BroadcastReceiver

Anderson Rodrigues

Intent

- O que são **Intents?**
- Tipos de **Intent**
- Utilizando **Intents**
- Como são instanciadas?

Intent

```
val intent = Intent( packageContext: this, FromNotificationActivity::class.java)
```

```
val intentActivity = Intent(App1BroadcastReceiver.ACTION)
```

```
val intent = Intent(context, CountDownService::class.java)
intent.putExtra(EXTRA_COUNTDOWN_TIME, countDownTime)
context.startService(intent)
```

PendingIntent

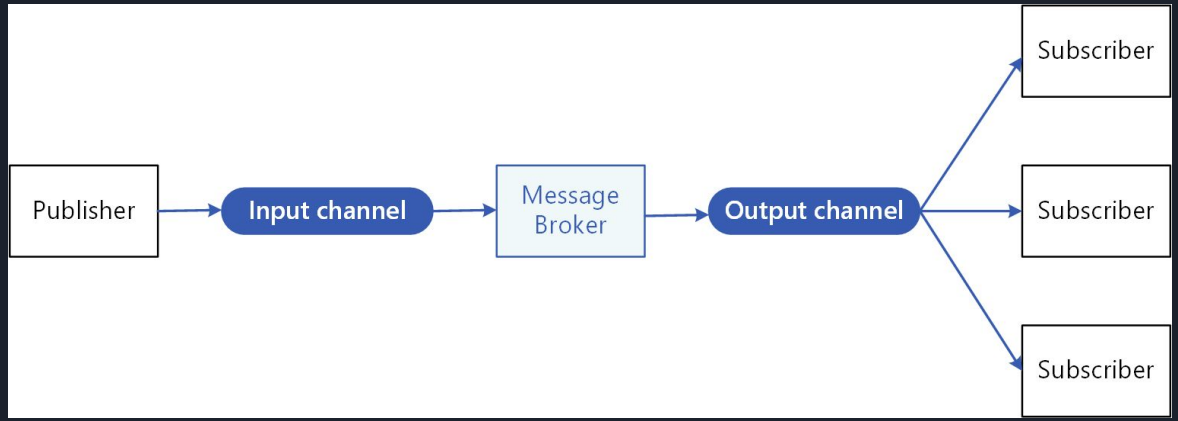
- O que é?
  - Possui uma ação associada
  - Permite a execução de uma ação
  - Activity, BroadcastReceiver, Service

Intent-filter

- Especifica os tipos de **Intents** que um componente pode responder
- Componentes:
  - Activity
  - Service
  - BroadcastReceiver
- Elementos:
  - Action
  - Category
  - Data

BroadcastReceiver

Publisher/Subscriber



https://docs.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber

Pub/Sub

- Youtube Subscriptions
- Mail list
- Twitch
- Push Notifications
- TV/Rádio
- Web feed - rss

System Services

- Serviços a nível de sistema
- Obtidos através de um *Context*

Currently available classes are: `WindowManager`, `LayoutInflater`, `ActivityManager`, `PowerManager`, `AlarmManager`, `NotificationManager`, `KeyguardManager`, `LocationManager`, `SearchManager`, `Vibrator`, `ConnectivityManager`, `WifiManager`, `AudioManager`, `MediaRouter`, `TelephonyManager`, `SubscriptionManager`, `InputMethodManager`, `UiModeManager`, `DownloadManager`, `BatteryManager`, `JobScheduler`, `NetworkStatsManager`, `DomainVerificationManager`.

```
getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
```

BroadcastReceiver

- Sistema de mensagens similar ao padrão pub/sub
- Permite:
  - Comunicação entre componentes
  - Comunicação com o sistema
  - Comunicação entre aplicações
- Executado em *background*
- Não são permitidas operações de longa duração

BroadcastReceiver

```
class App1BroadcastReceiver: BroadcastReceiver() {

    override fun onReceive(context: Context?, intent: Intent?) {
        intent?.let { receivedIntent ->
            Log.i( tag: "jamal",  msg: "onReceive - action: ${receivedIntent.action}")

            printExtras(receivedIntent.extras)
        }
    }
```

BroadcastReceiver

Podem ser divididos em dois tipos quanto ao seu registro:

- Estáticos ou declarados no Manifest
- Dinâmicos ou declarados via componente

BroadcastReceiver

Estático

- Geralmente utilizado para comunicação entre aplicações e com o sistema
- Não são atrelados ao ciclo de vida da aplicação
- Estado do Bluetooth, SMS, Push Notifications

BroadcastReceiver

```xml
<receiver android:name=".receiver.App1BroadcastReceiver">
    <intent-filter>
        <action android:name="com.ansorod.interapps1.App1BroadcastReceiver.ACTION" />
    </intent-filter>
</receiver>
```

```kotlin
val intentActivity = Intent(App1BroadcastReceiver.ACTION)
intentActivity.`package` = packageName
intentActivity.putExtra(App1BroadcastReceiver.EXTRA_MESSAGE, value: "This is an ordinary message sent locally to a Context registered receiver")

sendBroadcast(intentActivity)
```

BroadcastReceiver

Dinâmico

- Atrelado ao ciclo de vida de um componente
- Utilizado para comunicação com o sistema
- Comunicação entre aplicações

BroadcastReceiver

```
val intentActivity = Intent(App1BroadcastReceiver.ACTION)
intentActivity.putExtra(App1BroadcastReceiver.EXTRA_MESSAGE, value: "This is an ordinary message sent locally to a Context registered receiver")
```

```
private val receiver = App1BroadcastReceiver()

override fun onResume() {
    super.onResume()

    val filter = IntentFilter(App1BroadcastReceiver.ACTION)
    registerReceiver(receiver, filter)
}

override fun onPause() {
    super.onPause()

    unregisterReceiver(receiver)
}
```

BroadcastReceiver

Atividade

- Criar uma aplicação que, ao clique de um botão, enviará um *Broadcast* para dois *BroadcastReceiver* diferentes: um estático e um dinâmico

BroadcastReceiver

LocalBroadcastReceiver

- Utilizado para envio/recebimento de *broadcasts* dentro do contexto de uma aplicação
- Melhor performance

BroadcastReceiver

```
LocalBroadcastManager.getInstance( context: this).sendBroadcast(intent)
LocalBroadcastManager.getInstance( context: this).registerReceiver(receiver, filter)
LocalBroadcastManager.getInstance( context: this).unregisterReceiver(receiver)
```

BroadcastReceiver

This class is deprecated.

LocalBroadcastManager is an application-wide event bus and embraces layer violations in your app: any component may listen events from any other. You can replace usage of **`LocalBroadcastManager`** with other implementation of observable pattern, depending on your usecase suitable options may be **`LiveData`** or reactive streams.

BroadcastReceiver

- Como estabelecer comunicação entre duas aplicações distintas?
- Deveríamos utilizar um *broadcast* registrado estática ou dinamicamente?

BroadcastReceiver

Entre componentes

- Utilizando um *Service*

BroadcastReceiver

System *broadcasts*

- Interação do sistema com a aplicação
- Status de partes específicas
  - *bluetooth, airplane mode, SMS, ...*

BroadcastReceiver

Background execution limits

- https://developer.android.com/about/versions/oreo/background
- https://developer.android.com/guide/components/broadcast-exceptions

- Performance atrelada ao número de aplicações respondendo a alguns *broadcasts*
- Lista atualizada com *release* de novos *sdk*s

BroadcastReceiver

Código para *broadcasts* do sistema

- Bluetooth
- Airplane mode
- Locale changes

```
android_adv_school / broadcast / samples / system /
```

BroadcastReceiver

Atividade

- Criar uma aplicação que faz *scan* em busca de dispositivos *bluetooth* próximos
- *Scan* é iniciado ao clique de um botão
- Resultado do *scan* deve ser exibido em uma RecyclerView

BroadcastReceiver

- Utilizar *BluetoothAdapter.getDefaultAdapter().startDiscovery()*
- *BluetoothAdapter*.ACTION_DISCOVERY_STARTED
  - Indica o início do *scan*
- *BluetoothAdapter*.ACTION_DISCOVERY_FINISHED
  - Indica o fim do *scan*
- *android.bluetooth.BluetoothDevice*.ACTION_FOUND
  - Indica que um dispositivo foi encontrado
- *android.bluetooth.BluetoothDevice*.EXTRA_DEVICE
  - Traz um *android.bluetooth.BluetoothDevice*

android_adv_school / broadcast / activities / **bluetooth** /