



JÖNKÖPING UNIVERSITY

*School of Engineering*

---

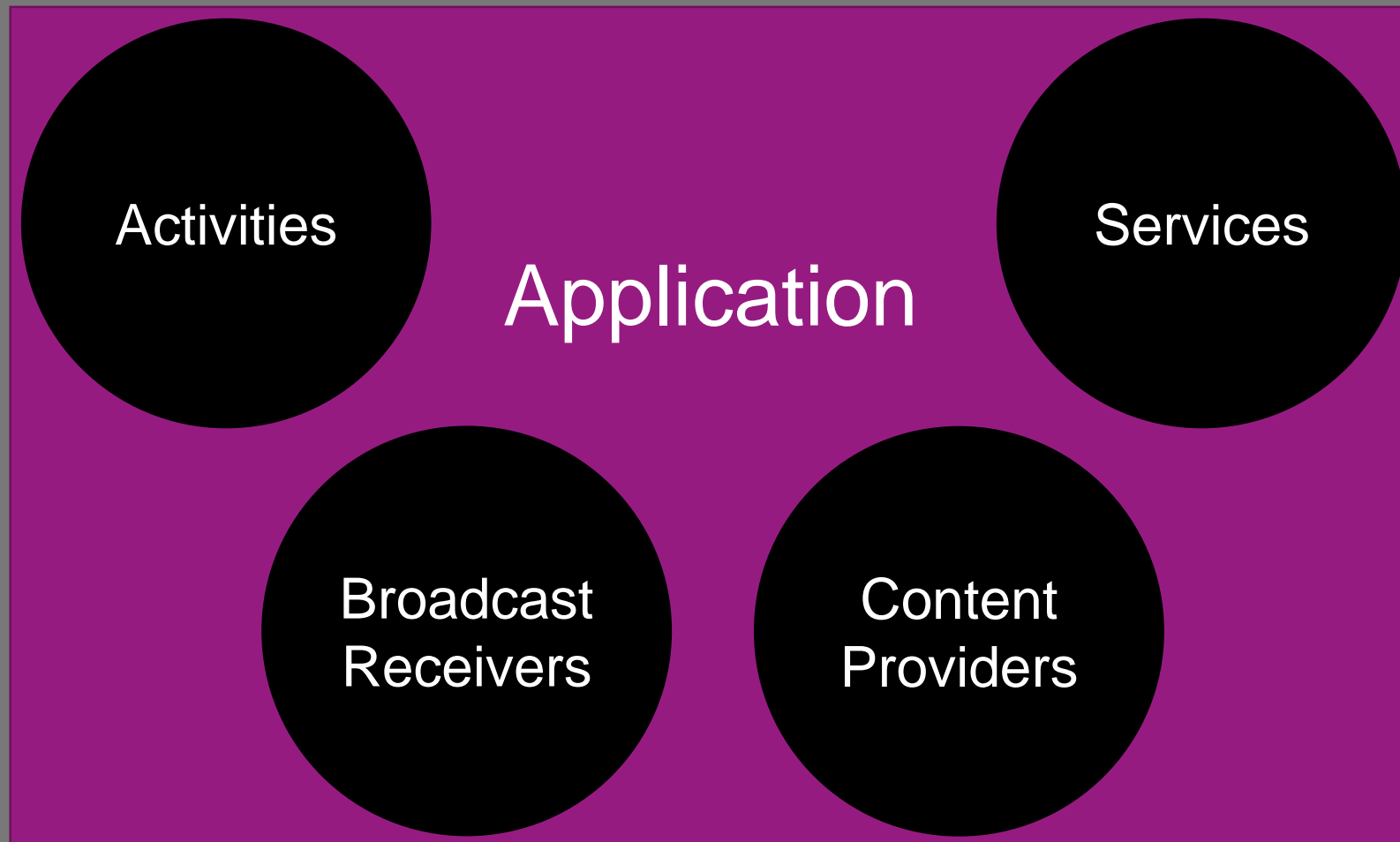
# ANDROID BROADCAST RECEIVERS

**Peter Larsson-Green**

Jönköping University

Spring 2020

# FUNDAMENTAL APP COMPONENTS



# WHAT'S A BROADCAST?

An intent sent from one application component or the OS...  
...and (possibly) received by broadcast receivers.

# BROADCASTS FROM ANDROID

The action specifies the type of broadcast:

- android.intent.action.NEW\_OUTGOING\_CALL
- android.intent.action.HEADSET\_PLUG
- android.intent.action.BOOT\_COMPLETED
- android.bluetooth.adapter.action.STATE\_CHANGED
- ...
- Permissions may be required to send broadcasts.
- Permissions may be required to receive broadcasts.

# SPOTIFY

## Play/Pause or track position change:

- **Action:** `com.spotify.music.playbackstatechanged`
- **Extras:** `playing`, `playbackPosition`.

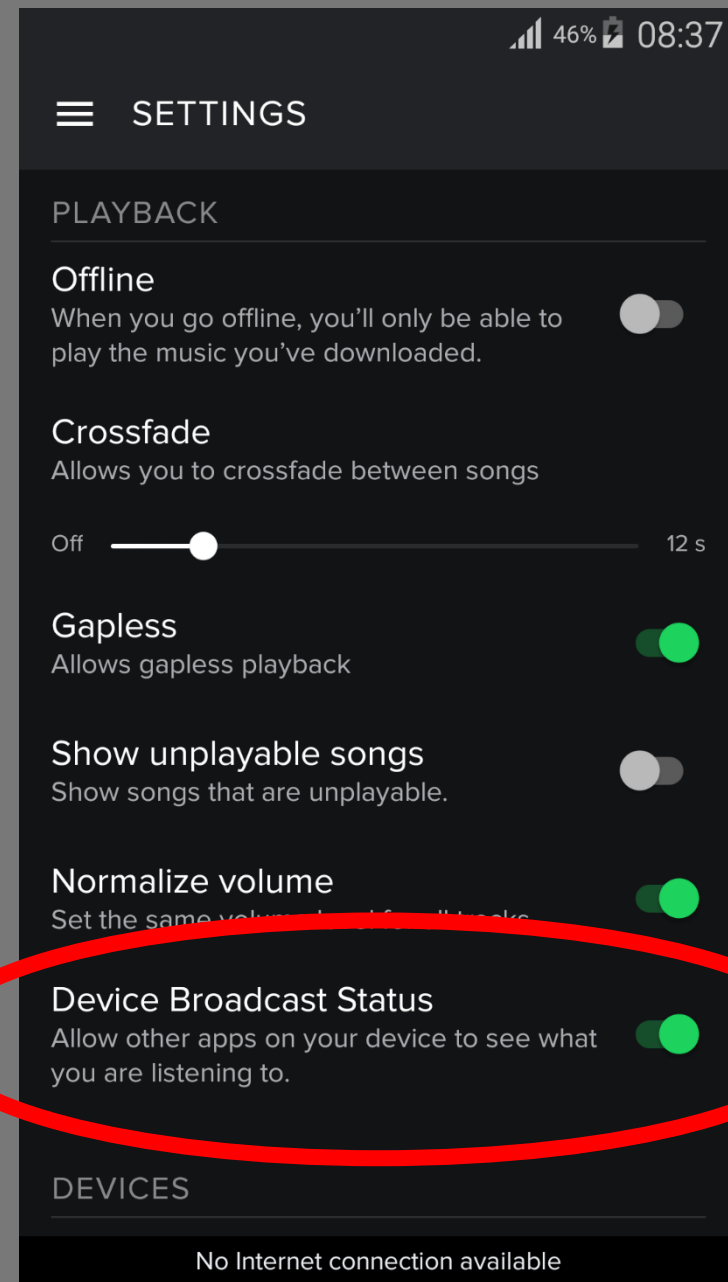
## New track starts playing:

- **Action:** `com.spotify.music.metadatachanged`
- **Extras:** `id`, `artist`, `album`, `track`, `length`.

## Play queue changed:

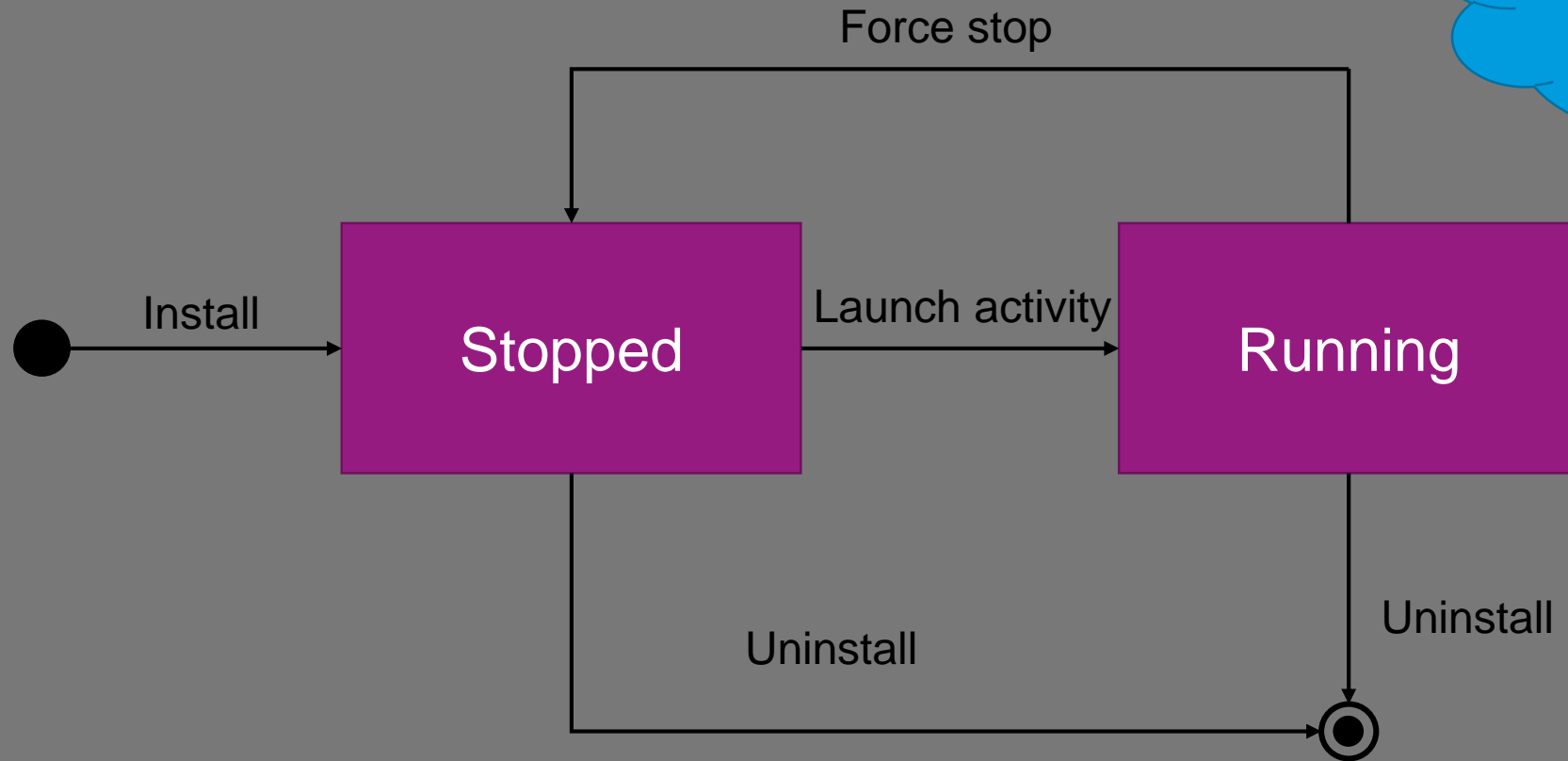
- **Action:** `com.spotify.music.queuechanged`

<https://developer.spotify.com/news-stories/2014/07/18/new-android-media-notifications-guide/>



# APPLICATION STATE

Only receivers whose app is in the Running state receives broadcasts.



More information: <https://developer.android.com/about/versions/android-3.1.html#launchcontrols>

# SENDING A BROADCAST

```
Intent intent = new Intent("action.to.be.handled");  
intent.putExtra("name", "Herbert A. Simon");  
aContext.sendBroadcast(intent);
```



# CREATING BROADCAST RECEIVERS

```
public class MyBroadcastReceiver extends BroadcastReceiver{  
    @Override  
    public void onReceive(Context context, Intent intent){  
        // Do your work!  
    }  
}
```

# CREATING BROADCAST RECEIVERS


```
<manifest package="the.package" ...>
  <application ...>
    <receiver android:name=".MyBroadcastReceiver">
      <intent-filter>
        <action android:name="the.action.name" />
      </intent-filter>
    </receiver>
  </application>
</manifest>
```

# IMPORTANT ABOUT RECEIVERS

- Runs on the main application thread.
- Is considered "done" when `onReceive()` returns.
  - Unless you call `goAsync()`.
- Global receivers must finish within 10 seconds.
  - Even if they call `goAsync()`.
    - A receiver having a very long long running operation can simply start a service.

# GOING ASYNCHRONOUS

```
public void onReceive(Context context, Intent intent) {  
    final PendingResult pendingResult = goAsync();  
    new Thread(new Runnable() {  
        @Override  
        public void run() {  
            // Do your work.  
            pendingResult.finish();  
        }  
    }).start();  
}
```



Must be  
called within  
10 seconds!

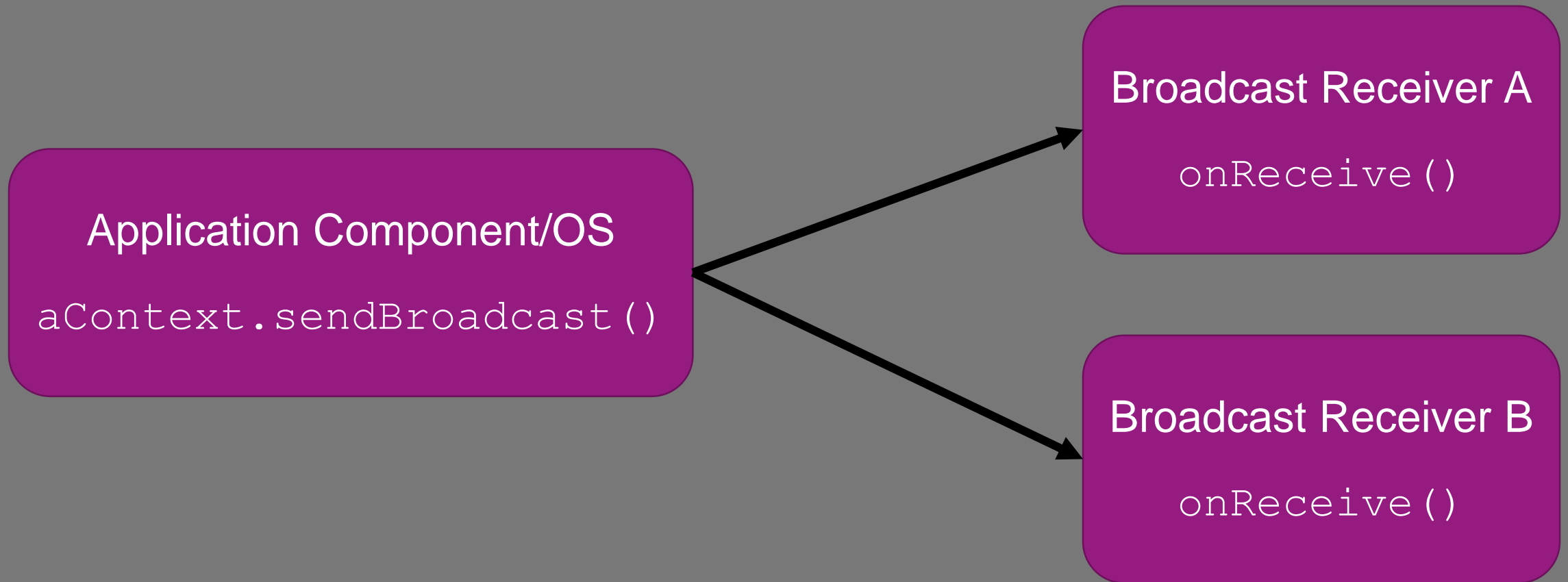
# DYNAMIC BROADCAST RECEIVERS

Only interested in broadcast when an activity is running?

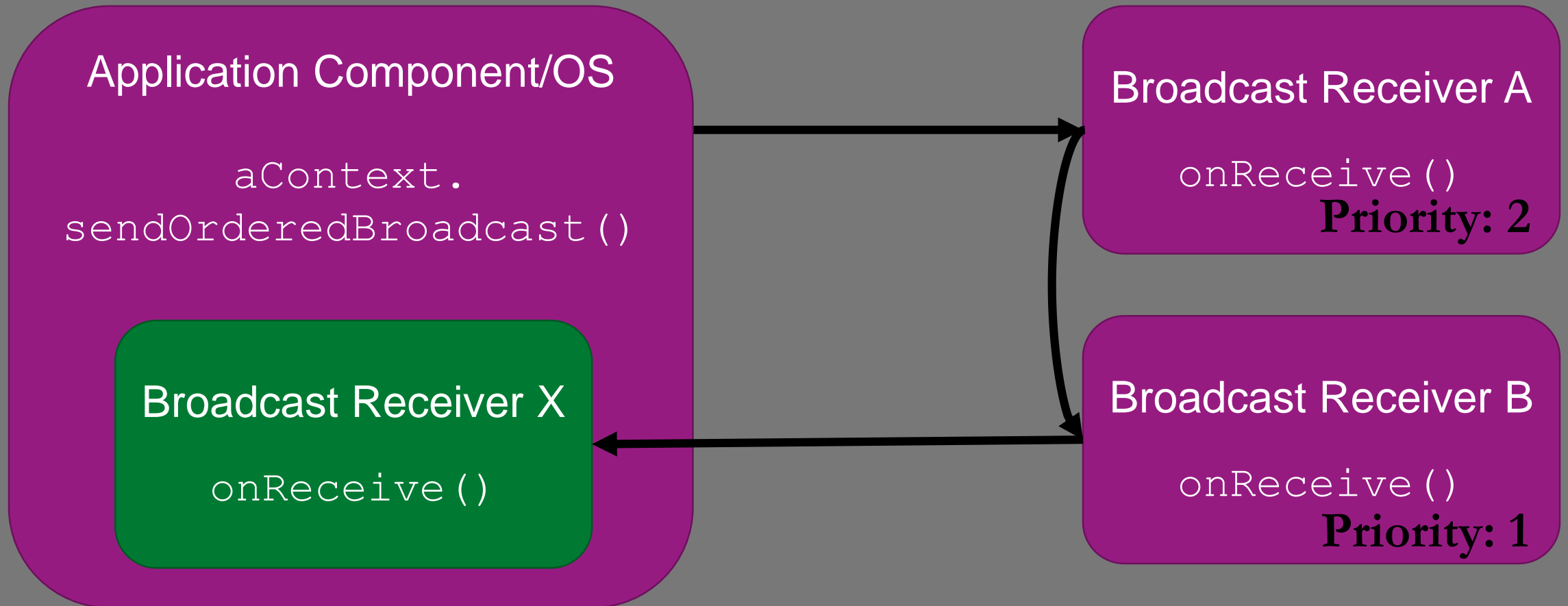
```
MyReceiver theReceiver = new MyReceiver();  
IntentFilter theIntentFilter = new IntentFilter();  
theIntentFilter.addAction("the.action.name");  
aContext.registerReceiver(theReceiver, theIntentFilter);
```

```
aContext.unregisterReceiver(theReceiver);
```

# NORMAL BROADCASTS



# ORDERED BROADCASTS



# SETTING THE PRIORITY

```
<manifest package="the.package" ...>
  <application ...>
    <receiver android:name=".MyBroadcastReceiver">
      <intent-filter android:priority="37">
        <action android:name="the.action.name" />
      </intent-filter>
    </receiver>
  </application>
</manifest>
```



# HANDLING ORDERED BROADCAST

- Stop propagation:

```
theReceiver.abortBroadcast();
```

getResult\*()

- Set the result:

```
theReceiver.setResultData("The Data.");
```

```
theReceiver.setResultCode(Activity.RESULT_OK);
```

```
theReceiver.setResultCode(Activity.RESULT_CANCELED);
```

```
theReceiver.setResultExtras(aBundle);
```

# SENDING ORDERED BROADCAST

```
aContext.sendOrderedBroadcast (  
    theIntent,  
    "a.permission.the.receivers.must.have",  
    broadcastResultReceiver,  
    scheduler, // A Handler.  
    initialResultCode,  
    initialResultData,  
    initialResultExtras  
);
```

Can be  
null.

Can be  
null.

Can be  
null.

Can be  
null.

# SENDING LOCAL BROADCASTS

## Global broadcasts

- Send it to all applications.


## Local broadcasts

- Send it only to your application.
- Not part of Android, but Support Library.

```
LocalBroadcastManager lbm =  
    LocalBroadcastManager.getInstance(aContext);  
lbm.registerReceiver(theReceiver, intentFilter);  
lbm.sendBroadcast(theIntent);  
lbm.unregisterReceiver(theReceiver);
```

# CUSTOM PERMISSIONS

```
<permission  
    android:name="my.package.THE_NAME"  
    android:protectionLevel="normal" ● ● ●  
    android:label="The label"  
    android:icon="@drawable/the_icon"  
    android:description="The description."  
>
```



normal  
dangerous  
signature