Android servis

Servis je komponenta koja se izvršava u pozadini kako bi se omogućila realizacija dugotrajnih operacija, kao i izvršavanje udaljenih procesa. Servis ne pruža korisnički interfejs. Kada se jednom pokrene, servis može da nastavi svoje izvršavanje čak i kada korisnik otvori drugu aplikaciju. Na primer, servis može da pušta muziku u pozadini, obavlja mrežnu komunikaciju, ulazno/izlazne operacije sa datotekama i slično. Dodatno, komponte aplikacije mogu da se povežu na servis i da komuniciraju sa njim. Takođe, pomoću servisa može da se obavlja interprocesna komunikacija (IPC).

Svaka klasa servisa mora imati odgovarajuću deklaraciju u AndroidManifest.xml datoteci:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.javatechig.serviceexample" >

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".HelloActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <!--Service declared in manifest -->
        <service android:name=".HelloService"
            android:exported="false"/>
    </application>

</manifest>
```

Slika 1 - Primer AndroidManifest.xml datoteke sa deklaracijom servisa HelloService

Servis ili nit?

Kako odabrati između kreiranja Android servisa ili niti (Thread/Runnable)?

Servis je komponenta koja izvršava operaciju u pozadini (čak i ako aplikacija nije pokrenuta). Ukoliko je potrebno da aplikacija izvršava određenu radnju samo dok je aplikacija pokrenuta (na primer

puštanje muzike dok je aplikacija u prvom planu) dovoljno je kreirati novu nit.

Kada se servis pokrene on će svoju radnju izvršavati u glavnoj niti, osim ukoliko se ne implementira drugačije. Ukoliko je potrebno da servis obavlja dugotrajne, opterećujuće operacije poželjno je kreirati novu nit unutar servisa koja obavlja blokirajuće operacije.

Tipovi servisa

Postoje tri tipa servisa:

1. Foreground servis
2. Background servis
3. Bound servis

Foreground tip servisa

Servis u prvom planu vrši neku operaciju koja je primetna korisniku. Na primer, audio aplikacija bi koristila servis u prvom planu za reprodukciju audio zapisa. Servis nastavlja sa radom i kada korisnik ne interaguje sa aplikacijom.

Ovaj tip servisa mora prikazati obaveštenje (Notification) kako bi korisnik bio svestan da je servis pokrenut. Obaveštenje ne može biti uklonjeno sve dok se servis ne zaustavi.

Background tip servisa

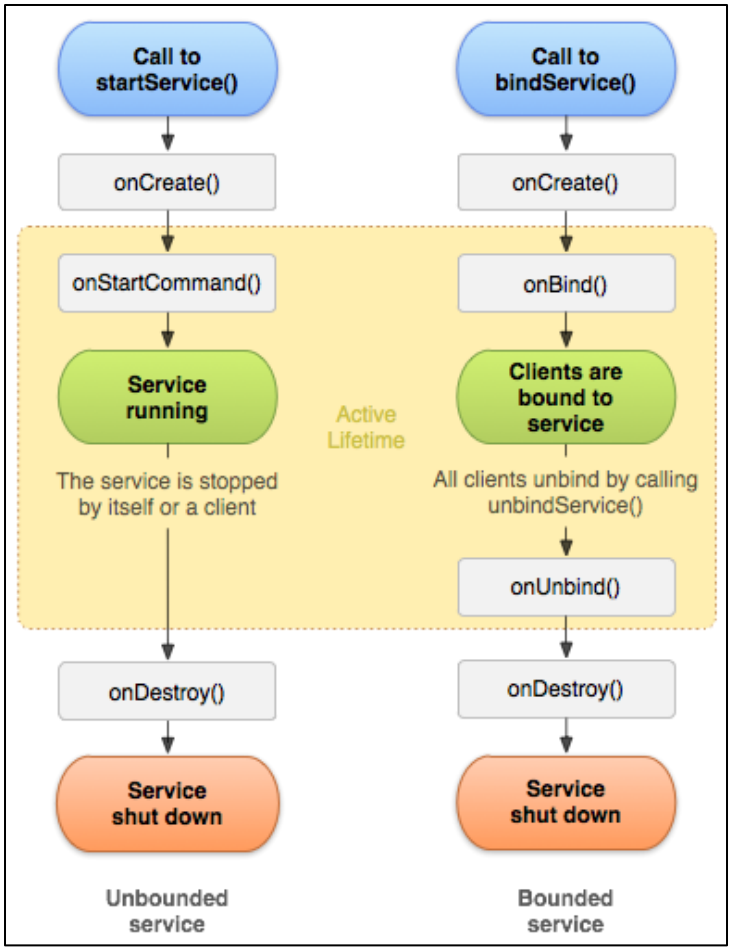
Pozadinski servis obavlja operaciju koju korisnik ne primećuje direktno. Na primer, čuvanje rezervnih kopija podataka (backup) bi mogao da obavlja pozadinski servis.

Bound tip servisa

Servis je vezan kada se komponenta aplikacije vezuje za njega pozivanjem metode `bindService`. Vezani servis nudi klijent-server interfejs koji omogućava komponentama da komuniciraju s servisom, šalju zahteve, primaju odgovore (čak i između različitih procesa - IPC). Vezani servis radi samo dok je druga komponenta aplikacije vezana za njega. Više različitih komponenti mogu da se vežu za servis. Servis se uništava tek kada se sve komponente odvežu.

Životni ciklus servisa

Metode životnog ciklusa servisa su prikazane na slici - Slika 2. Životni ciklus vezanog servisa se razlikuje od životnog ciklusa servisa koji nije vezan (za ovakav servis se kaže da je on *pokrenut*).



Slika 2 - Životni ciklus servisa

Pokrenut servis se aktivira kada neka komponenta aplikacije pozove metodu `startService`. Servis se mora zaustaviti. U suprotnom, servis će biti pokrenut zauvek (neće se sam zaustaviti, osim ukoliko ga Android sistem ne uništi). Servis je moguće zaustaviti na dva načina:

- servis poziva metodu `stopSelf`
- druga komponenta aplikacije poziva metodu `stopService`.

Vezan servis se aktivira kada neka komponenta (klijent) pozove metodu `bindService`. Klijent komunicira sa servisom kroz `IBinder` interfejs. Klijent zatvara vezu ka servisu pozivom metode `unbindService`. Svaki povezani klijent mora da zatvori vezu ka servisu. Tek kada su sve veze zatvorene, servis se uništava.

Komponte aplikacije mogu da se povežu i na servis koji je pokrenut metodom `startService` pozivom metode `bindService`. Tada pokrenut servis postaje vezan servis.

Kreiranje servisa

Kreiranje instance servisa: File -> New -> Service -> Service. Kada se klasa servisa kreira na ovaj način AndroidStudio će automatski dodati instancu servisa u AndroidManifest.xml datoteku.

Svaki servis koji se implementira mora da nasledi klasu `Service`. Ova klasa je apstraktna i pruža apstraktnu metodu `onBind` koja mora biti implementirana (jer svaki servis potencijalno može postati vezan servis). Ova callback metoda se poziva kada se komponenta poveže na servis pozivom metode `bindService`. Povratna vrednost funkcije je instanca `IBinder` interfejsa koji treba da sadrži interfejs za komunikaciju komponenti sa servisom.

Dodatne metode `Service` klase koje je poželjno implementirati su:

- `onStartCommand` - ova callback metoda se poziva kada se servis pokrene pozivom metode `startService`; nakon izvršavanja ova metode, servis je pokrenut sve dok se ne zaustavi pozivom metode `stopSelf` ili `stopService`.
- `onCreate` - ova callback metoda se poziva prvi put kada se servis pokrene, pre poziva metoda `onStartCommand` ili `onBind` (poziva se samo jednom, ukoliko je servis već pokrenut `onCreate` metoda se neće pozvati)
- `onDestroy` - ova callback metoda se poziva kada se servis uništava.

Pokretanje servisa

Servis se pokreće pozivanjem metode `startService`. Ovoj metodi se prosleđuje objekat klase `Intent` u kome se navodi koji servis se pokreće. Na sličan način, servis se može uništiti pozivom metode `stopService` i prosleđivanjem istog objekta klase `Intent`.

Poziv metode startService rezultuje pozivom callback metode `onStartCommand` (i `onCreate` ukoliko se servis prvi put kreira).

```
Intent intent = new Intent(this, MyService.class);
startService(intent);
stopService(intent);
```