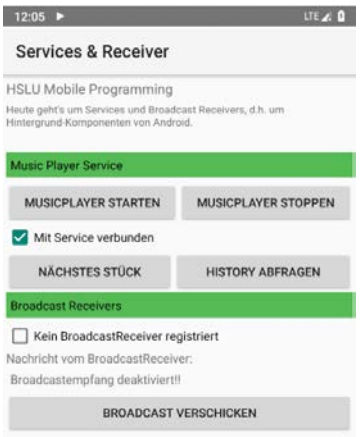
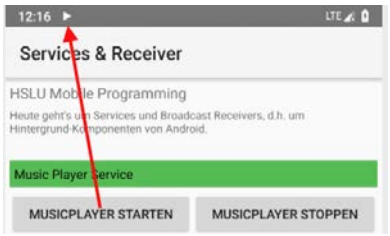
Übung 5: Services & Broadcast Receiver

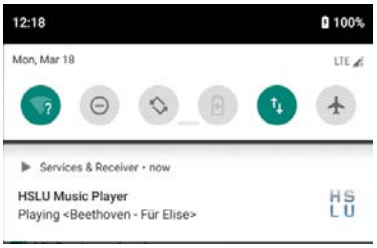


In dieser Übung geht es um Services und Broadcast Receiver. Sie werden einen eignen Foreground Service erstellen und mit diesem über ein Binding interagieren. Im zweiten Teil dieser Übung geht's um Broadcast Receiver und programmatische bzw. deklarative Intent-Filter. Sie werden dazu eigene Broadcasts versenden und empfangen. Im letzten Teil verwenden Sie einen WorkManager um Arbeiten im Hintergrund auszuführen und Resultate asynchron mit Broadcasts zurückzumelden.

1. **Neue App: Services & Receiver**
   Erstellen Sie ein neues Android-Projekt. Diese Übung besteht aus verschiedenen Teilaufgaben. Dazu benötigen Sie diverse Text-Felder, Knöpfe usw. auf dem Layout der `MainActivity`. Die `MainActivity` soll am Schluss ungefähr so aussehen wie auf dem Screenshot rechts oben ersichtlich. Verwenden Sie eine `ScrollView` mit einem `LinearLayout`, siehe Übung 2 für Details dazu.

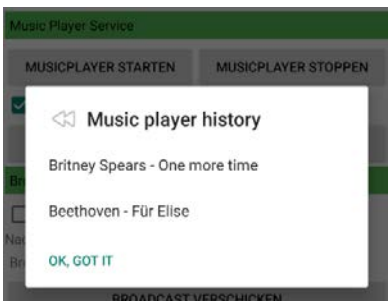
2. **Foreground Service: Music Player Service**
   Erstellen Sie eine eigene Service-Klasse `MusicPlayerService`. Bei Start (`Service`-Methode `onStartCommand`) soll (nach Prüfung, ob Service schon läuft) der Service mit `startForeground(...)` zum Vordergrund-Service befördert werden. Erstellen Sie dafür eine geeignete Notification.Optional können Sie hier auch noch einen Worker-Thread starten.





   Entfernen Sie den Service in `onDestroy` wieder aus dem Vordergrund und stoppen Sie einen allfälligen Worker-Thread. Fügen Sie bei den `Service`-Methoden `onCreate`, `onStartCommand` und `onDestroy` Log-Ausgab en ein, sodass Sie die Reihenfolge bei mehrmaligem Starten (durch Knopf „Service starten") von einem `MusicPlayerService` nachvollziehen können (siehe Folien dazu, bzw. Screenshot unten rechts).
   Fügen Sie auf der Main-Activity Knöpfe zum Starten/Stoppen des Services hinzu.

3. **Binden des MusicPlayerServices**
   Erweitern Sie ihren `MusicPlayerService`, so dass dieser Service ge bunden werden kann. Implementieren Sie also die Methode `onBind`. Die von dieser Methode zurückgelieferte `IBinder`-Instanz soll dabei folgendes Interface `DemoServiceApi` implementieren:

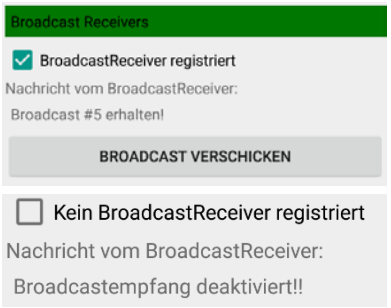



```
public interface MusicPlayerApi {
      String playNextItem();
      List<String> getHistory();
}
```

   Die Methode `playNextItem` () soll zufällig ein weiteres Stück auswählen und (simuliert) abspielen. Der Service soll

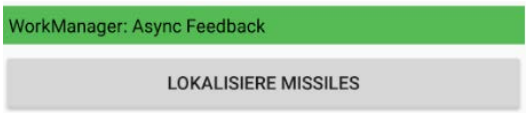
sich in einer Liste merken, welche Stücke ausgewählt und abgespielt wurden, diese Liste kann mit `getHistory()` abgefragt werden.
Binden Sie dann diesen Service an die `MainActivity`. Implementieren Sie dazu je eine Methode zum Binden und Unbinden von diesem Service in der `MainActivity`. Verwenden Sie dazu u.a. die Methode `bindService(...)` und implementieren Sie eine eigene Klasse `MusicPlayerConnection` als Unterklasse von `ServiceConnection`. Fügen Sie wie im Screenshot rechts oben angedeutet Ihrer `MainActivity` eine Checkbox zum Binden/Unbinden des `MusicPlayerServices` hinzu und zwei Knöpfe zum Abspielen des nächsten Stücks und zur Abfrage der Player History. Diese Funktionalität soll natürlich über den gebundenen Service zur Verfügung gestellt werden, zeigen Sie die Resultate als Toasts oder mit einem Dialog.

4. **Dynamischer Broadcast-Receiver**



Erweitern Sie ihre `MainActivity` so, dass diese auch Broadcast-Intents mit der Action `ACTION_MY_BROADCAST` empfangen kann. Die Registration für diesen `Intent` soll dynamisch im Code passieren und durch eine Checkbox gesteuert sein, siehe Screenshot rechts. Beim Empfang von einem entsprechenden Broadcast soll angezeigt werden, um den wievielten empfangenen Broadcast es sich dabei handelt, siehe Screenshot rechts. Falls die Checkbox nicht aktiviert ist, soll die `MainActivity` keine Broadcasts empfangen, eine entsprechende Nachricht anzeigen (siehe Screenshot rechts) und den Zähler auf 0 zurücksetzen.
Beim Klick auf den Knopf „Broadcast verschicken“ versenden Sie entweder implizite Broadcasts über den `LocalBroadcastManager` oder explizite Broadcasts (d.h. mittels `setPackage()` an Ihre App adressiert) über die Activity-Methode `sendBroadcast()`.

5. **[Optional] Hintergrund-Tasks mit WorkManager**



Fügen Sie der `MainActivity` einen weiteren Button hinzu, mit dem Sie einen lange dauernden Hintergrundtask starten können.
Benutzen Sie dazu eine selber definierte `Worker` Klasse, um die Arbeit zu erledigen. Benutzen Sie ggf. `Thread.sleep(…)` um Ihren Task künstlich zu verlängern und nutzen sie entweder einen globalen oder lokalen Broadcast, um die Resultate Ihres Work-Tasks zurückzumelden.
Empfangen Sie den Broadcast in Ihrer App (entweder via statisch registriertem BC-Receiver im Manifest oder via dynamisch registriertem BC-Receiver im Code) und stellen Sie die Resultate im UI dar oder laden Sie entsprechende Informationen nach, falls Sie den BC nur als Trigger verwenden.