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. Na primer, servis može da pušta muziku u pozadini dok se korisnik nalazi u nekoj drugoj aplikacija. Aktivnost može da pokrene servis.

Svaka klasa servisa mora imati odgovarajuću <service> deklaraciju u AndroidManifest.xml paketu.

```
<?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>
```

Postoje tri vrste servisa:

Foreground

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 u budućnosti mora prikazati obaveštenje, takođe nastavlja rad i kada korisnik ne interaktuje sa aplikacijom.

Background

Pozadinski servis obavlja operaciju koju korisnik ne primećuje direktno. Na primer, ako je aplikacija koristila servis za kompaktno skladištenje, to je obično servis pozadine.

Bound

Servis je vezan kada se komponenta aplikacije vezuje za njega pozivanjem bindService (). Vezani servis nudi interfejs između klijenta i servera koji omogućava komponentama da komuniciraju s servisom, šalju zahteve, primaju rezultate i čak i to rade preko procesa sa interprocess komunikacijom (IPC). Vezani servis radi samo dok je druga komponenta aplikacije vezana za to. Višestruke komponente mogu da se vezuju za servis odjednom, ali kada se svi razdvoje, servis je uništen.

Zivotni ciklus

Postoje dva razloga da sistem može pokrenuti servis. Ako neko pozove Context.startService () onda će sistem preuzeti servis (kreirajući ga i pozvati svoj metod onCreate () ako je potrebno), a zatim pozovite metodu onStartCommand (Intent, int, int) sa argumentima koje je dao klijent. Servis će u ovom trenutku nastaviti da radi sve dok se ne pozove Context.stopService () ili stopSelf ().
Klijenti takođe mogu da koriste Context.bindService () da bi dobili trajno povezivanje sa servisom. Ovo takođe stvara servis ako se već ne pokrene (pozivajući na onCreate () dok to radi), ali ne poziva onStartCommand (). Klijent će primiti IBinder objekat koji server vraća iz metode onBind (Intent), omogućavajući klijentu da pozove pozive na servis. Servis će ostati aktivan sve dok se veza uspostavi (bez obzira da li klijent zadržava referencu na IBinder servisu.

Najznačajnije metode su:

onStartCommand ()

Sistem poziva ovu metodu pozivom startService () kada druga komponenta (kao što je aktivnost) zahteva da se usluga pokrene. Kada se ovaj metod izvršava, usluga se pokreće i može se pokrenuti u pozadini na neodređeno vreme.

onBind ()

Sistem poziva ovu metodu pozivom bindService () kada se druga komponenta želi povezati sa uslugom (kao što je izvršavanje RPC-a). Prilikom implementacije ovog metoda, morate osigurati interfejs koji klijenti koriste za komunikaciju sa uslugom vraćanjem IBinder-a. Morate uvek da implementirate ovaj metod.

onCreate ()

Sistem poziva ovaj metod da izvrši jednokratne procedure za podešavanje kada je usluga prvobitno kreirana (pre nego što pozove ili onStartCommand () ili onBind ()). Ako se usluga već pokreće, ovaj metod se ne poziva.

onDestroy ()

Sistem poziva ovaj metod kada se usluga više ne koristi i uništava se. Vaša usluga treba da implementira ovo da očisti bilo koji resurs, kao što su niti, registrovani slušaoci ili prijemnici. Ovo je poslednji poziv koji usluga dobija.

```
public class HelloService extends Service {

    private static final String TAG = "HelloService";

    private boolean isRunning  = false;

    @Override
    public void onCreate() {
        Log.i(TAG, "Service onCreate");

        isRunning = true;
    }
```