treejsh.tistory.com/12#about)

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

2.<**linearlayout** xmlns:android="<http://schemas.android.com/apk/res/android>"android:layout\_width="match\_parent" android:layout\_height="match\_parent"android:orientation="horizontal">

3.<**button** android:id="@+id/service\_start" android:layout\_width="wrap\_content"android:layout\_height="wrap\_content" android:text="서비스시작">

4.</**button**><**button** android:id="@+id/service\_end"android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"android:text="서비스종료">

5.

6.</**button**></**linearlayout**>

간단하게 서비스시작 버튼과 서비스종료버튼을 배치하여 서비스시작버튼을 클릭했을때 서비스를 시작하여 test.mp3파일을 실행하고 서비스종료버튼을 클릭했을때 서비스종료와 동시에 MediaPlayer를 종료하게끔 하였습니다.

그럼 이제 Activity 를 작성하겠습니다.

[view source](http://treejsh.tistory.com/12#viewSource)

[print](http://treejsh.tistory.com/12#printSource)[?](http://treejsh.tistory.com/12#about)

01.**public** **class** MyServiceActivity **extends** Activity **implements** OnClickListener{

02.@Override

03.**protected** **void** onCreate(Bundle savedInstanceState) {

04.**super**.onCreate(savedInstanceState);

05.setContentView(R.layout.service\_test);

06.

07.Button service\_start = (Button)findViewById(R.id.service\_start);

08.Button service\_end = (Button)findViewById(R.id.service\_end);

09.

10.service\_start.setOnClickListener(**this**);

11.service\_end.setOnClickListener(**this**);

12.}

13.

14.**public** **void** onClick(View v) {

15.**if**(v.getId() == R.id.service\_start){

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

17.}**else** **if**(v.getId() == R.id.service\_end){

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

19.}

20.}

21.}

서비스시작 버튼을 클릭하면 startService메서드를 호출하여 서비스가 실행이 됩니다.

호출방식은 명시적인 방법과 암시적인 방법이 있는데 위 코드에서는 직접 클래스 이름을 넣어준 명시적인 방법으로 실행해보았습니다.

위 코드를 실행해보면 서비스시작 버튼을 클릭하면 서비스가 실행이되어 mp3파일이 재생이 됩니다.

서비스종료 버튼을 클릭하거나 mp3파일재생을 마치면 서비스가 종료가 됩니다.

이제까지 서비스에 대해서 알아보았습니다.

다음 포스트에서는 play버튼과 pause버튼을 두고 AIDL을 사용하여 원격 호출 인터페이스방식에 대해서 알아보겠습니다.