



SERVICES


Professor Emerson Alencar
emerson@imd.ufrn.br

SERVICES

- ▶ Services são componentes do Android que permitem executar processamento em segundo plano
 - Não estão ligados a uma interface gráfica
- ▶ Os services em execução são considerados de alta prioridade pelo Android
 - Só perdem para a activity visível
 - O Android destrói outros componentes antes de destruir os services

A CLASSE SERVICE

- ▶ Para criar um **service**, é preciso estender a **classe Service**



```
MyService.kt
2
3 import android.app.Service
4 import android.content.Intent
5 import android.os.IBinder
6
7 class MyService: Service() {
8
9     override fun onCreate() {
10         super.onCreate()
11     }
12
13     override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {
14         return super.onStartCommand(intent, flags, startId)
15     }
16
17     override fun onDestroy() {
18         super.onDestroy()
19     }
20
21
22     override fun onBind(p0: Intent?): IBinder? {
23         return null
24     }
25
26
27 }
```

CONFIGURANDO OS SERVICES

- Os services são configurados no arquivo AndroidManifest.xml

```
12
13
14     <service
15         android:name=".MyService"
16         android:enabled="true"
17         android:exported="true">
18
19         <intent-filter>
20             <action android:name="START_SERVICE"/>
21         </intent-filter>
22     </service>
23
```

EXECUTANDO UM SERVICE

- ▷ Um service pode ser iniciado por uma activity, broadcast receiver ou outro service
- ▷ Uma das formas de iniciá-lo é através do método **startService()**

```
16  
17 |startService(Intent( packageContext: this,HelloService::class.java))  
18
```

```
15  
16 i = Intent( action: "START_SERVICE")  
17 startService(i)  
18
```

EXECUTANDO UM SERVICE

- ▷ Depois que o service é iniciado com `startService()`, ele fica desvinculado de quem o criou
 - O service executa de forma independente
- ▷ Services não são processos ou threads
 - Eles executam na UI thread
 - É recomendado que o código do service seja executado em uma thread separada, que deve ser criada pelo programador

PARANDO UM SERVICE

- ▶ Para parar a execução de um service iniciado com `startService()`, existem duas formas
 - Invocar o método `stopService()`, fornecendo a intent que corresponde ao service

```
25  
26     if (i != null){  
27         stopService(i)  
28     }
```

- ▶ O próprio service invocar o método `stopSelf()`

ESCREVENDO UM SERVICE

- ▷ Um service deve estender da classe Service
 - O método `IBinder onBind(Intent)` deve ser implementado
- ▷ Métodos de callback normalmente utilizados
 - **`onCreate()`**
 - Invocado quando o service é criado
 - **`onStartCommand(Intent, int, int)`**
 - Invocado toda vez que `startService()` é chamado
 - O Android associa um `startId` diferente cada vez que `startService()` é invocado
 - **`onDestroy()`**
 - Invocado quando o service é destruído

CONECTANDO A UM SERVICE

- ▷ O Android proporciona uma forma para que seu código possa se conectar a um serviço em execução
- ▷ – Obter uma referência do serviço para invocar métodos
- ▷ Isto é feito através dos métodos **bindService()** e **unbindService()**

BINDER

- ▶ Um objeto chamado binder é responsável por fazer a "ponte" entre o seu código e o service

```
24  
25 override fun onBind(intent: Intent): IBinder {  
26     return binder  
27 }  
28  
29 inner class TimeServiceBinder: Binder(){  
30  
31     fun getService(): MyService {  
32         return this@MyService  
33     }  
34 }  
35
```

SERVICE CONNECTION

- ▶ Para que possa haver a conexão ao serviço, é preciso uma referência a um objeto do tipo **ServiceConnection**
- ▶ Os métodos dessa classe são invocados pelo Android
 - `onServiceConnected()`
 - Quando a conexão com o serviço foi feita com sucesso
 - `onServiceDisconnected()`
 - Quando a conexão com o serviço foi terminada

A CLASSE INTENTSERVICE

- ▷ Um service que herda de IntentService executa suas tarefas em uma thread a parte
 - IntentService herda de Service
 - É preciso implementar o método onHandleIntent()
 - Não é preciso se preocupar em criar uma thread manualmente
 - É criada uma thread a parte (worker) que executa as tarefas sequencialmente
 - Também não é preciso se preocupar com a parada do service
 - Ele termina automaticamente ao final da execução da tarefa

Obrigado!
Dúvidas?

Professor Emerson Alencar
emerson@imd.ufrn.br