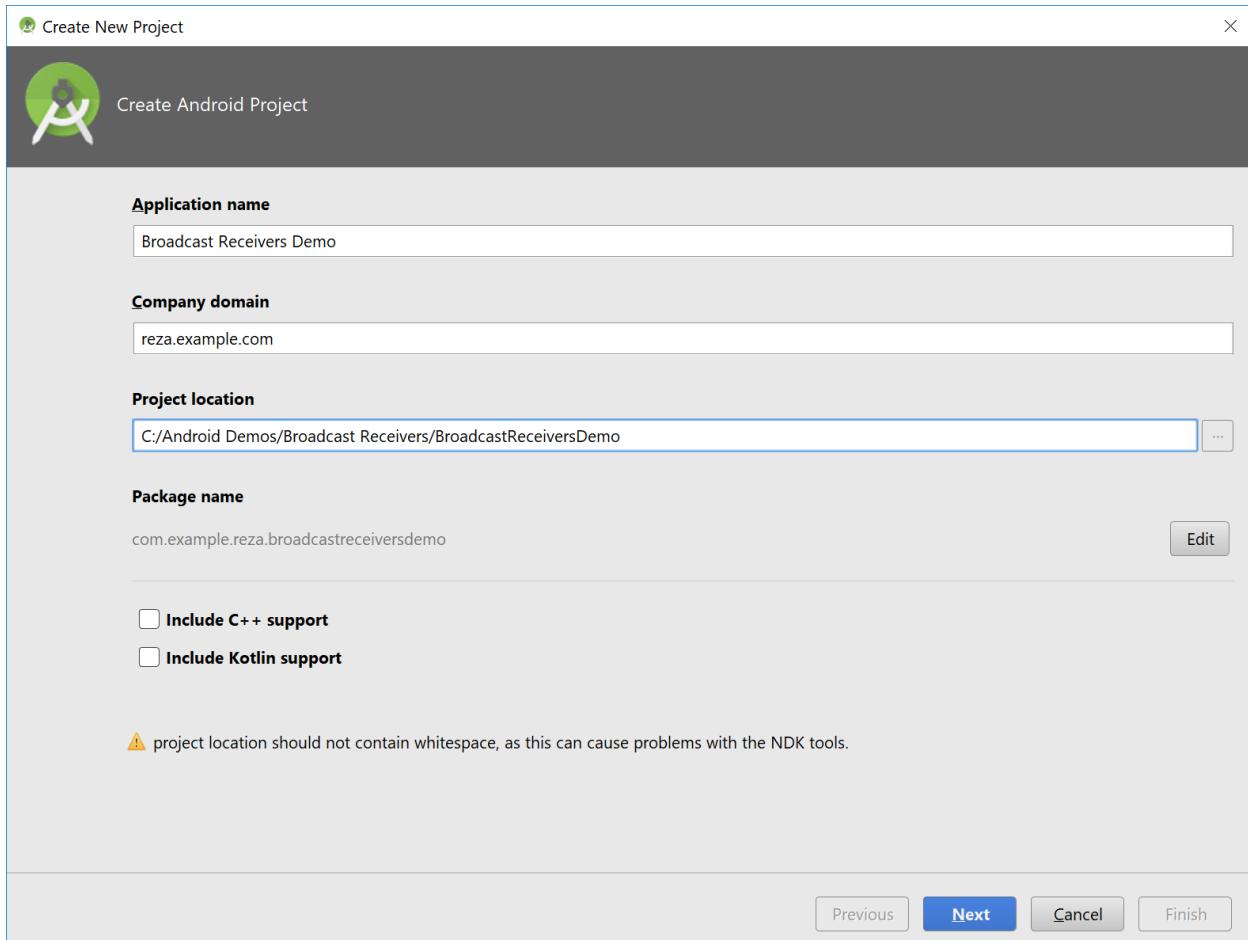


## Broadcast Receivers

Create a new project as shown:



Create New Project X

## Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

**Phone and Tablet**

API 15: Android 4.0.3 (IceCreamSandwich) ▼

By targeting **API 15 and later**, your app will run on approximately **100%** of devices. [Help me choose](#)

Include Android Instant App support

**Wear**

API 21: Android 5.0 (Lollipop) ▼

**TV**

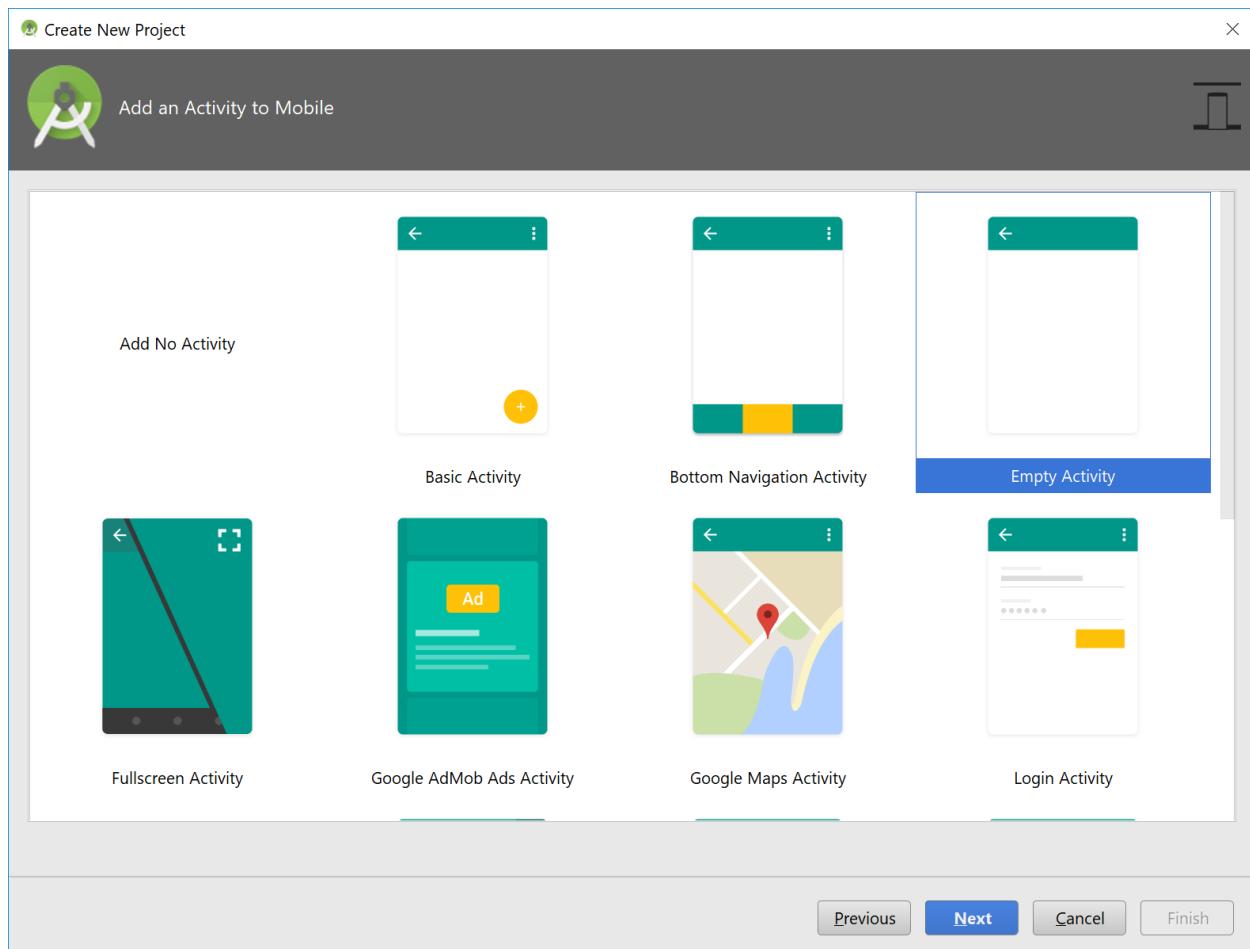
API 21: Android 5.0 (Lollipop) ▼

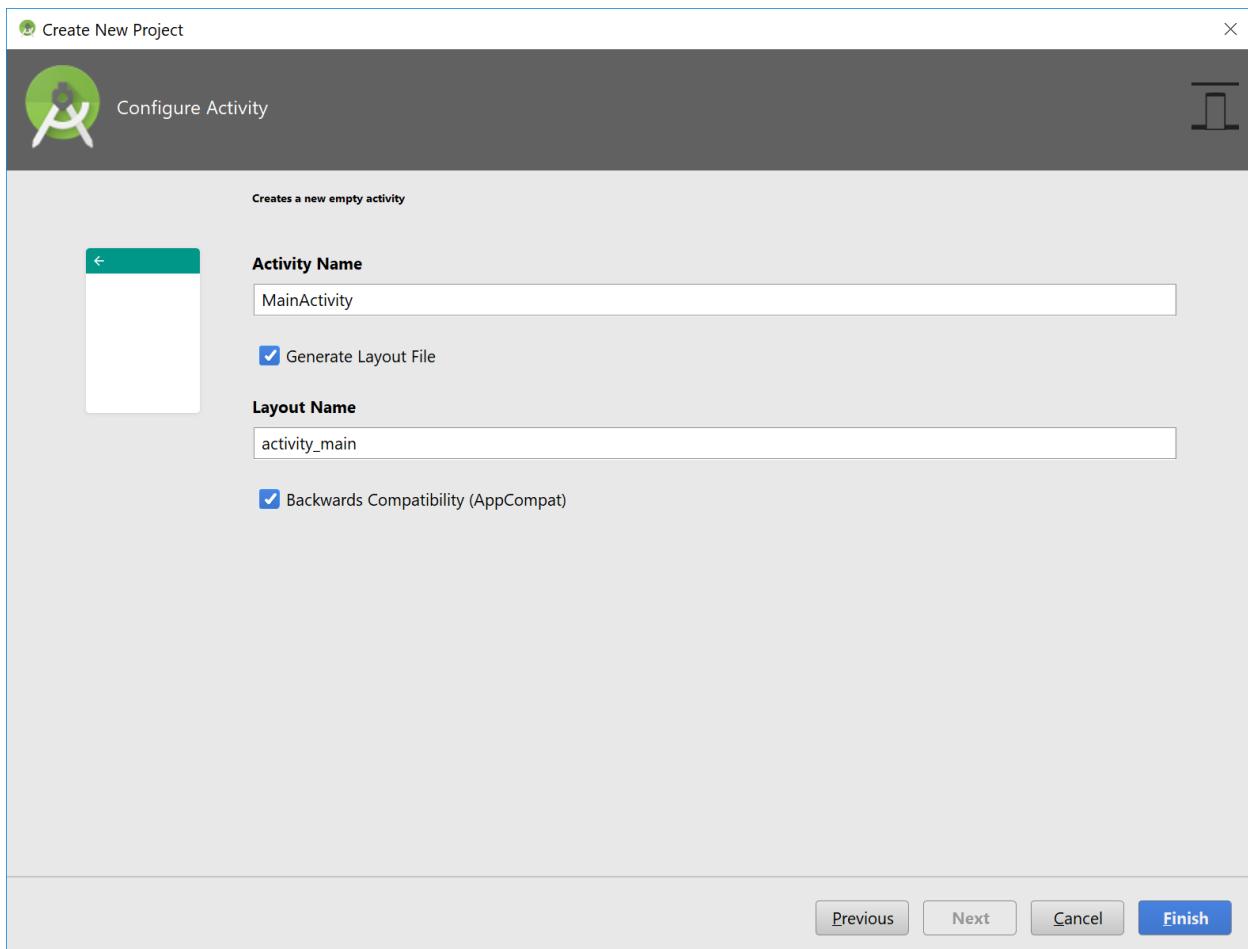
**Android Auto**

**Android Things**

API 24: Android 7.0 (Nougat) ▼

[Previous](#) Next [Cancel](#) [Finish](#)





The screenshot shows the Android Studio interface with the following details:

- File Bar:** BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...app\src\main\java\com\example\reza\broadcastreceiversdemo\Mai... File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
- Toolbar:** No, app, src, main, java, com, example, reza, broadcastreceiversdemo, MainActivity, app, Run, Stop, Refresh, Sync, Clean, Invalidate Caches, Device File Explorer, Find, Replace, Settings, Help.
- Project Navigators:** Project (Android), Captures, Favorites, Build Variants.
- Code Editor:** activity\_main.xml (Preview tab) and MainActivity.java. The code in MainActivity.java is:1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class MainActivity extends AppCompatActivity {
7
8 @Override
9 protected void onCreate(Bundle savedInstanceState) {
10 super.onCreate(savedInstanceState);
11 setContentView(R.layout.activity\_main);
12 }
13 }
14
- Toolbars:** Device File Explorer, Event Log, Gradle Console.
- Status Bar:** Gradle build finished in 5s 711ms (moments ago), 14:1 CRLF, UTF-8, Context: <no context>, Battery icon.

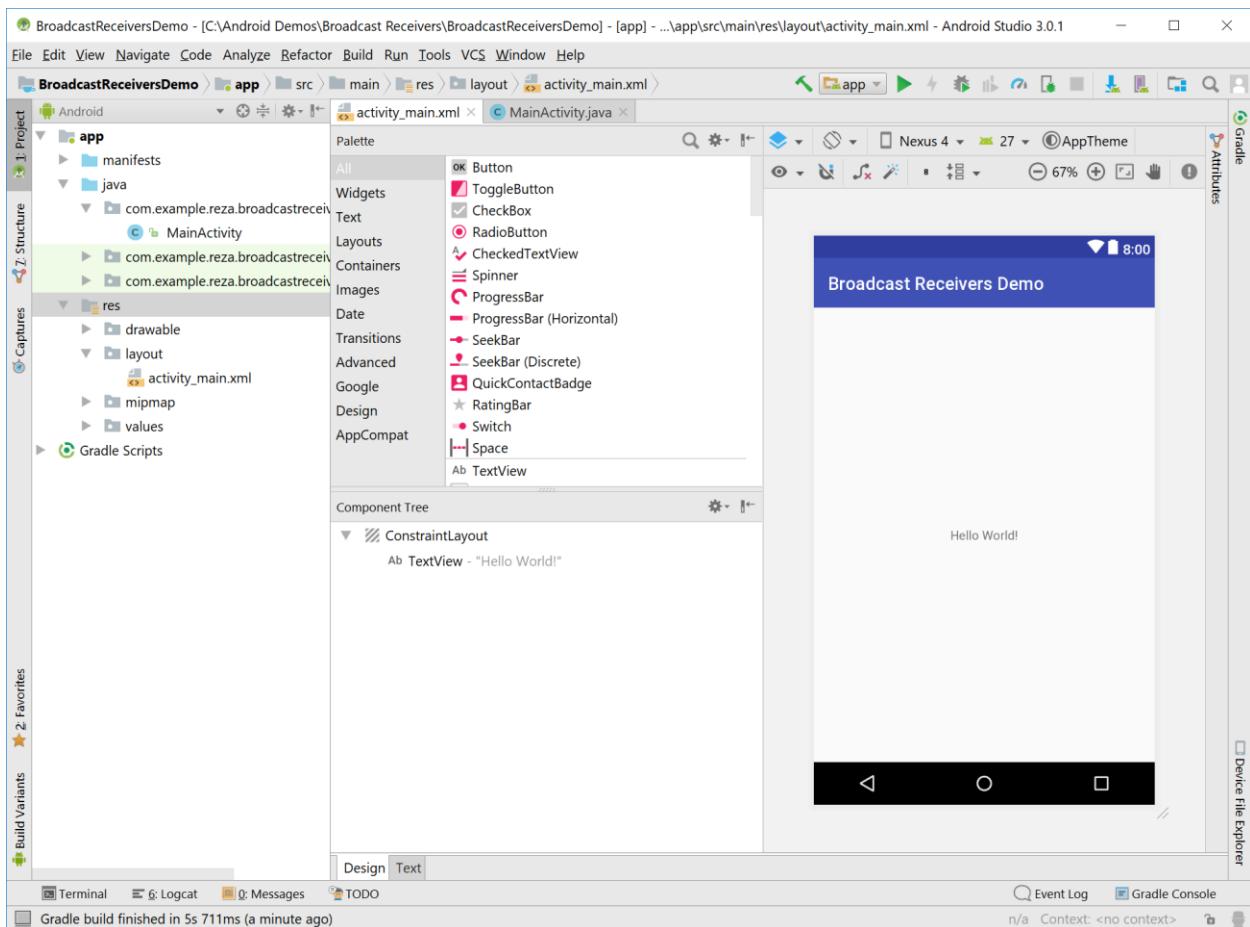
The screenshot shows the Android Studio interface with the following details:

- File Path:** BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...app\src\main\res\layout\activity\_main.xml - Android Studio 3.0.1
- Toolbar:** File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
- SIDE BAR:** Project, Captures, Favorites, Build Variants.
- Central Area:** Shows the XML code for `activity_main.xml`. The code defines a ConstraintLayout with a single TextView containing "Hello World!".

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.reza.broadcastreceiversdemo.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```
- Bottom Bar:** Design, Text, Terminal, Logcat, Messages, TODO, Event Log, Gradle Console.
- Status Bar:** Gradle build finished in 5s 711ms (moments ago), 1:1 CRLF, UTF-8, Context: <no context>, Device File Explorer.



Change the activity\_main.xml file as shown:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'activity\_main.xml' file is selected in the 'Text' tab. The code in the XML file is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.reza.broadcastreceiversdemo.MainActivity">

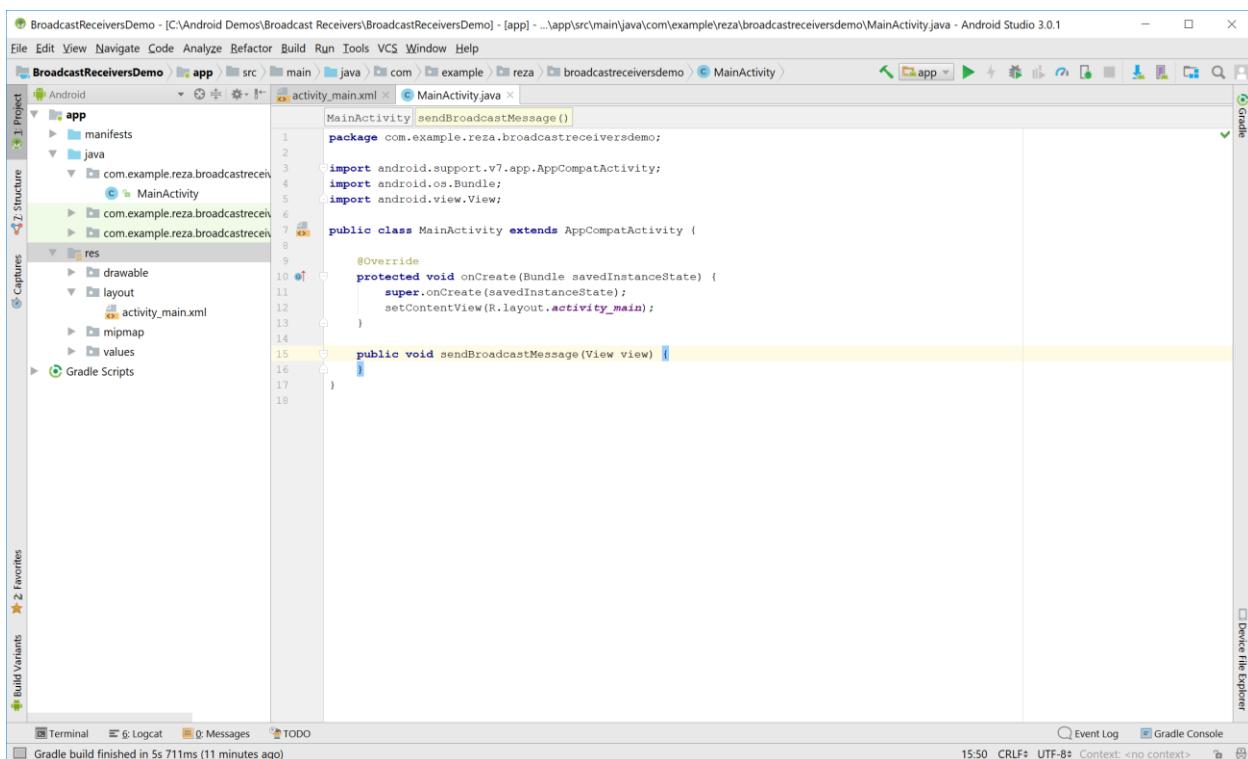
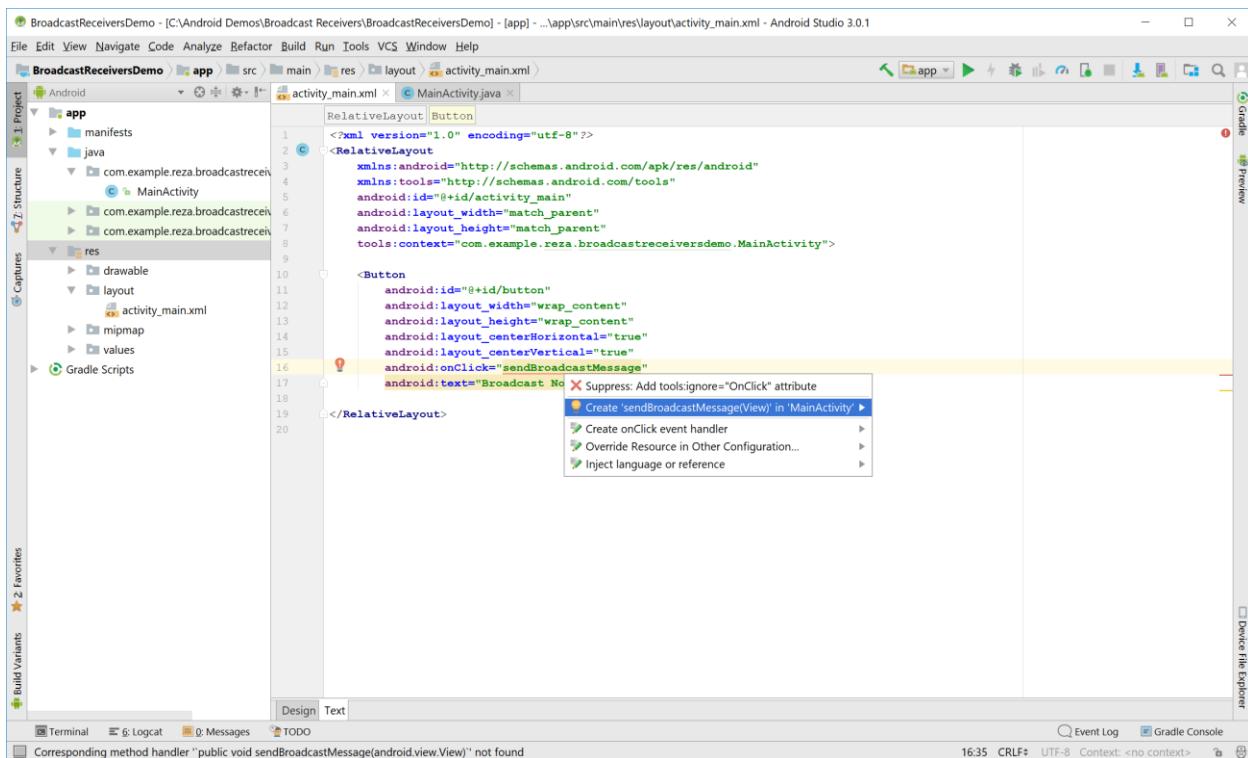
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:onClick="sendBroadcastMessage"
        android:text="Broadcast Now"/>

</RelativeLayout>
```

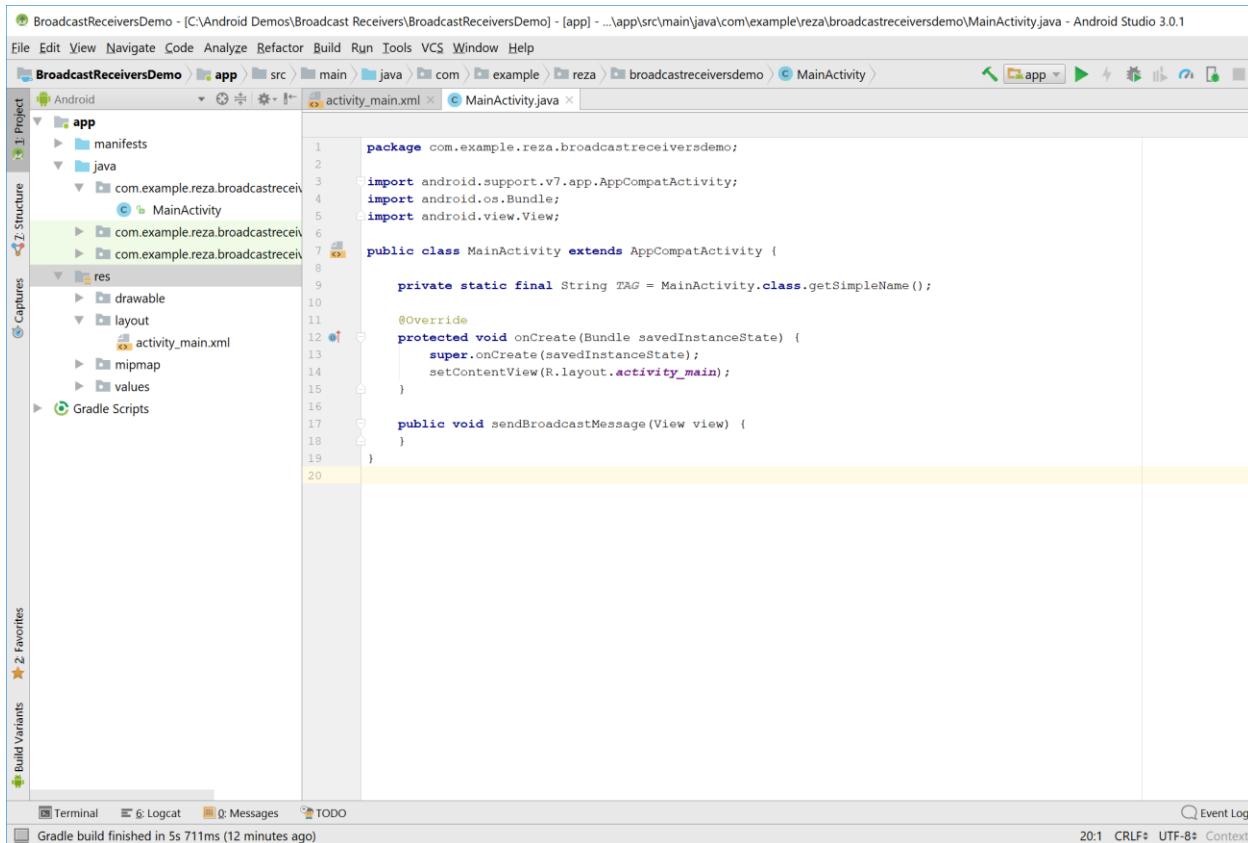
Press Alt + Enter:

The screenshot shows the same Android Studio interface. A context menu has been opened over the 'onClick="sendBroadcastMessage"' attribute in the XML code. The menu options are:

- Suppress: Add tools:ignore="OnClick" attribute
- Create 'sendBroadcastMessage(View)' in 'MainActivity'
- Create onClick event handler
- Override Resource in Other Configuration...
- Inject language or reference

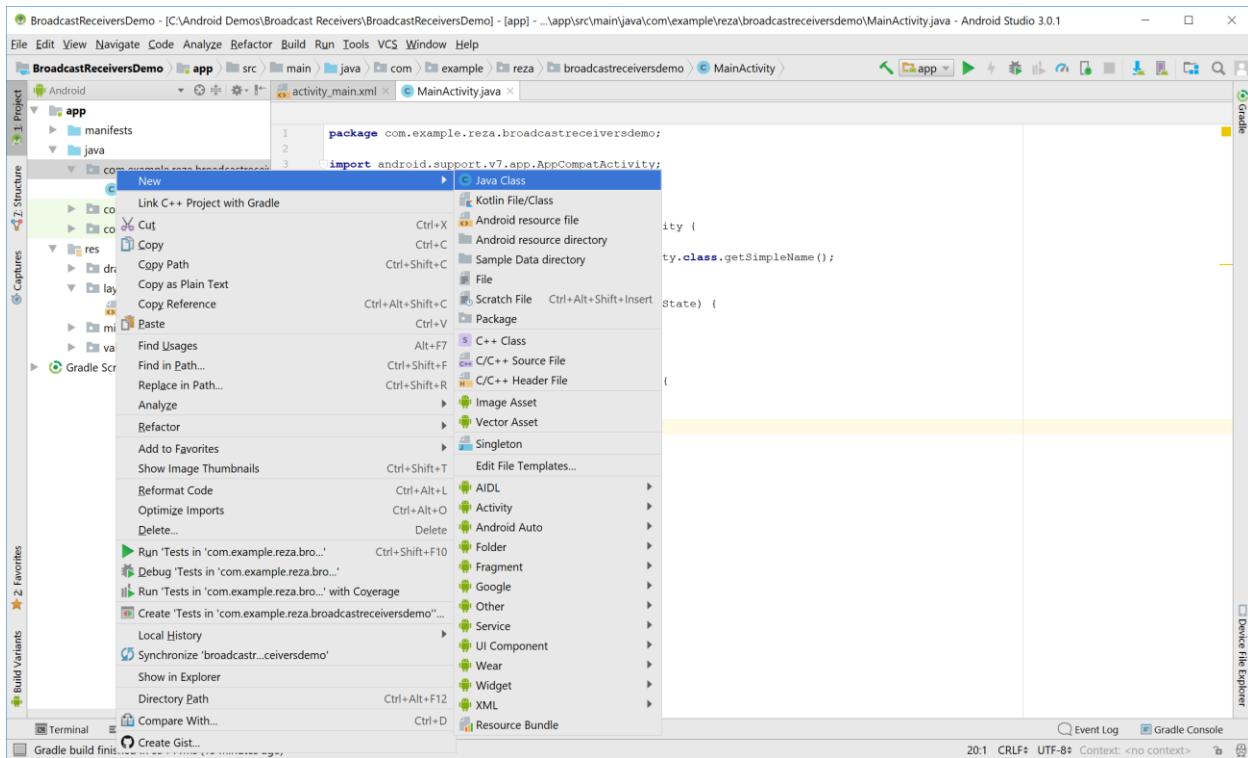


Add line 9 as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file. Line 9 contains the following code:

```
private static final String TAG = MainActivity.class.getSimpleName();
```



 Create New Class X

Name:

Kind:  Class ▼

Superclass:

Interface(s):

Package: com.example.reza.broadcastreceiversdemo

Visibility:  Public  Package Private

Modifiers:  None  Abstract  Final

Show Select Overrides Dialog

OK Cancel Help

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MyFirstReceiver.java - Android Studio 3.0.1

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo MyFirstReceiver
activity_main.xml MainActivity.java MyFirstReceiver.java

Project Z. Structure Build Variants Favorites
Gradle Scripts

1 package com.example.reza.broadcastreceiversdemo;
2
3 /**
4 * Created by Reza on 2018-03-23.
5 */
6
7 public class MyFirstReceiver {
8 }
9

```

Device File Explorer

Terminal Logcat Messages TODO Event Log Gradle Console

Class 'MyFirstReceiver' is never used

7:29 CRLF+ UTF-8+ Context: <no context>

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MyFirstReceiver.java - Android Studio 3.0.1

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo MyFirstReceiver
activity_main.xml MainActivity.java MyFirstReceiver.java

Project Z. Structure Build Variants Favorites
Gradle Scripts

1 package com.example.reza.broadcastreceiversdemo;
2
3 /**
4 * Created by Reza on 2018-03-23.
5 */
6
7 public class MyFirstReceiver extends Br {
8 }
9

```

BrokenBarrierException (java.util.concurrent)  
BroadcastReceiver (android.content)  
Browser (android.provider)  
BrowserRoot (android.service.media.MediaBrowserService)  
BrowserRoot (android.support.v4.media.MediaBrowserServiceCompat)  
BrowserCompatHostnameVerifier (org.apache.http.conn.ssl)  
MediaBrowser (android.media.browse)  
MediaBrowserService (android.service.media)  
LocalBroadcastManager (android.support.v4.content)  
MediaBrowserCompat (android.support.v4.media)

Device File Explorer

Terminal Logcat Messages TODO Event Log Gradle Console

Cannot resolve symbol 'Br'

7:41 CRLF+ UTF-8+ Context: <no context>

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the file 'MyFirstReceiver.java' which contains the following code:

```
MyFirstReceiver
package com.example.reza.broadcastreceiversdemo;

import android.content.BroadcastReceiver;

/**
 * Created by Reza on 2018-03-23.
 */
public class MyFirstReceiver extends BroadcastReceiver {
```

A yellow background highlights the entire code block. A red warning icon is visible in the top right corner of the code area. The status bar at the bottom shows the message: 'Class 'MyFirstReceiver' must either be declared abstract or implement abstract method 'onReceive(Context, Intent)' in 'BroadcastReceiver''.

Press Alt + Enter:

The screenshot shows the same Android Studio interface after pressing Alt + Enter. A context menu has appeared over the warning message in the code editor. The menu items are:

- Implement methods
- Make 'MyFirstReceiver' abstract
- Add broadcast receiver to manifest
- Create Test
- Create subclass
- Unimplement Class
- Annotate class 'BroadcastReceiver' as @Deprecated



## Select Methods to Implement



▼ (c) android.content.BroadcastReceiver

(m) onReceive(context:Context, intent:Intent):void



Copy JavaDoc



Insert @Override

OK

Cancel

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "BroadcastReceiversDemo". The "app" module contains "Manifests", "src" (with Java and XML files), and "res" (with drawable, layout, and values folders).
- Code Editor:** The file "MyFirstReceiver.java" is open. The code defines a class "MyFirstReceiver" that extends "BroadcastReceiver". It overrides the "onReceive" method to receive intents.

```
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6
7 /**
8 * Created by Reza on 2018-03-23.
9 */
10
11 public class MyFirstReceiver extends BroadcastReceiver {
12     @Override
13     public void onReceive(Context context, Intent intent) {
14     }
15 }
16
17
```

- Toolbars and Menus:** Standard Android Studio menus like File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help are visible at the top.
- Bottom Bar:** Includes tabs for Terminal, Logcat, Messages, TODO, Event Log, and Gradle Console, along with build status and time information.



## Select Classes to Import



The code fragment which you have pasted uses classes that are not accessible by imports in the new context.

Select classes that you want to import to the new file.



android.util.Log



android.widget.Toast

OK

Cancel

Help

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MyFirstReceiver.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo > app > src > main > java > com > example > reza > broadcastreceiversdemo > MyFirstReceiver >
activity_main.xml MainActivity.java MyFirstReceiver.java

MyFirstReceiver onReceive()
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6 import android.util.Log;
7 import android.widget.Toast;
8
9 /**
10 * Created by Reza on 2018-03-23.
11 */
12
13 public class MyFirstReceiver extends BroadcastReceiver {
14
15     private static final String TAG = MyFirstReceiver.class.getSimpleName();
16
17     @Override
18     public void onReceive(Context context, Intent intent) {
19
20         Log.i(TAG, msg:"Hello from 1st Receiver");
21         Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();
22
23     }
24
25 }
26

```

Event Log Gradle Console

Gradle build finished in 5s 711ms (22 minutes ago)

21:45 CRLF+ UTF-8 Context: <no context>

## Your current manifest file:

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\AndroidManifest.xml - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo > app > src > main > AndroidManifest.xml >
activity_main.xml MainActivity.java MyFirstReceiver.java AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

```

Event Log Gradle Console

Gradle build finished in 5s 711ms (23 minutes ago)

1:1 CRLF+ UTF-8 Context: <no context>

## Modify the file as shown:

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
            </intent-filter>
            <category android:name="android.intent.category.LAUNCHER" />
        </activity>
    <!-- receiver -->
    <receiver android:name=".MyFirstReceiver" />
    <!-- provider -->
    <!-- service -->
    <uses-library
        Press Ctrl+Space to view tags from other namespaces &lt;&gt;>
    </uses-library>
</manifest>

```

Documentation for MyFirstReceiver

Did you know that Quick Documentation View (Ctrl+Q) works in completion lookups as well? >>

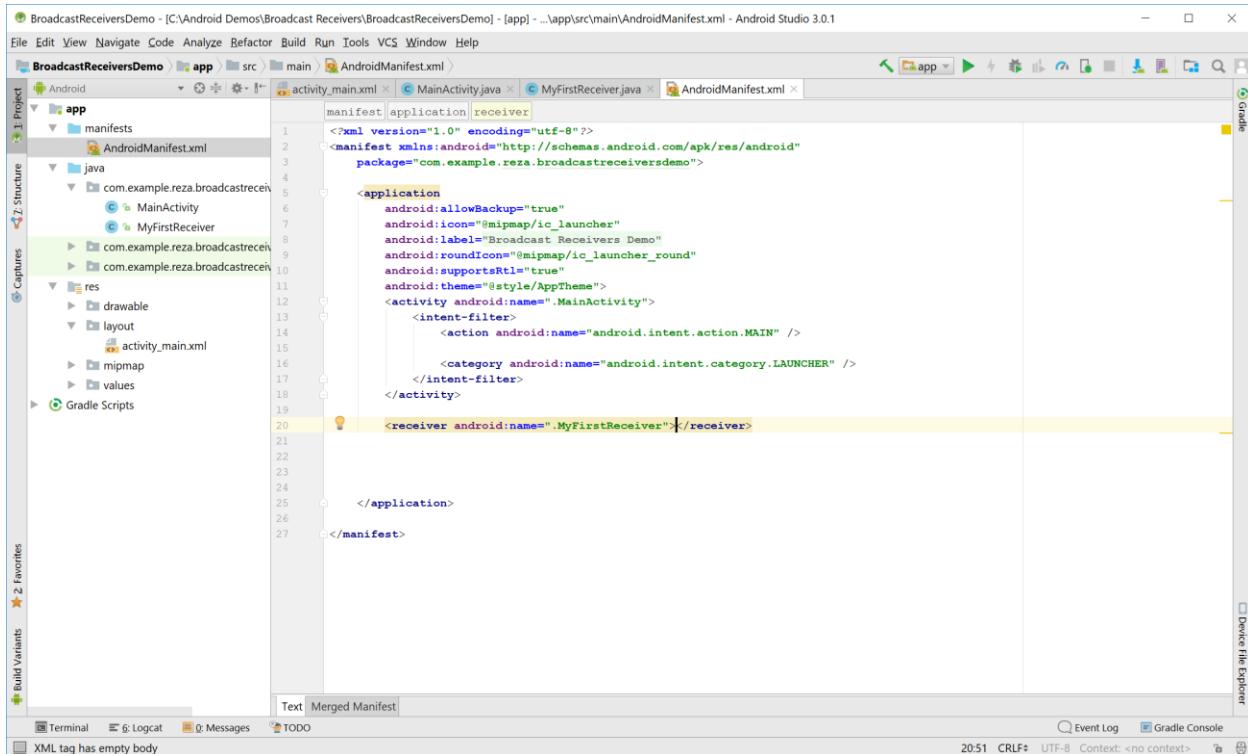
```

<receiver android:name=".MyFirstReceiver" />

```

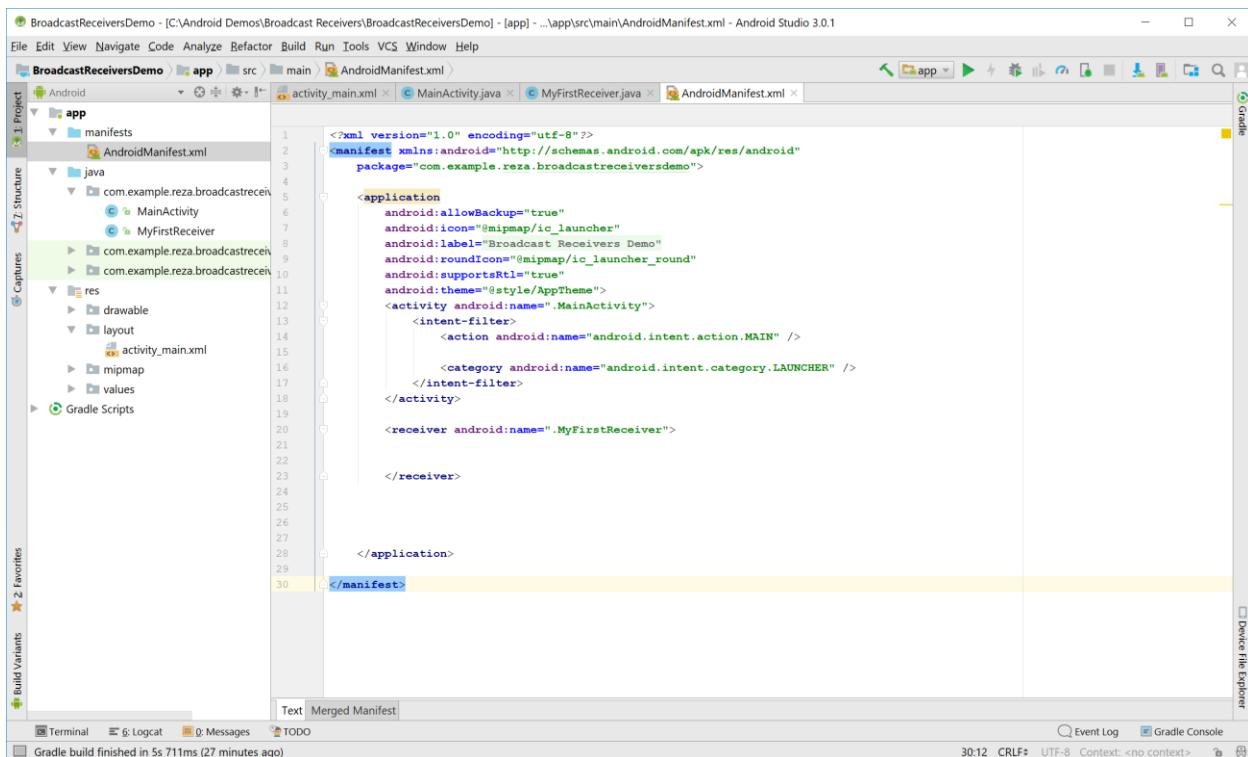
**com.example.reza.broadcastreceiversdemo**

public class MyFirstReceiver  
extends BroadcastReceiver  
Created by Reza on 2018-03-23.



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The left sidebar displays the project structure under the 'app' module. The main editor window shows the 'AndroidManifest.xml' file. A specific line of code, containing the declaration of a broadcast receiver, is highlighted in yellow:

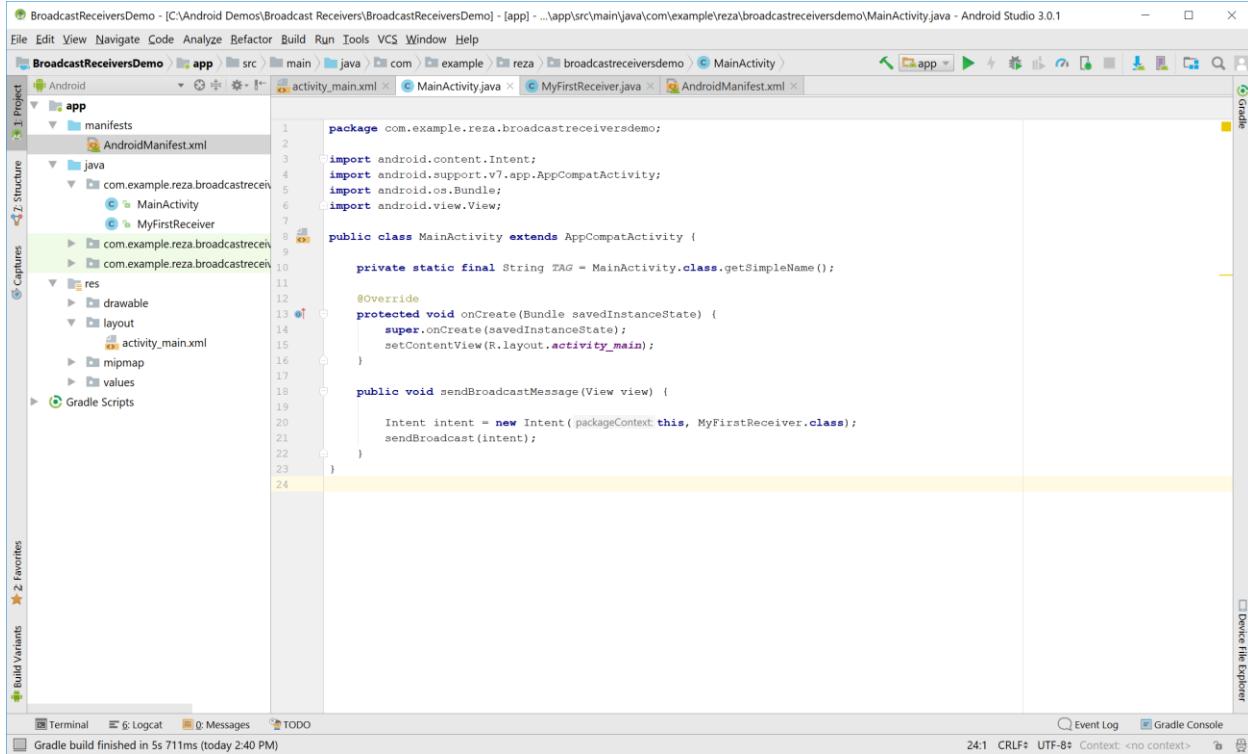
```
<receiver android:name=".MyFirstReceiver"></receiver>
```



The screenshot shows the same Android Studio interface after the receiver node has been moved. The 'Text' tab of the code editor is selected, showing the XML code. The receiver node is now located directly under the manifest tag, outside the application tag:

```
<receiver android:name=".MyFirstReceiver"></receiver>
```

Now you need to write code to send a Broadcast from your activity to your receiver and to do that you need to write your code in sendBroadcastMessage() method in the main activity:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file, which contains the following Java code:

```
package com.example.reza.broadcastreceiversdemo;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = MainActivity.class.getSimpleName();

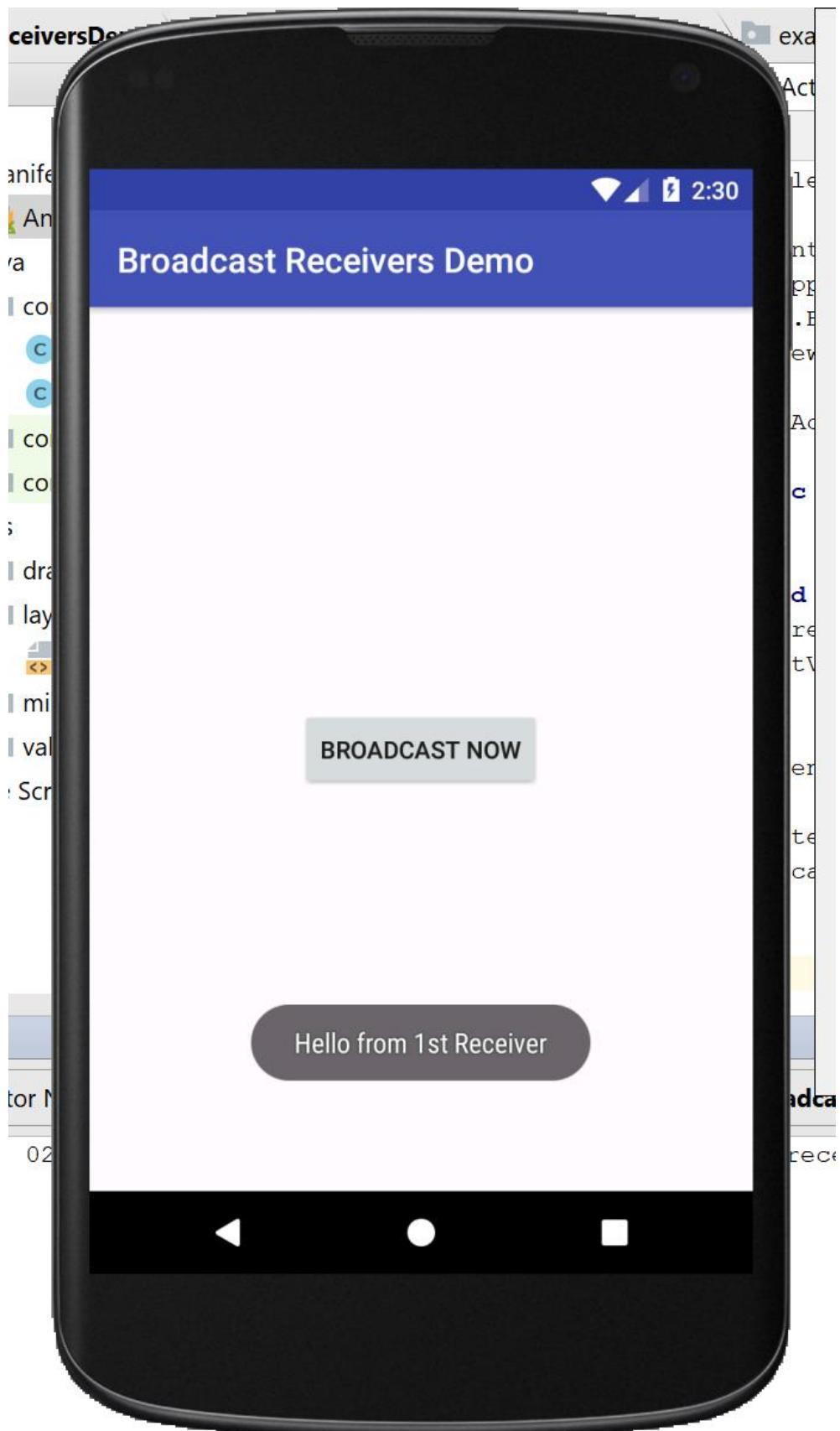
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }
}
```

The code editor highlights the line 'Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);' with a yellow background. The bottom status bar indicates 'Gradle build finished in 5s 711ms (today 2:40 PM)'.

Now run the application:





The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the file `MainActivity.java` which contains the following Java code:

```
package com.example.reza.broadcastreceiversdemo;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }
}
```

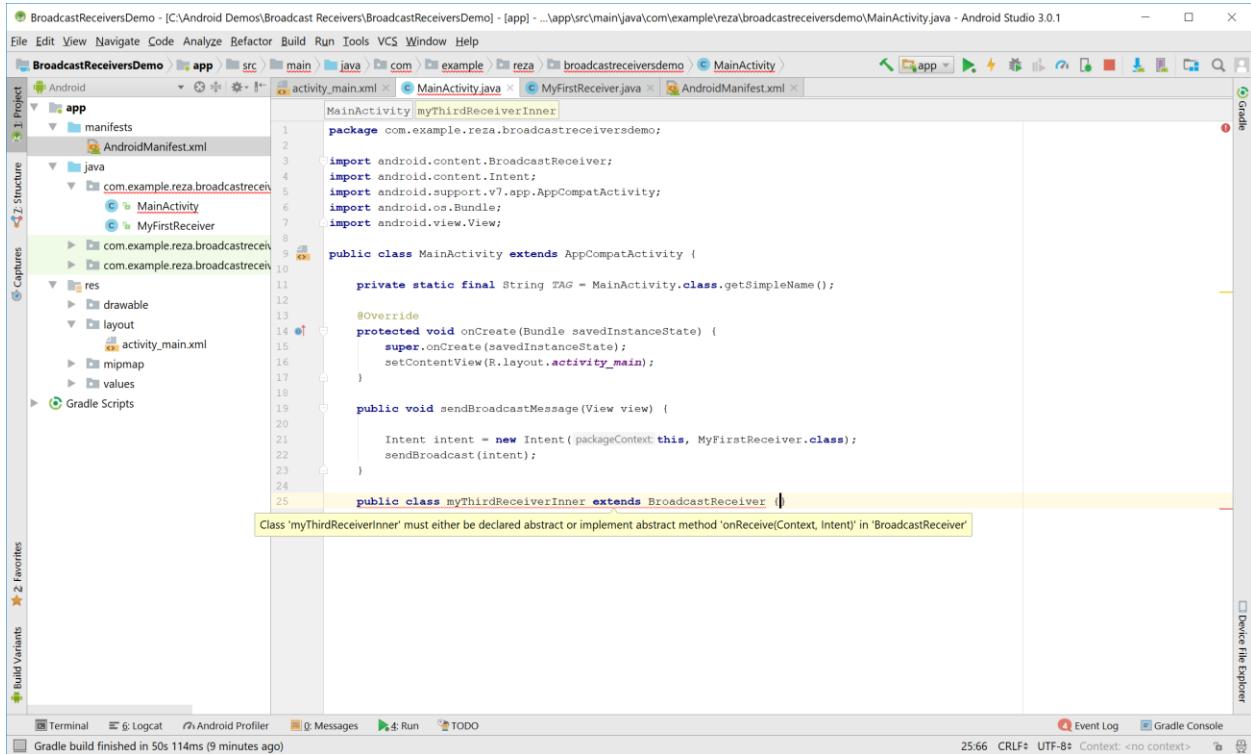
The Logcat tab shows the output from the emulator:

```
03-24 02:30:13.849 4408-4408/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
```

So everything is working.

## Declaring the Broadcast as an inner class

You can also declare your receiver inside the activity as shown below:

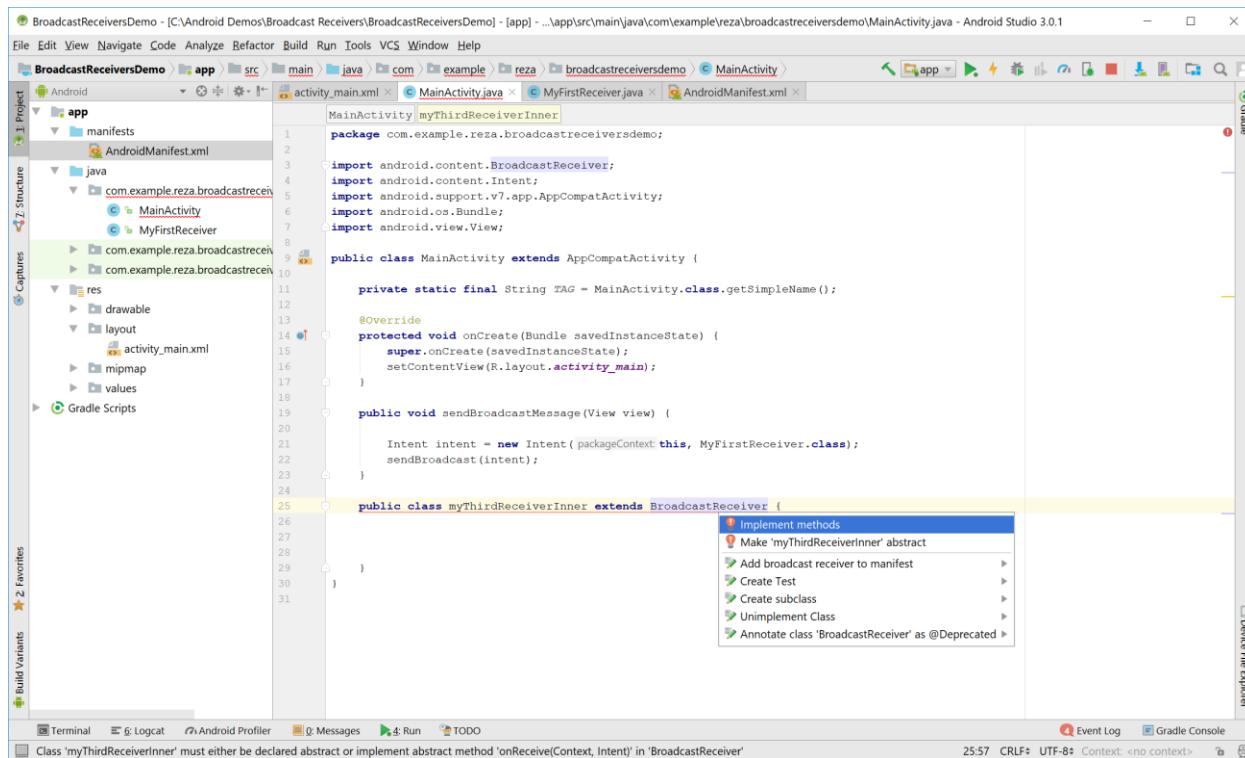


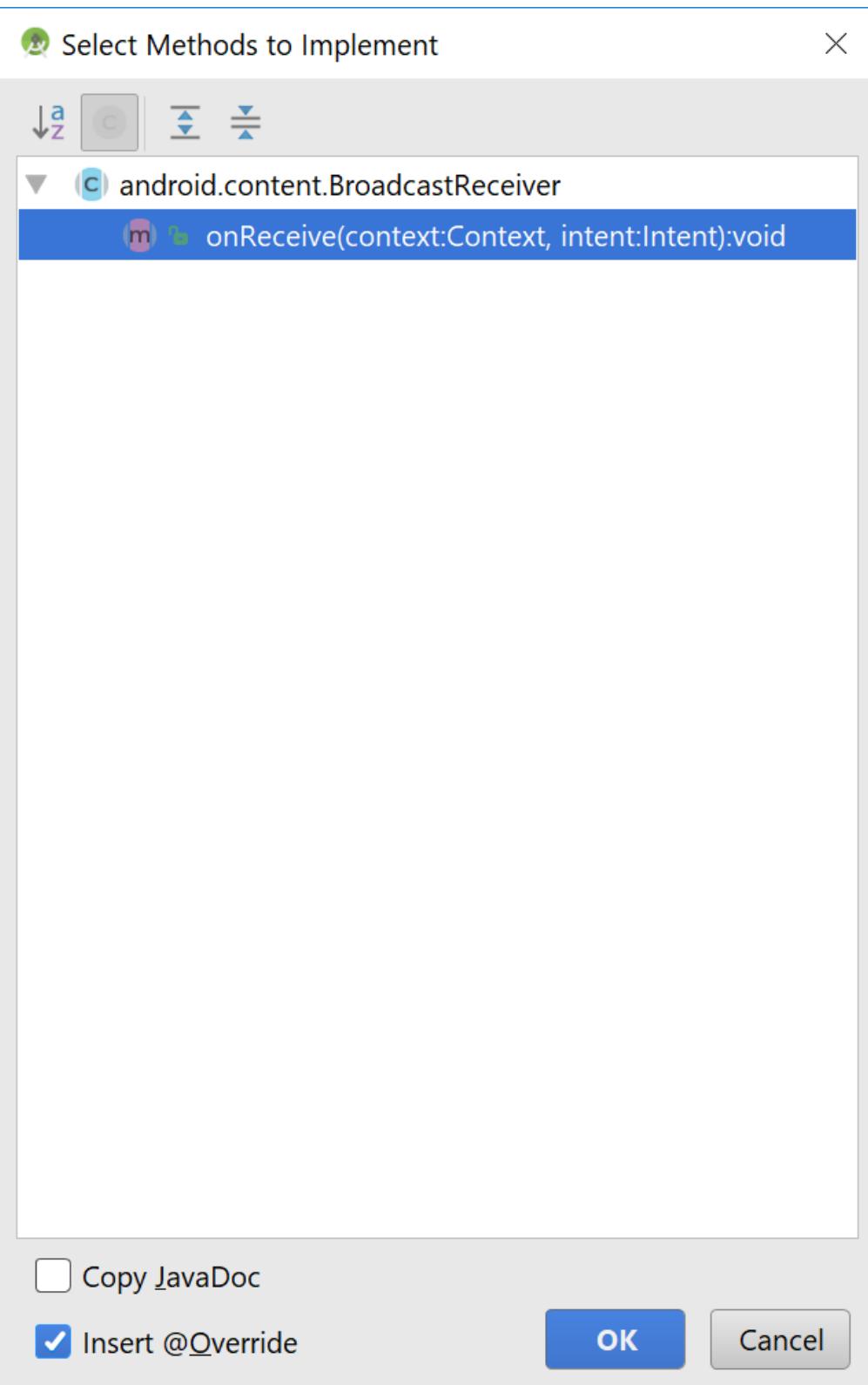
The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays `MainActivity.java`. The code defines a class `myThirdReceiverInner` which extends `BroadcastReceiver`. This class must implement the abstract method `onReceive(Context, Intent)`, as indicated by the error message in the code editor.

```
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Intent;
5 import android.support.v7.app.AppCompatActivity;
6 import android.os.Bundle;
7 import android.view.View;
8
9 public class MainActivity extends AppCompatActivity {
10
11     private static final String TAG = MainActivity.class.getSimpleName();
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17     }
18
19     public void sendBroadcastMessage(View view) {
20
21         Intent intent = new Intent(packageContext: this, MyFirstReceiver.class);
22         sendBroadcast(intent);
23     }
24
25     public class myThirdReceiverInner extends BroadcastReceiver {
26
27     }
28 }
```

Class 'myThirdReceiverInner' must either be declared abstract or implement abstract method 'onReceive(Context, Intent)' in 'BroadcastReceiver'

Press Alt + Enter:





```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo MainActivity.java
activity_main.xml MainActivity.java MyFirstReceiver.java AndroidManifest.xml
MainActivity myThirdReceiverInner onReceive()
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
package com.example.reza.broadcastreceiversdemo;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getSimpleName();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }
    public class myThirdReceiverInner extends BroadcastReceiver {
        @Override
        public void onReceive(Context context, Intent intent) {
        }
    }
}

```

Now go to the manifest file and declare your new receiver as shown:

```

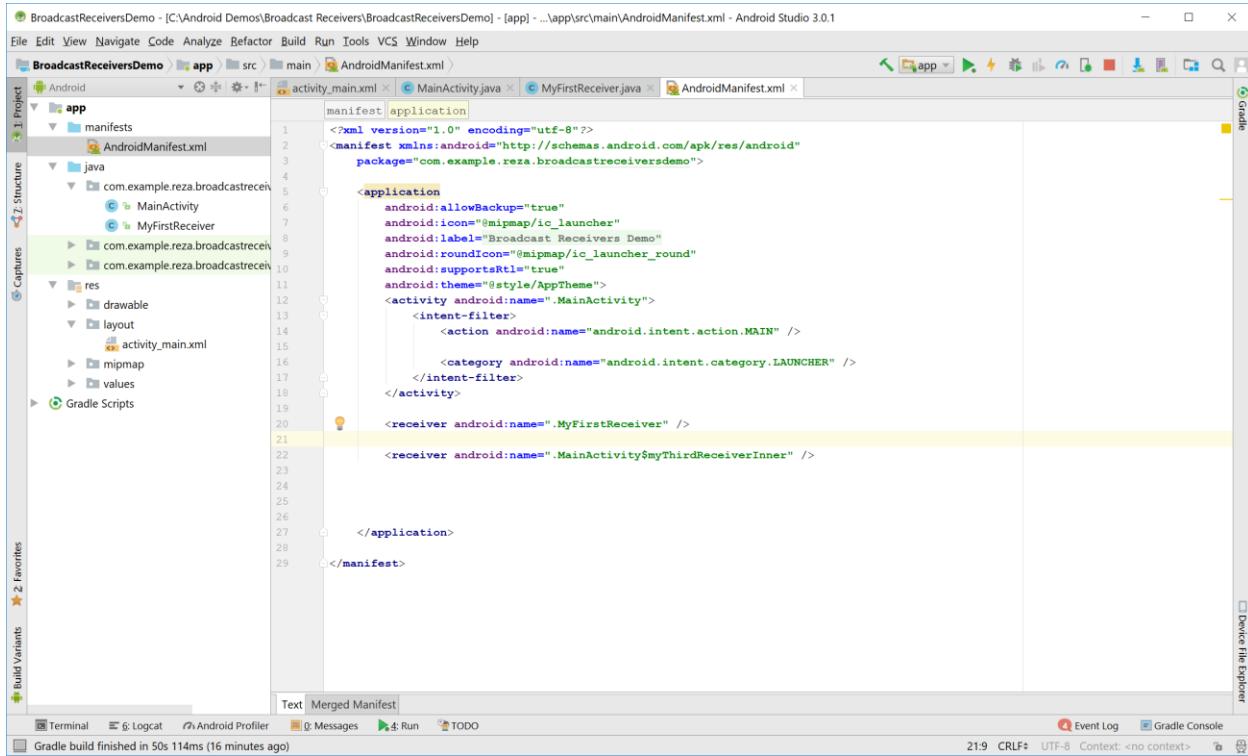
BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\AndroidManifest.xml - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main AndroidManifest.xml
activity_main.xml MainActivity.java MyFirstReceiver.java AndroidManifest.xml
manifest application receiver
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".MyFirstReceiver">
        </receiver>
        <receiver android:name=".MainActivity$MyThirdReceiverInner" (com.example.reza.broadcastreceiversdemo...)>
        <MyFirstReceiver (com.example.reza.broadcastreceiversdemo...)>
        </receiver>
    </application>
</manifest>

```

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The main window displays the `AndroidManifest.xml` file. A code completion dropdown is visible, listing `com.example.reza.broadcastreceiversdemo.MainActivity$MyThirdReceiverInner` and `com.example.reza.broadcastreceiversdemo.MyFirstReceiver`. The manifest file contains XML code defining an application with activities and receivers.

```
<manifest>
    <application>
        <activity android:name=".MainActivity" />
        <activity android:name=".MyFirstReceiver" />
        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </receiver>
        <receiver android:name=".MainActivity$MyThirdReceiverInner">
            <intent-filter>
                <action android:name="com.example.reza.broadcastreceiversdemo.MY_THIRD_RECEIVER_ACTION" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

Notice the proper syntax to access the inner class inside the main activity (“`.MainActivity$myThirdReceiverInner`”) as shown. Note the way we are accessing the inner class, indicates that this class is static so we need to add the static keyword in our main activity.



```

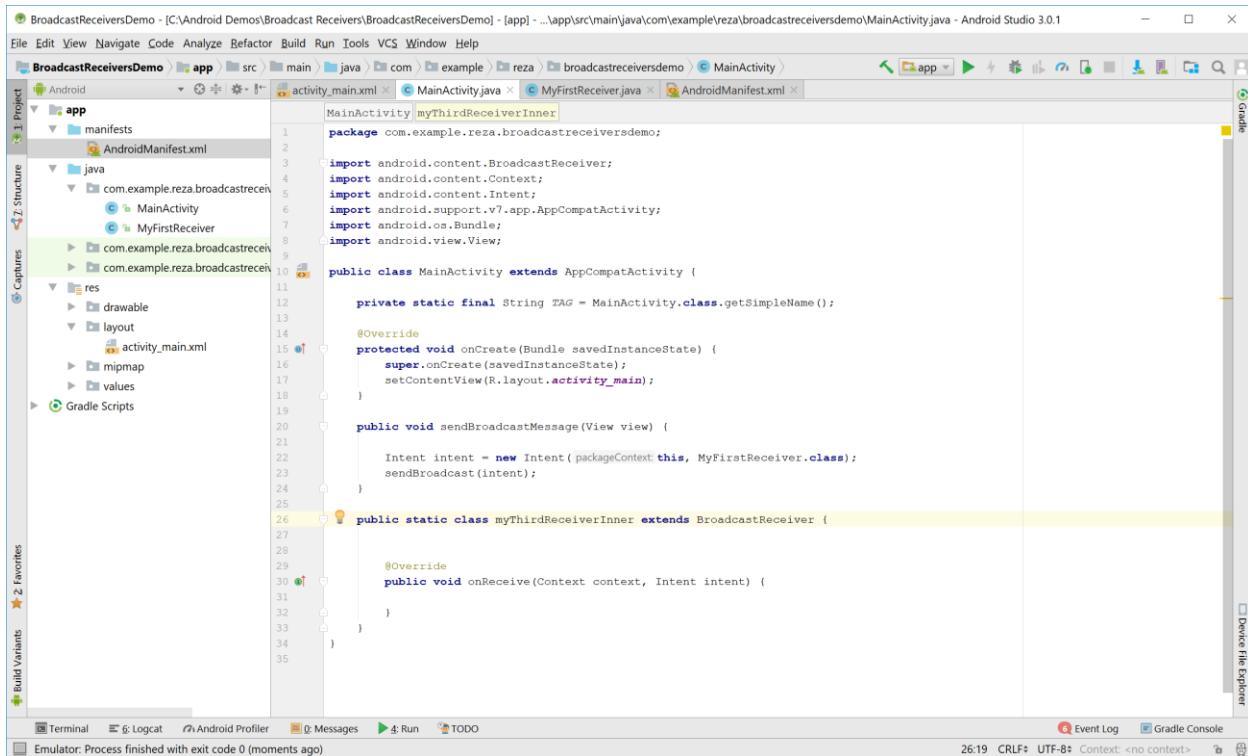
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MyFirstReceiver" />
        <receiver android:name=".MainActivity$MyThirdReceiverInner" />
    </application>
</manifest>

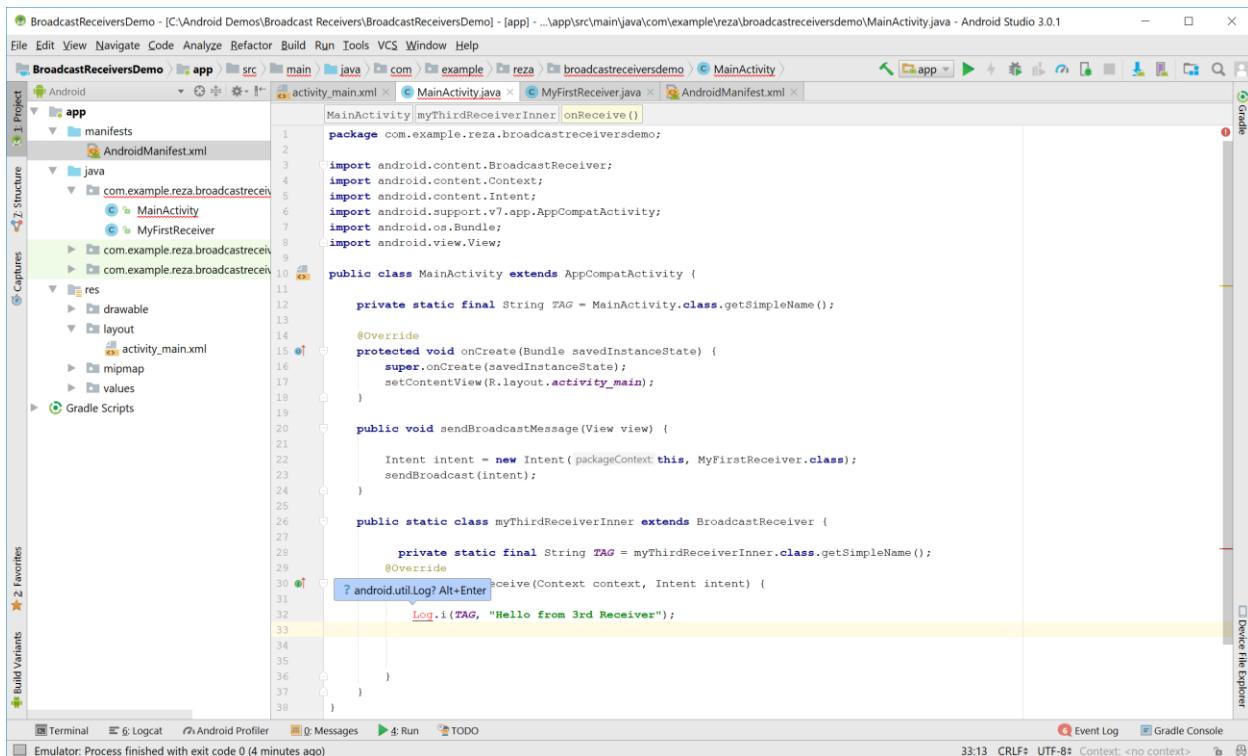
```

In the main activity use the static keyword for the inner class as shown:



```
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6 import android.support.v7.app.AppCompatActivity;
7 import android.os.Bundle;
8 import android.view.View;
9
10 public class MainActivity extends AppCompatActivity {
11
12     private static final String TAG = MainActivity.class.getSimpleName();
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18     }
19
20     public void sendBroadcastMessage(View view) {
21
22         Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
23         sendBroadcast(intent);
24     }
25
26     public static class myThirdReceiverInner extends BroadcastReceiver {
27
28
29         @Override
30         public void onReceive(Context context, Intent intent) {
31
32
33     }
34 }
35 }
```

Now you need to implement the onReceive() method of this inner class as shown:



```
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6 import android.support.v7.app.AppCompatActivity;
7 import android.os.Bundle;
8 import android.view.View;
9
10 public class MainActivity extends AppCompatActivity {
11
12     private static final String TAG = MainActivity.class.getSimpleName();
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18     }
19
20     public void sendBroadcastMessage(View view) {
21
22         Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
23         sendBroadcast(intent);
24     }
25
26     public static class myThirdReceiverInner extends BroadcastReceiver {
27
28         private static final String TAG = myThirdReceiverInner.class.getSimpleName();
29
30         @Override
31         public void onReceive(Context context, Intent intent) {
32
33             Log.i(TAG, "Hello from 3rd Receiver");
34
35         }
36     }
37 }
38 }
```

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo > app > src > main > java > com > example > reza > broadcastreceiversdemo > MainActivity >
activity_main.xml < MainActivity.java < MyFirstReceiver.java < AndroidManifest.xml

MainActivity myThirdReceiverInner
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }

    public static class myThirdReceiverInner extends BroadcastReceiver {

        private static final String TAG = myThirdReceiverInner.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg: "Hello from 3rd Receiver");
            Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
        }
    }
}

```

Also modify the activiy\_main.xml and add a new button for our new receiver:

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\res\\layout\\activity_main.xml - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo > app > src > main > res > layout > activity_main.xml >
activity_main.xml < MainActivity.java < MyFirstReceiver.java < AndroidManifest.xml

<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.reza.broadcastreceiversdemo.MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:onClick="sendBroadcastMessage"
        android:text="Broadcast Now"/>

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="87dp"
        android:onClick="broadcastToInnerReceiver"
        android:text="To Inner Receiver"/>

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\res\\layout\\activity\_main.xml - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

BroadcastReceiversDemo > app > src > main > res > layout > activity\_main.xml >

activity\_main.xml < MainActivity.java < MyFirstReceiver.java < AndroidManifest.xml

Project Z. Structure Captures Build Variants Favorites

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout>
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:onClick="sendBroadcastMessage"
        android:text="Broadcast Now"/>
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="87dp"
        android:onClick="broadcastToInnerReceiver"
        android:text="To Inner Receiver" />
</RelativeLayout>

```

Corresponding method handler 'public void broadcastToInnerReceiver(android.view.View)' not found

Event Log Gradle Console

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

BroadcastReceiversDemo > app > src > java > com > example > reza > broadcastreceiversdemo > MainActivity.java >

MainActivity broadcastToInnerReceiver()

```

public class MainActivity extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }

    public void broadcastToInnerReceiver(View view) {
    }

    public static class myThirdReceiverInner extends BroadcastReceiver {
        private static final String TAG = myThirdReceiverInner.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg: "Hello from 3rd Receiver");
            Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
        }
    }
}

```

Emulator: Process finished with exit code 0 (10 minutes ago)

Event Log Gradle Console

Now complete the code for broadcastToInnnerReceiver() method as shown:

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1

public class MainActivity extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }

    public void broadcastToInnnerReceiver(View view) {
        Intent intent = new Intent(getApplicationContext(), myThirdReceiverInner.class);
        sendBroadcast(intent);
    }

    public static class myThirdReceiverInner extends BroadcastReceiver {
        private static final String TAG = myThirdReceiverInner.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg: "Hello from 3rd Receiver");
            Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
        }
    }
}

```

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1

public class MainActivity extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
        sendBroadcast(intent);
    }

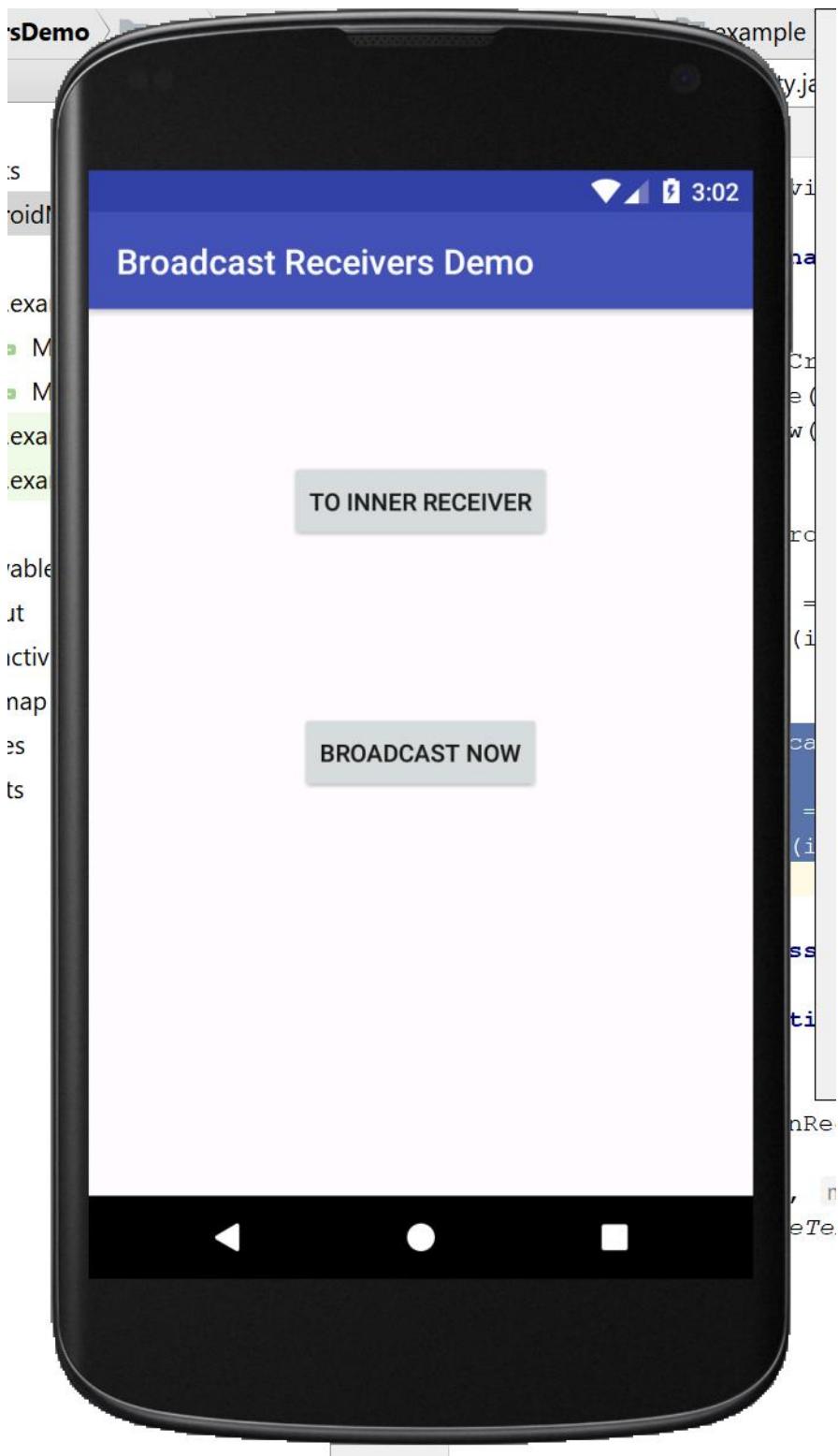
    public void broadcastToInnnerReceiver(View view) {
        Intent intent = new Intent(getApplicationContext(), myThirdReceiverInner.class);
        sendBroadcast(intent);
    }

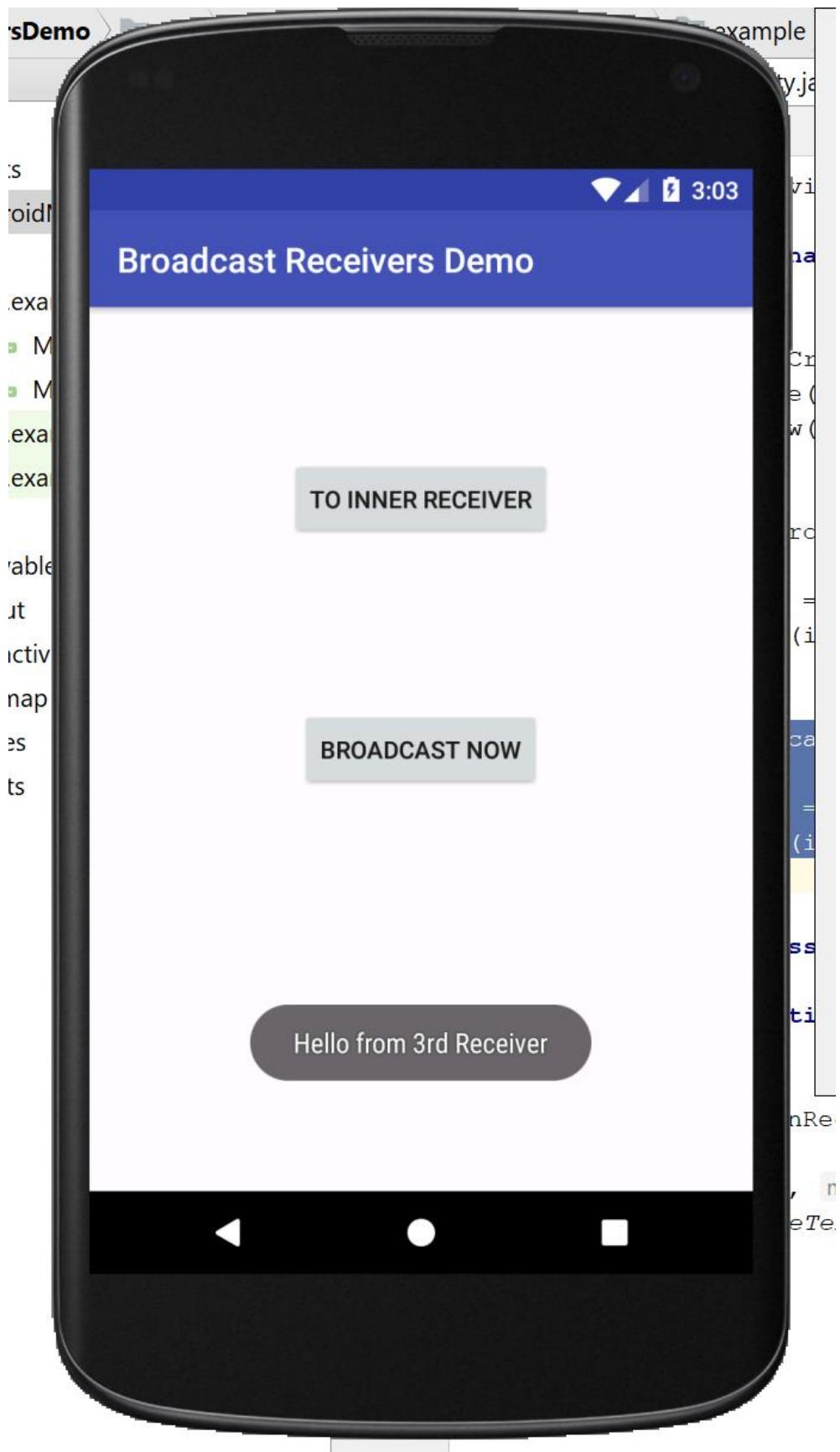
    public static class myThirdReceiverInner extends BroadcastReceiver {
        private static final String TAG = myThirdReceiverInner.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg: "Hello from 3rd Receiver");
            Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
        }
    }
}

```

Now run the application to test your new receiver:



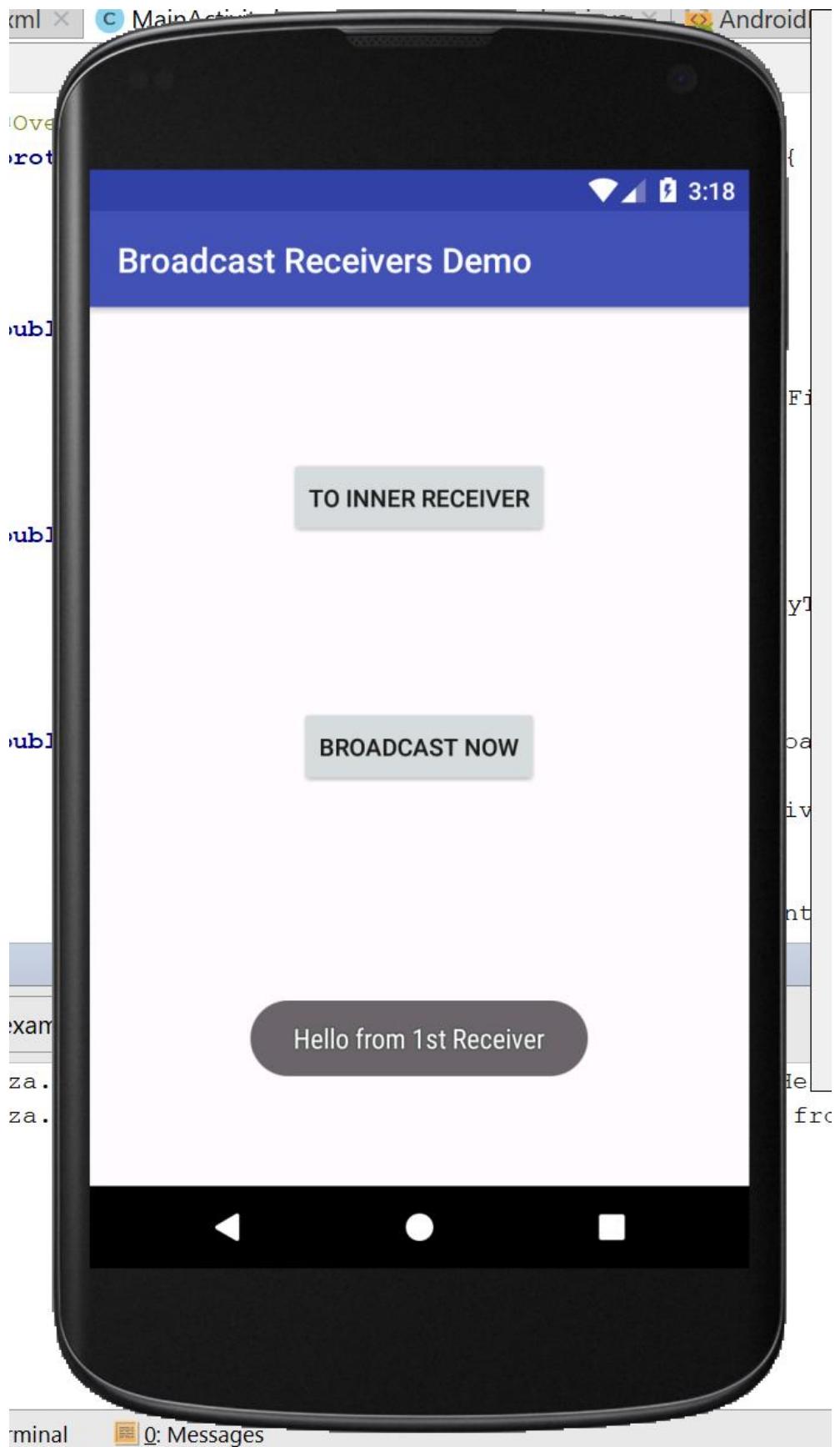


The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays `MainActivity.java` containing methods for sending broadcast messages and receiving them via an inner class. The Logcat tab shows a log entry from an emulator indicating a message received by a third receiver.

```
16    @Override
17    protected void onCreate(Bundle savedInstanceState) {
18        super.onCreate(savedInstanceState);
19        setContentView(R.layout.activity_main);
20    }
21
22    public void sendBroadcastMessage(View view) {
23
24        Intent intent = new Intent(getApplicationContext()>this, MyFirstReceiver.class);
25        sendBroadcast(intent);
26    }
27
28    public void broadcastToInnerReceiver(View view) {
29
30        Intent intent = new Intent(getApplicationContext()>this, myThirdReceiverInner.class);
31        sendBroadcast(intent);
32    }
33
34    public static class myThirdReceiverInner extends BroadcastReceiver {
35
36        private static final String TAG = myThirdReceiverInner.class.getSimpleName();
37
38        @Override
39        public void onReceive(Context context, Intent intent) {
```

Logcat output:

```
03-24 03:17:22.765 4569-4569/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
```



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays `MainActivity.java` containing methods for sending broadcast messages and receiving them via inner classes. The Logcat tab shows log entries from an Emulator Nexus\_4 API\_25 device, indicating successful broadcast delivery to both a third-party receiver and an inner receiver.

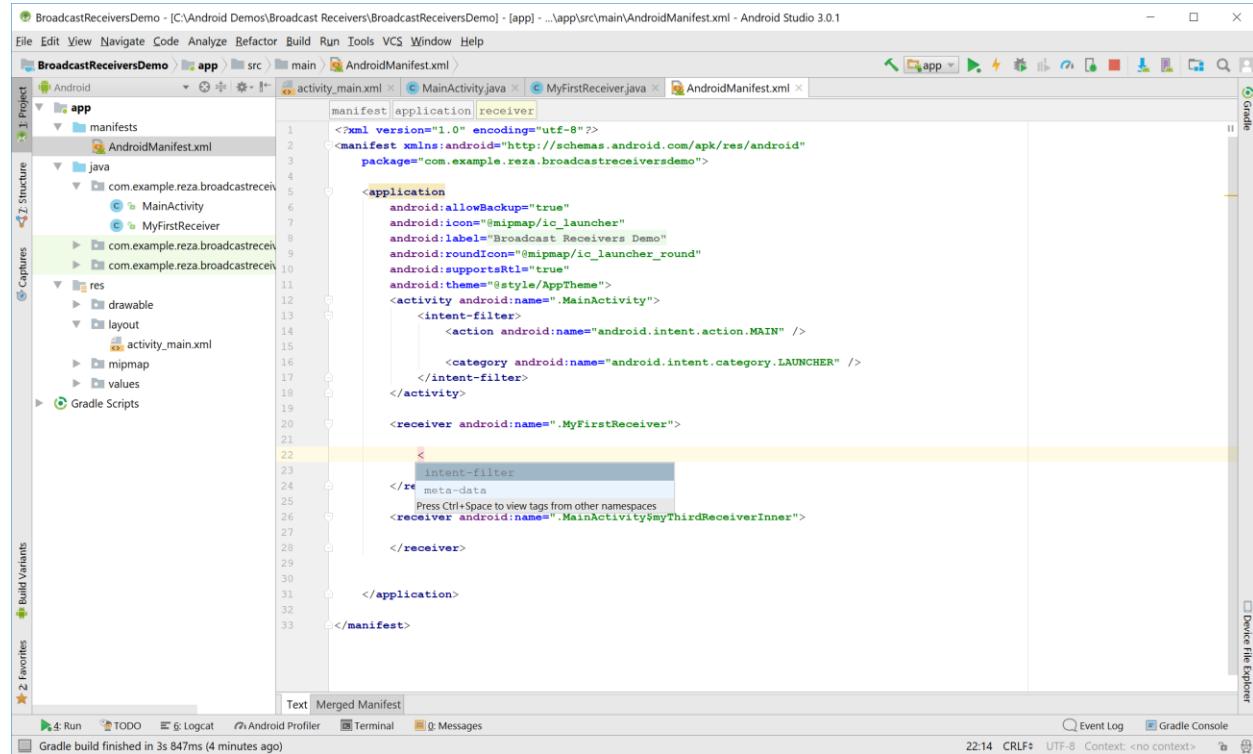
```
16    @Override
17    protected void onCreate(Bundle savedInstanceState) {
18        super.onCreate(savedInstanceState);
19        setContentView(R.layout.activity_main);
20    }
21
22    public void sendBroadcastMessage(View view) {
23
24        Intent intent = new Intent(getApplicationContext(), MyFirstReceiver.class);
25        sendBroadcast(intent);
26    }
27
28    public void broadcastToInnerReceiver(View view) {
29
30        Intent intent = new Intent(getApplicationContext(), myThirdReceiverInner.class);
31        sendBroadcast(intent);
32    }
33
34    public static class myThirdReceiverInner extends BroadcastReceiver {
35
36        private static final String TAG = myThirdReceiverInner.class.getSimpleName();
37
38        @Override
39        public void onReceive(Context context, Intent intent) {
```

Logcat Output:

```
Emulator Nexus_4 API_25 Android 7.1.1, API 25 com.example.reza.broadcastreceiversdemo (4569)
03-24 03:17:22.765 4569-4569/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
03-24 03:18:08.696 4569-4569/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
```

## Using Intent Filter with Custom Action

Modify your manifest file and define your intent filter as shown:

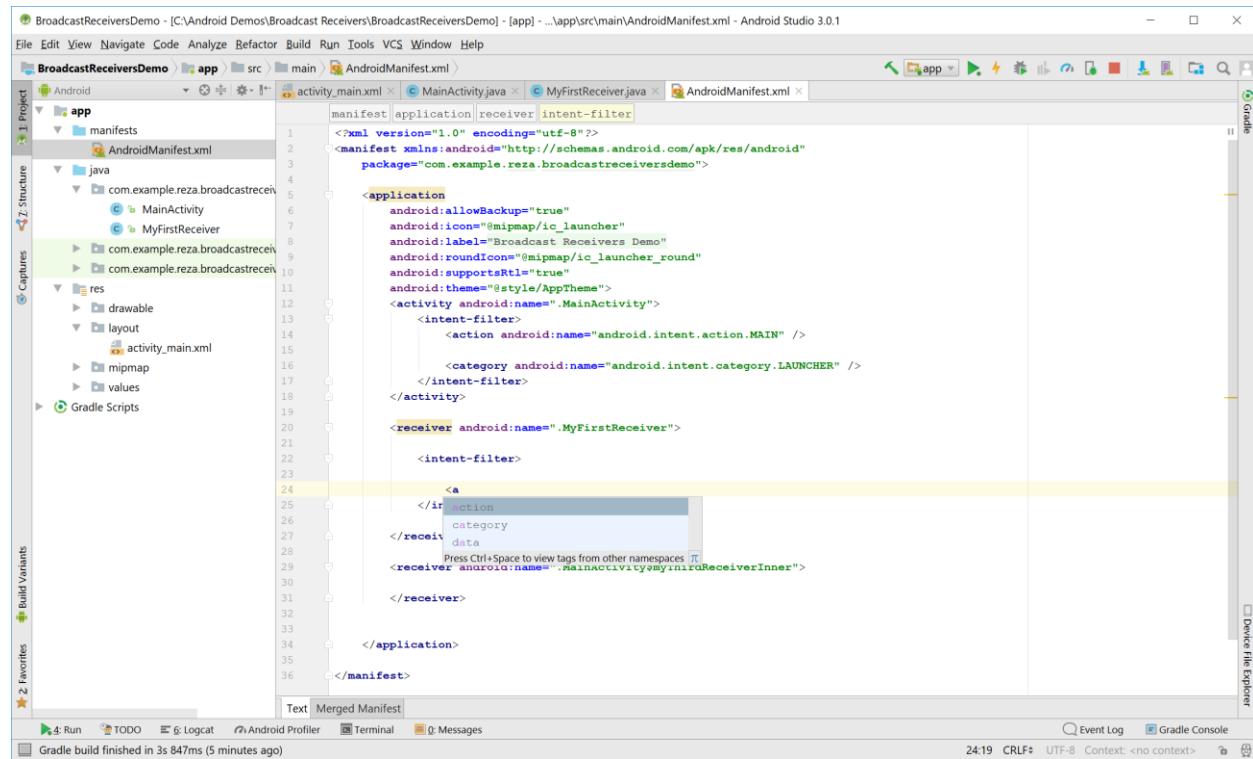


The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the `AndroidManifest.xml` file. A specific section of the XML code is highlighted with a yellow background, showing the intent filter configuration for the receiver. The code snippet is as follows:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="com.example.reza.broadcastreceiversdemo.myfirstreceiverinner" />
                <category android:name="com.example.reza.broadcastreceiversdemo.myfirstreceiverinner" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```



The screenshot shows the same Android Studio interface with the `AndroidManifest.xml` file open. This time, the intent filter for the receiver is defined with a custom action name, `com.example.reza.broadcastreceiversdemo.myfirstreceiverinner`. The code snippet is as follows:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="com.example.reza.broadcastreceiversdemo.myfirstreceiverinner" />
                <category android:name="com.example.reza.broadcastreceiversdemo.myfirstreceiverinner" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...app\src\main\AndroidManifest.xml - Android Studio 3.0.1

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="node" />
            </intent-filter>
        </receiver>
        <receiver android:name=".Main" android:strict="rInner">
            <intent-filter>
                <action android:name="remove" />
            </intent-filter>
        </receiver>
        <receiver android:name=".replace" android:strict="rInInner">
            <intent-filter>
                <action android:name="strict" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...app\src\main\AndroidManifest.xml - Android Studio 3.0.1

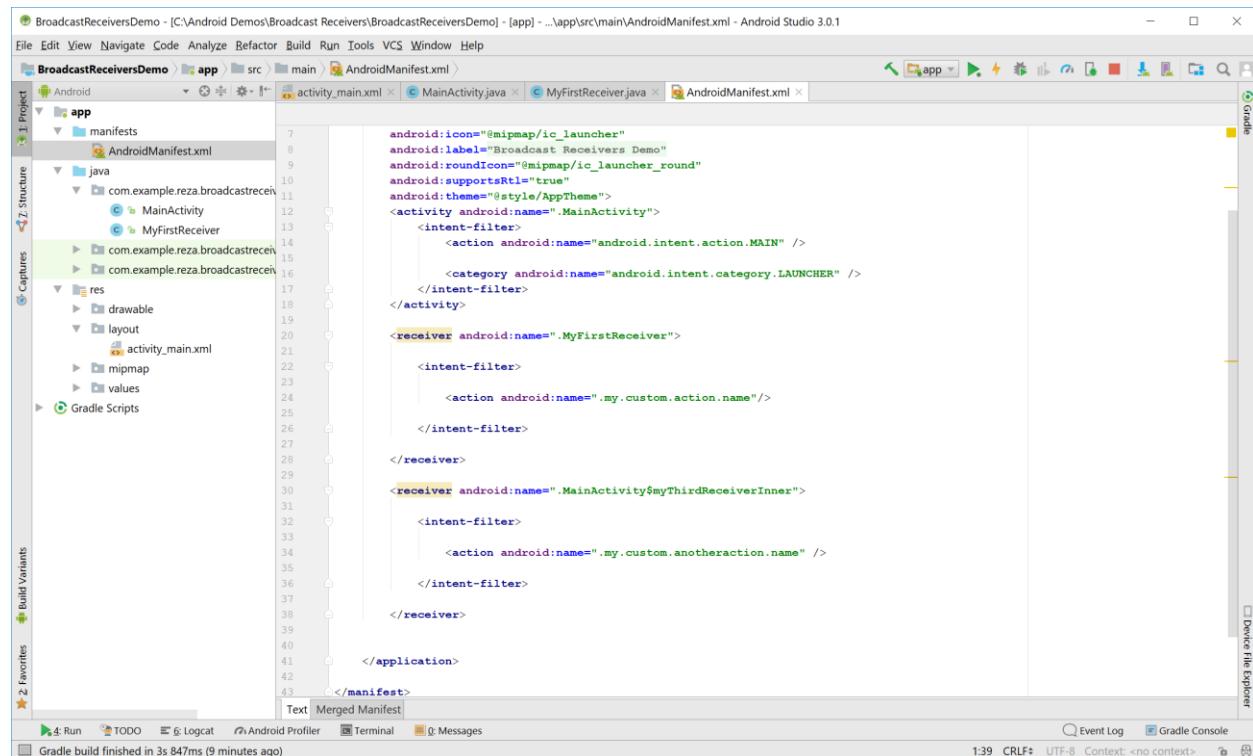
```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.reza.broadcastreceiversdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Broadcast Receivers Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="" />
            </intent-filter>
        </receiver>
        <receiver android:name=".Main" />
        <receiver android:name=".replace" />
    </application>
</manifest>
```

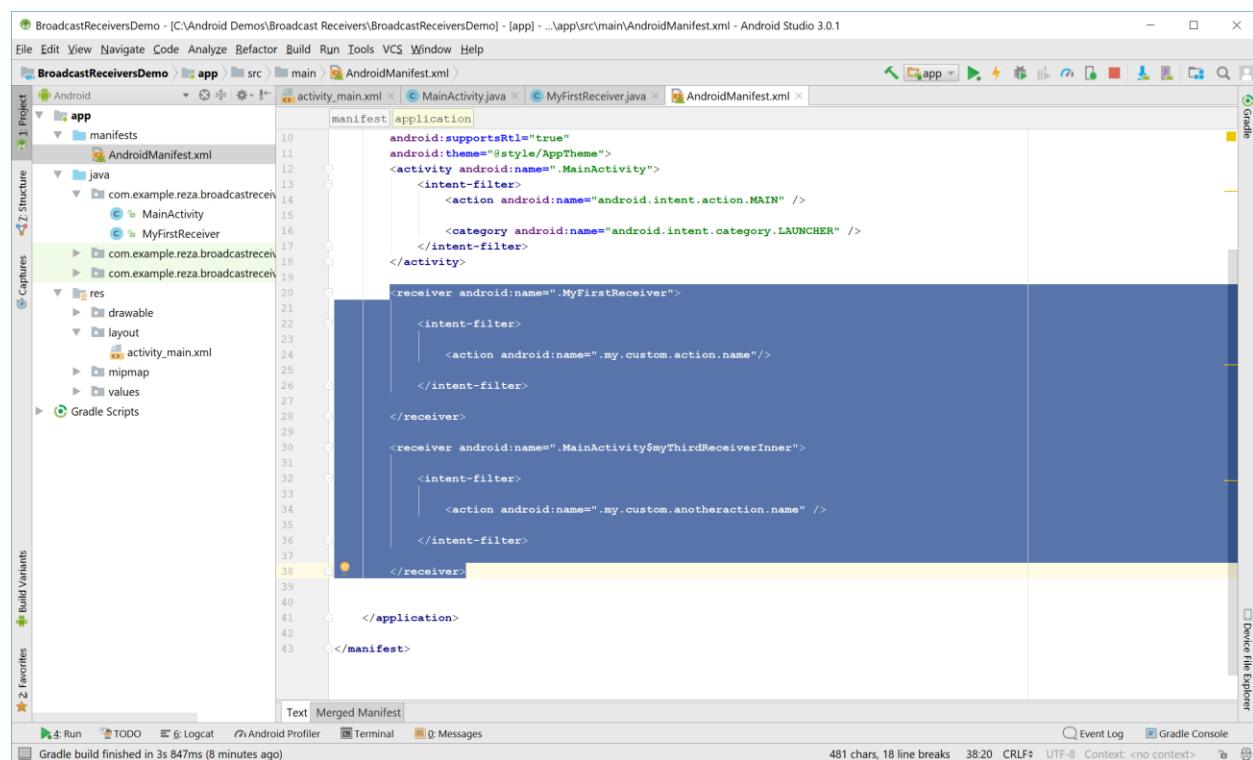
Here is your completed manifest file:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The manifest file is selected in the Project structure. The code editor displays the following XML configuration:

```
1 BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\AndroidManifest.xml - Android Studio 3.0.1
2 File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
3 BroadcastReceiversDemo > app > src > main > AndroidManifest.xml
4 activity_main.xml < MainActivity.java < MyFirstReceiver.java < AndroidManifest.xml
5
6
7 <manifest>
8   <application>
9     <activity android:name=".MainActivity">
10       <intent-filter>
11         <action android:name="android.intent.action.MAIN" />
12       </intent-filter>
13     </activity>
14
15     <receiver android:name=".MyFirstReceiver">
16       <intent-filter>
17         <action android:name=".my.custom.action.name" />
18       </intent-filter>
19     </receiver>
20
21     <receiver android:name=".MainActivity$MyThirdReceiverInner">
22       <intent-filter>
23         <action android:name=".my.custom.anotheraction.name" />
24       </intent-filter>
25     </receiver>
26
27   </application>
28
29 </manifest>
```

The code editor has tabs for 'Text' and 'Merged Manifest'. The status bar at the bottom shows 'Gradle build finished in 3s 847ms (9 minutes ago)'. The bottom right corner shows '1:39 CRLF: UTF-8 Context: <no context>'.



The screenshot shows the same Android Studio interface as the previous one, but with a yellow highlight on the receiver section of the manifest file. The code editor displays the same XML configuration as above. The status bar at the bottom shows 'Gradle build finished in 3s 847ms (8 minutes ago)'. The bottom right corner shows '481 chars, 18 line breaks 38:20 CRLF: UTF-8 Context: <no context>'.

With defining an intent-filter and an action tag, you can send a broadcast implicitly versus sending a broadcast explicitly. In our code before we sent a broadcast explicitly to our first receiver or to our third receiver. Now we can send a broadcast implicitly as shown. Modify the main activity java file as shown:

```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo MainActivity
activity_main.xml MainActivity MyFirstReceiver AndroidManifest.xml
Project Android app manifests AndroidManifest.xml
Structure com.example.reza.broadcastreceiversdemo MainActivity MyFirstReceiver
com.example.reza.broadcastreceiversdemo MyFirstReceiver
com.example.reza.broadcastreceiversdemo MyFirstReceiver
res drawable layout activity_main.xml mipmap values
Gradle Scripts
Captures
Favorites
Build Variants
Event Log Gradle Console
Device File Explorer
32-69 CRLF: UTF-8: Context: <no context>
Gradle build finished in 3s 847ms (15 minutes ago)

```

```

MainActivity broadcastToInnerReceiver()
{
    import android.os.Bundle;
    import android.util.Log;
    import android.view.View;
    import android.widget.Toast;

    public class MainActivity extends AppCompatActivity {
        private static final String TAG = MainActivity.class.getSimpleName();

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void sendBroadcastMessage(View view) {
            // Intent intent = new Intent(this, MyFirstReceiver.class);
            Intent intent = new Intent("action\\.my\\.custom\\.action\\.name");
            sendBroadcast(intent);
        }

        public void broadcastToInnerReceiver(View view) {
            // Intent intent = new Intent (this, myThirdReceiverInner.class);
            Intent intent = new Intent("action\\.my\\.custom\\.anotheraction\\.name");
            sendBroadcast(intent);
        }

        public static class myThirdReceiverInner extends BroadcastReceiver {
            private static final String TAG = myThirdReceiverInner.class.getSimpleName();

            @Override
            public void onReceive(Context context, Intent intent) {
                Log.i(TAG, msg: "Hello from 3rd Receiver");
                Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
            }
        }
    }
}

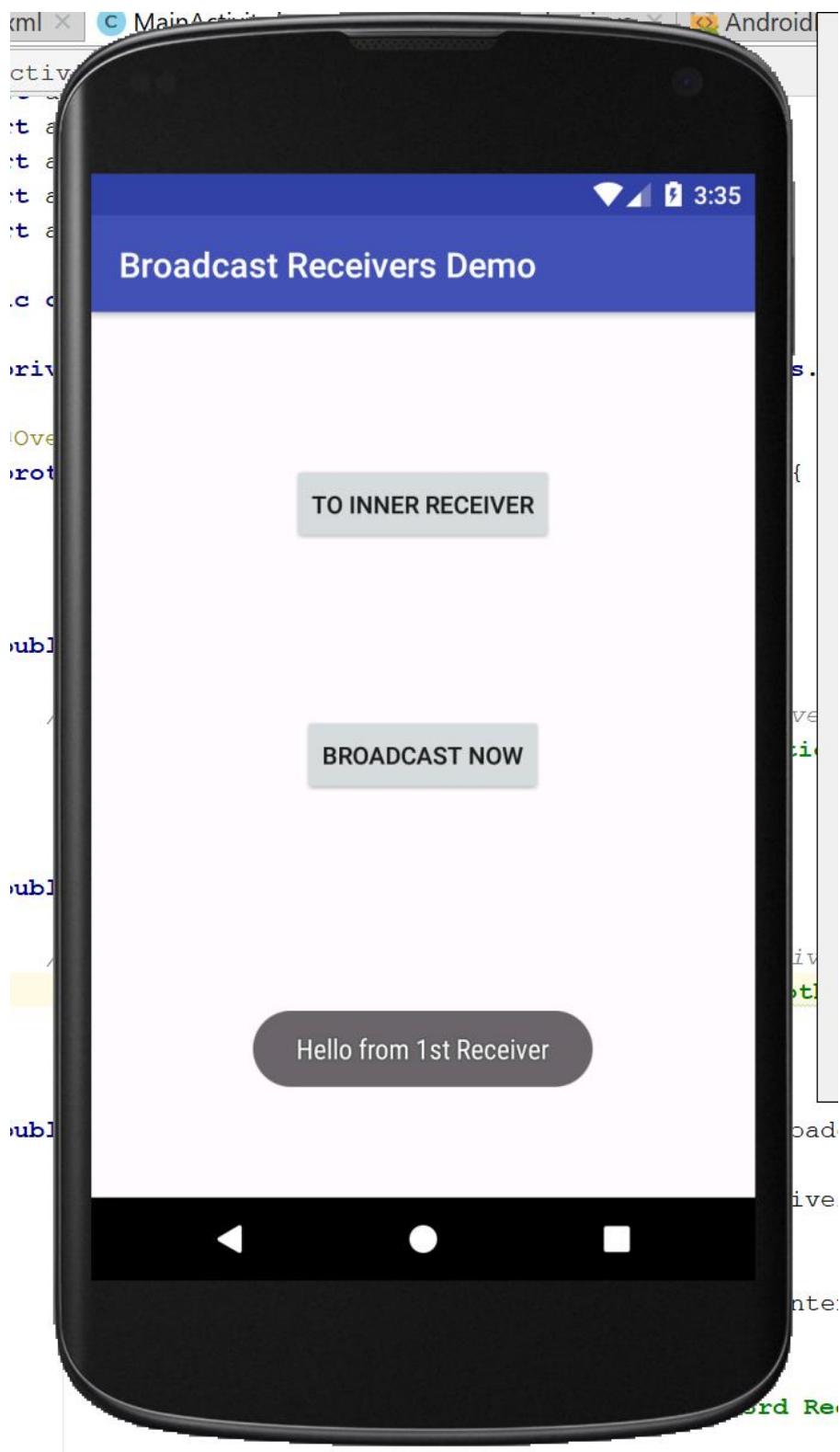
```

In our code we have commented out lines, 24 and 31 and added lines 25 and 32 which create our intents based on the action strings we specified in our manifest file.

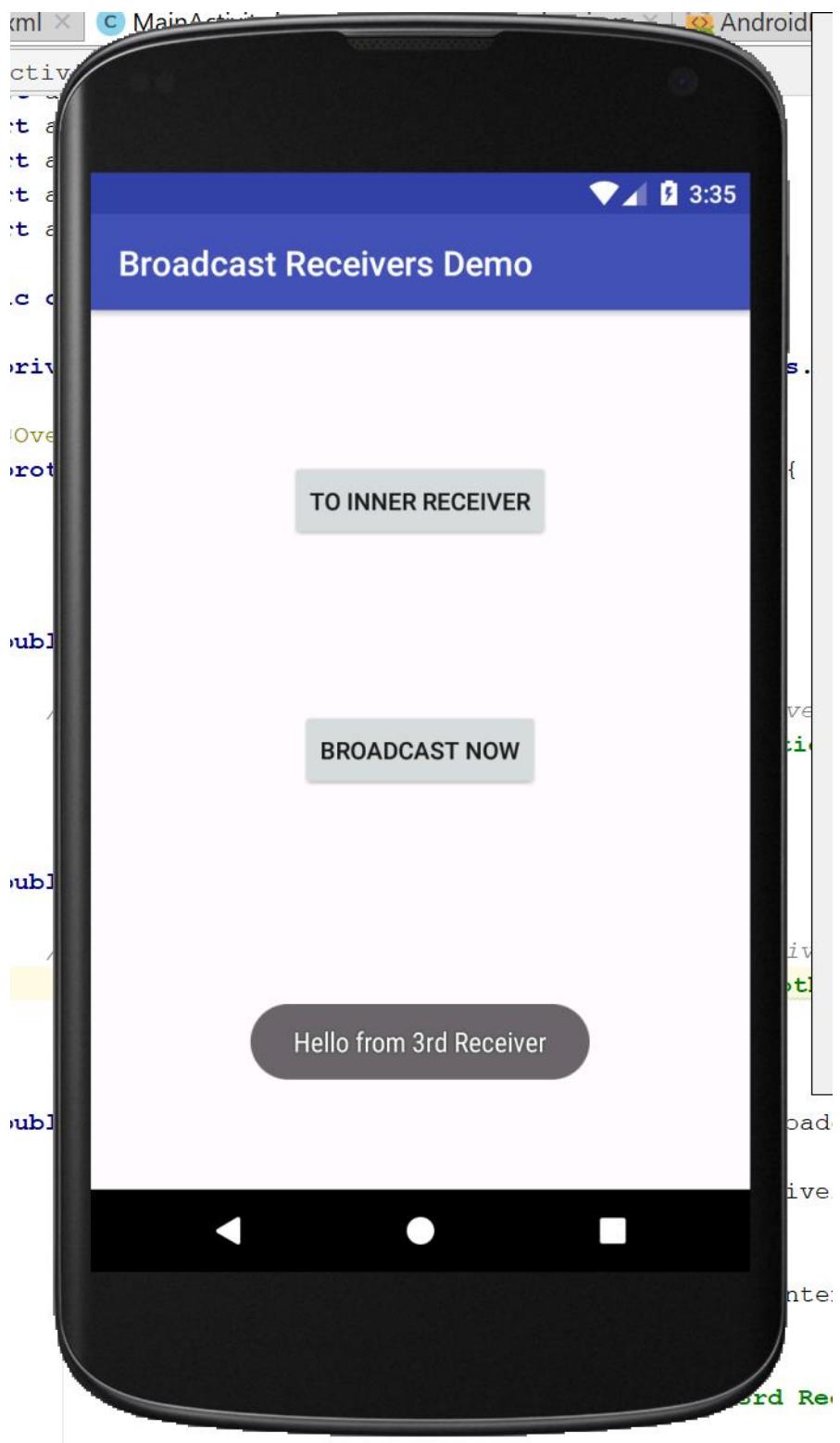
Now run the application:



Click “Broadcast Now” button:



Click “To Inner Receiver” button:



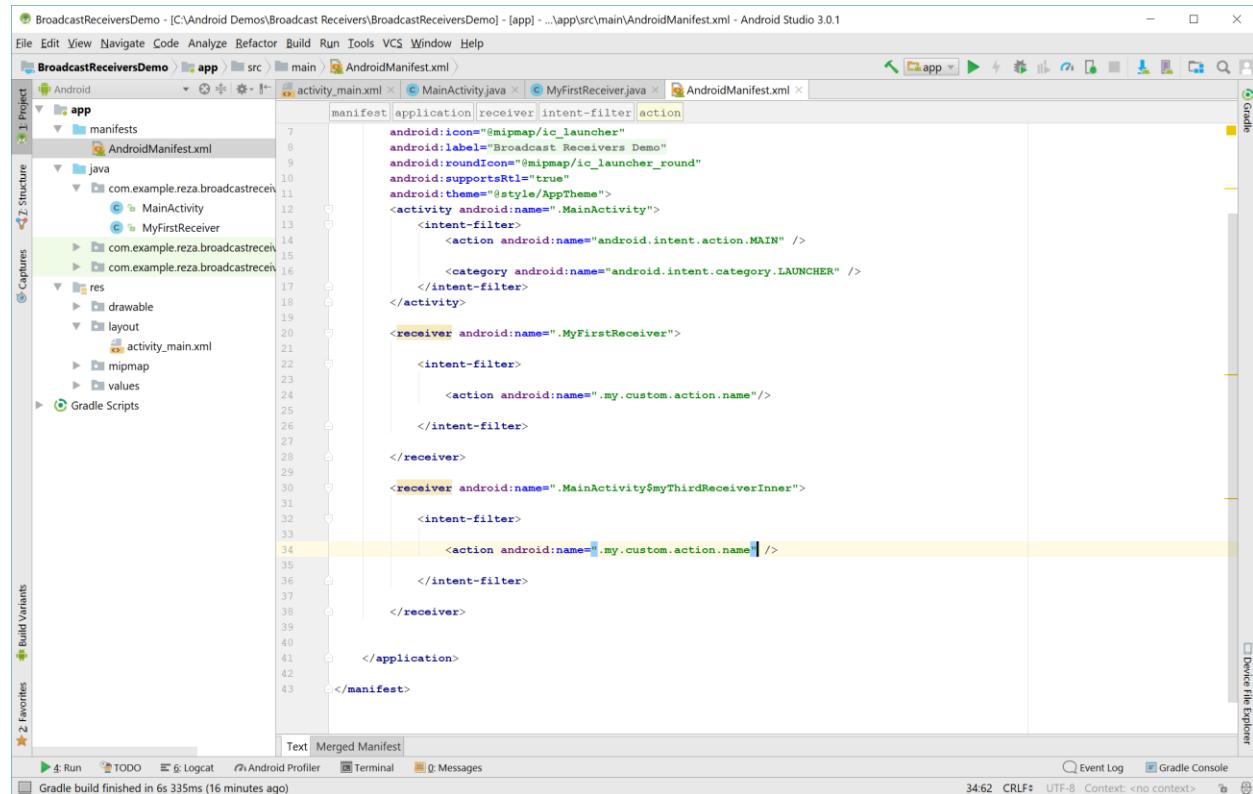
So the application works as before and delivers the broadcast to those Broadcast Receivers that are registered to those actions. So, for “my.custom.action.name” we have “MyFirstReceiver” registered and for “my.custom.anotheraction.name” we have our “myThirdReceiverInner” registered inside our manifest file.

Note that you can define as many receivers as you want inside the manifest file and then you should define the action for each of those receivers.

You can also have multiple receivers with the same action name which is the next topic.

## Multiple Receivers with the Same Action Name

Modify the manifest file and choose the same action for both receivers as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'AndroidManifest.xml' file is selected in the editor. The code highlights two receiver declarations, both using the action `my.custom.action.name`:

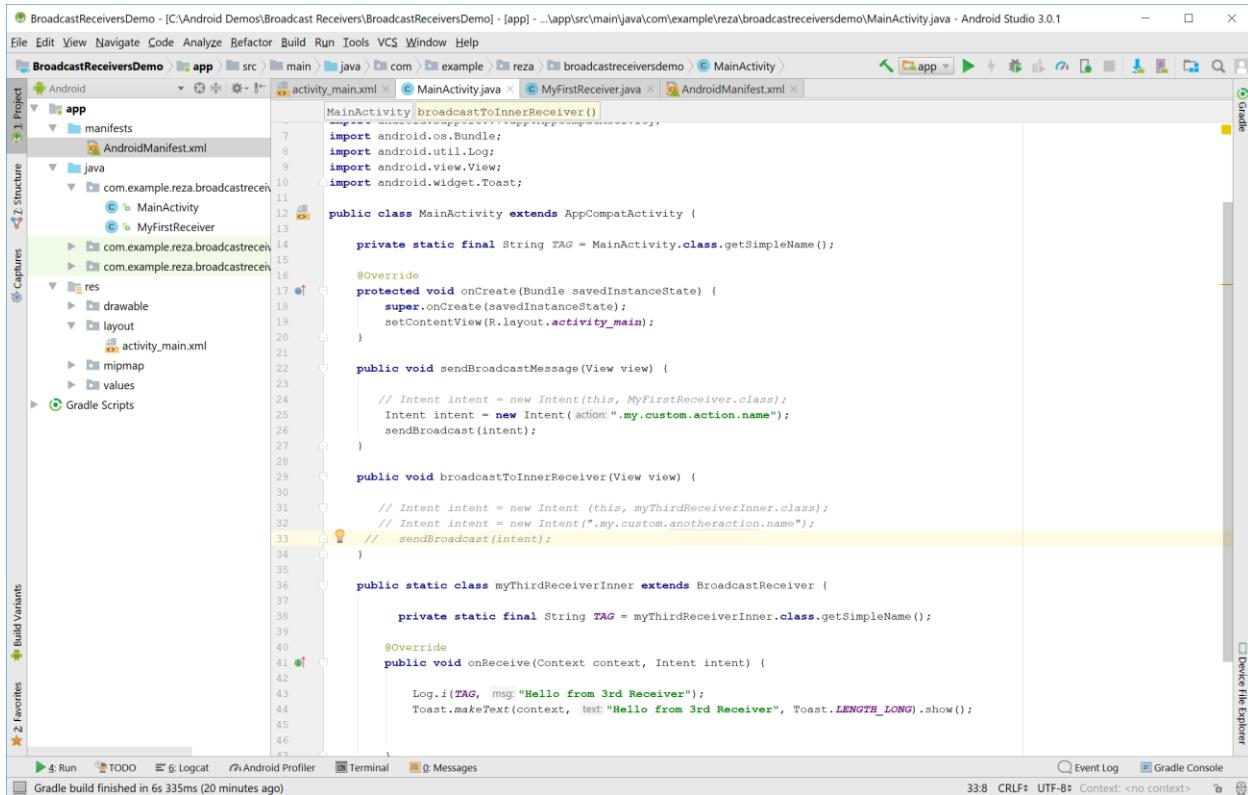
```
manifest application receiver intent-filter action
    android:icon="@mipmap/ic_launcher"
    android:label="Broadcast Receivers Demo"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <receiver android:name=".MyFirstReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name"/>
        </intent-filter>
    </receiver>

    <receiver android:name=".MainActivity$MyThirdReceiverInner">
        <intent-filter>
            <action android:name=".my.custom.action.name" />
        </intent-filter>
    </receiver>
</application>
</manifest>
```

The bottom status bar indicates a successful Gradle build.

Now go back inside the main activity and modify the code as shown (I have commented out lines 32 and 33, since our sendBroadcastMessage() method defined in line 22 is sufficient now):



```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MainActivity.java - Android Studio 3.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo MainActivity
activity_main.xml MainActivity.java MyFirstReceiver.java AndroidManifest.xml
Project Z: Structure Captures Gradle
1. Project app manifests AndroidManifest.xml
2. Java com.example.reza.broadcastreceiversdemo
   - MainActivity
   - MyFirstReceiver
3. com.example.reza.broadcastreceiversdemo
4. res drawable activity_main.xml
5. layout activity_main.xml
6. mipmap
7. values
Gradle Scripts
Build Variants
Favorites
Device File Explorer
1. Run TODO Logcat Android Profiler Terminal Messages
Event Log Gradle Console
Gradle build finished in 6s 335ms (20 minutes ago)
33:8 CRLF+ UTF-8+ Context: <no context>

```

```

MainActivity.broadcastToInnerReceiver()
{
    ...
    import android.os.Bundle;
    import android.util.Log;
    import android.view.View;
    import android.widget.Toast;

    public class MainActivity extends AppCompatActivity {

        private static final String TAG = MainActivity.class.getSimpleName();

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void sendBroadcastMessage(View view) {
            // Intent intent = new Intent(this, MyFirstReceiver.class);
            Intent intent = new Intent("com.example.reza.broadcastreceiversdemo.action.custom.action.name");
            sendBroadcast(intent);
        }

        public void broadcastToInnerReceiver(View view) {
            // Intent intent = new Intent(this, myThirdReceiverInner.class);
            // Intent intent = new Intent("com.example.reza.broadcastreceiversdemo.action.custom.anotheraction.name");
            // sendBroadcast(intent);
        }

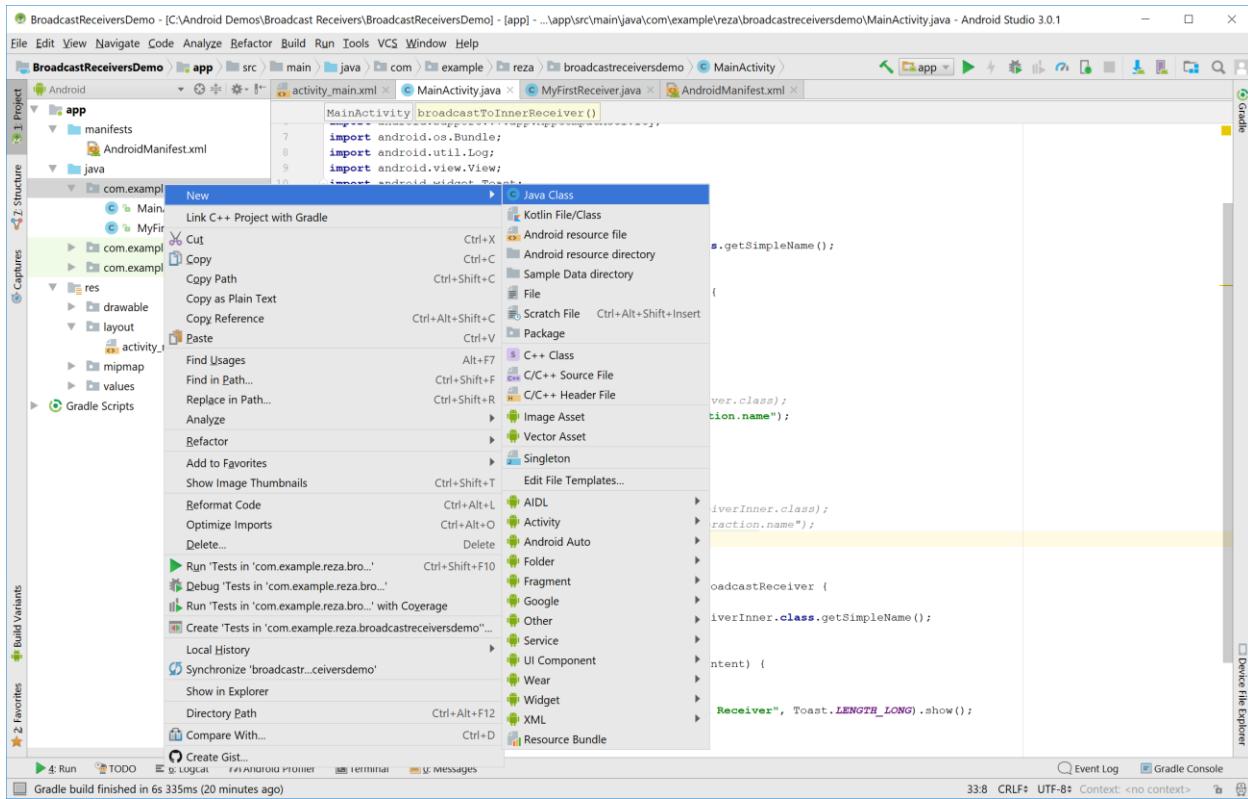
        public static class myThirdReceiverInner extends BroadcastReceiver {

            private static final String TAG = myThirdReceiverInner.class.getSimpleName();

            @Override
            public void onReceive(Context context, Intent intent) {
                Log.i(TAG, msg: "Hello from 3rd Receiver");
                Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
            }
        }
    }
}

```

Now create your second receiver as shown:



 Create New Class X

Name:

Kind:  Class ▼

Superclass:

Interface(s):

Package: com.example.reza.broadcastreceiversdemo

Visibility:  Public  Package Private

Modifiers:  None  Abstract  Final

Show Select Overrides Dialog

OK Cancel Help

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MySecondReceiver.java - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo C MySecondReceiver

activity\_main.xml C MainActivity.java C MySecondReceiver.java C MyFirstReceiver.java C AndroidManifest.xml

Project Z. Structure Captures Device File Explorer

app manifests AndroidManifest.xml

java com.example.reza.broadcastreceiversdemo MainActivity MyFirstReceiver MySecondReceiver

res drawable layout activity\_main.xml mipmap values

Gradle Scripts

MySecondReceiver

```
package com.example.reza.broadcastreceiversdemo;
```

```
/** * Created by Reza on 2018-03-23.
```

```
public class MySecondReceiver {
```

BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\java\\com\\example\\reza\\broadcastreceiversdemo\\MySecondReceiver.java - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

BroadcastReceiversDemo app src main java com example reza broadcastreceiversdemo C MySecondReceiver

activity\_main.xml C MainActivity.java C MySecondReceiver.java C MyFirstReceiver.java C AndroidManifest.xml

Project Z. Structure Captures Device File Explorer

app manifests AndroidManifest.xml

java com.example.reza.broadcastreceiversdemo MainActivity MyFirstReceiver MySecondReceiver

res drawable layout activity\_main.xml mipmap values

Gradle Scripts

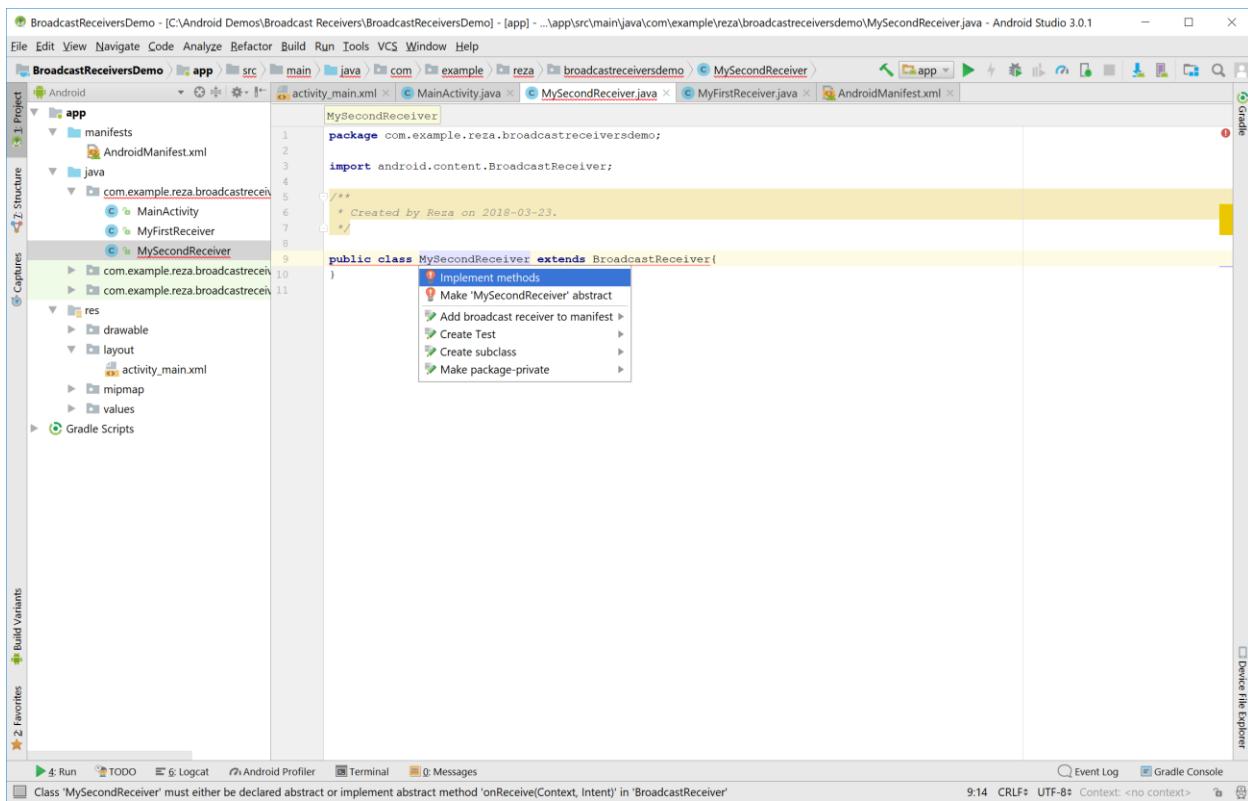
MySecondReceiver

```
package com.example.reza.broadcastreceiversdemo;
```

```
import android.content.BroadcastReceiver;
```

```
/** * Created by Reza on 2018-03-23.
```

```
public class MySecondReceiver extends BroadcastReceiver {
```





## Select Methods to Implement

▼ (c) android.content.BroadcastReceiver

(m) onReceive(context:Context, intent:Intent):void



Copy JavaDoc



Insert @Override

OK

Cancel

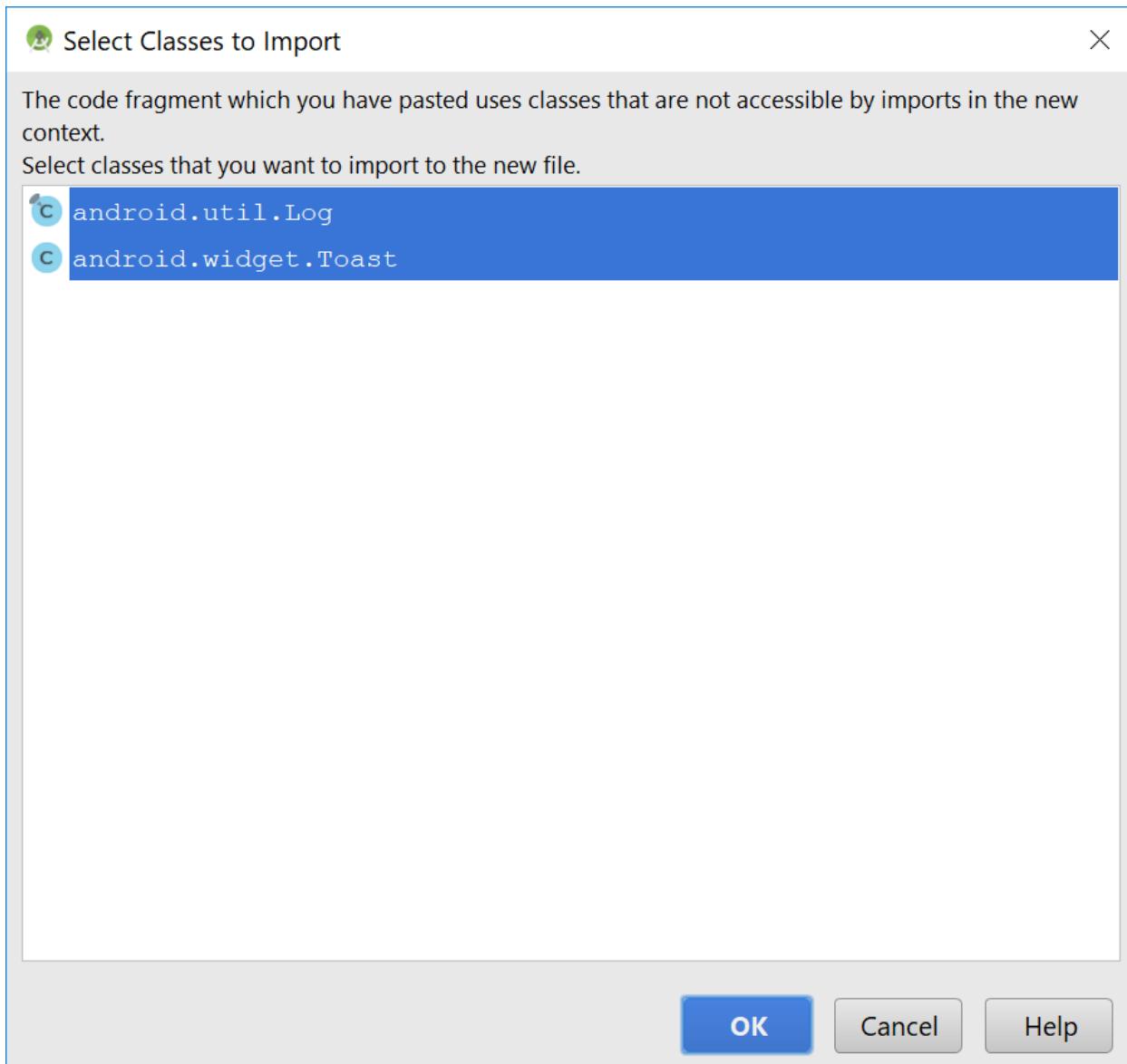
The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar displays the project structure under the "app" module. It includes the "manifests" folder containing "AndroidManifest.xml", the "java" folder containing "MainActivity.java", "MyFirstReceiver.java", and "MySecondReceiver.java", and the "res" folder containing "drawable", "layout", "mipmap", and "values".
- Code Editor:** The main window shows the code for "MySecondReceiver.java". The code defines a class "MySecondReceiver" that extends "BroadcastReceiver". It overrides the "onReceive" method to handle broadcast intents.

```
1 package com.example.reza.broadcastreceiversdemo;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6
7 /**
8 * Created by Reza on 2018-03-23.
9 */
10
11 public class MySecondReceiver extends BroadcastReceiver {
12     @Override
13     public void onReceive(Context context, Intent intent) {
14     }
15 }
```

- Toolbars and Status Bar:** The top bar contains standard Android Studio menu items like File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help. The status bar at the bottom shows "Gradle build finished in 6s 335ms (23 minutes ago)" and "14:9 CRLF+ UTF-8+ Context: <no context>".

Now add the string tag and the toast message as you did for the previous receivers as shown:



```
MySecondReceiver onReceive()
package com.example.reza.broadcastreceiversdemo;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

/**
 * Created by Reza on 2018-03-23.
 */

public class MySecondReceiver extends BroadcastReceiver {

    private static final String TAG = MySecondReceiver.class.getSimpleName();

    @Override
    public void onReceive(Context context, Intent intent) {

        Log.i(TAG, msg: "Hello from 2nd Receiver");
        Toast.makeText(context, text: "Hello from 2nd Receiver", Toast.LENGTH_LONG).show();
    }
}
```

Now declare your receiver inside android manifest file as you have done it before as shown:

```
<manifest>
    <application>
        <activity android:name=".MainActivity" />
        <receiver android:name=".MyFirstReceiver" />
        <receiver android:name=".MySecondReceiver" />
        <receiver android:name=".MyThirdReceiverInner" />
    </application>
</manifest>
```

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the `AndroidManifest.xml` file. A code completion dropdown is open at the end of the line `<receiver android:name=".MySecondReceiver">`, listing several receiver classes from the project: `MainActivity$MyThirdReceiverInner`, `MyFirstReceiver`, and `MySecondReceiver`. The `MySecondReceiver` entry is highlighted.

```
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <receiver android:name=".MyFirstReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name"/>
        </intent-filter>
    </receiver>

    <receiver android:name=".MySecondReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name"/>
        </intent-filter>
    </receiver>
```

The screenshot shows the same Android Studio interface with the `AndroidManifest.xml` file. The code completion dropdown is now open at the end of the line `<receiver android:name=".MainActivity$MyThirdReceiverInner">`, listing the same three receiver classes. The `MainActivity$MyThirdReceiverInner` entry is highlighted.

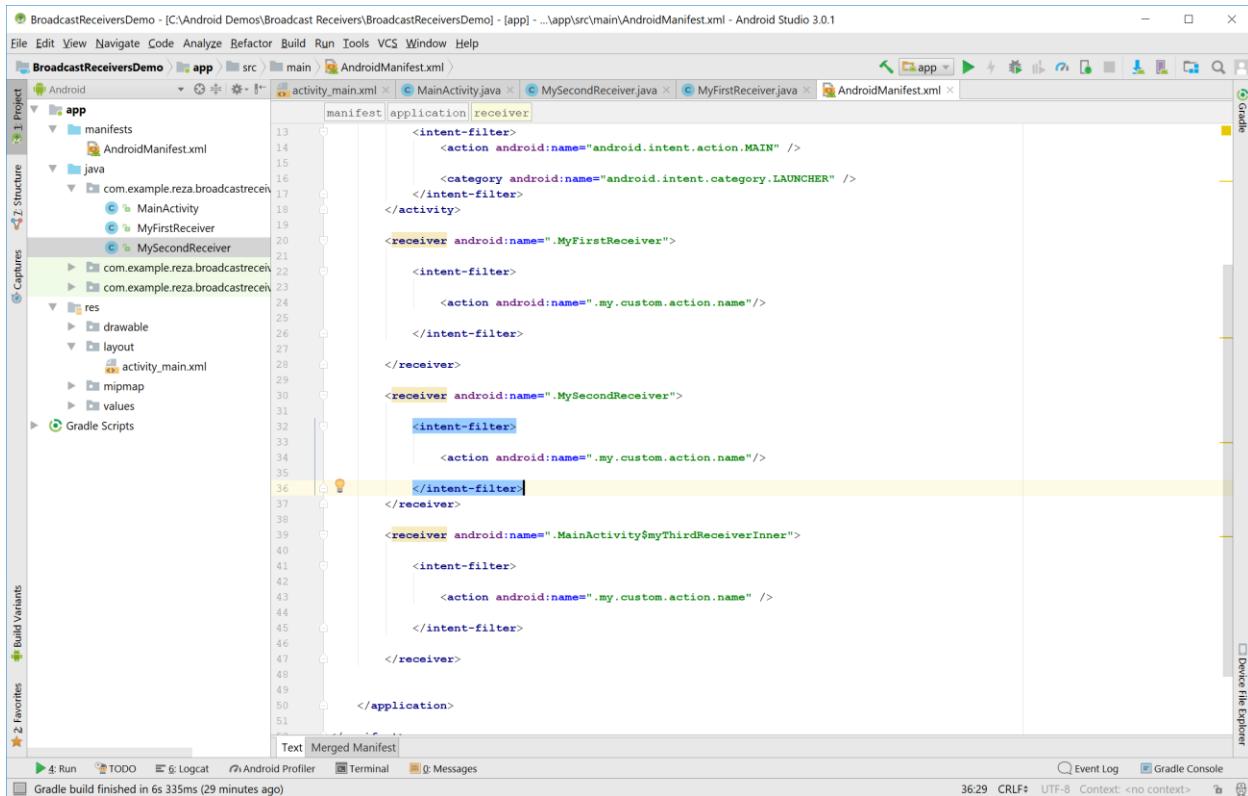
```
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <receiver android:name=".MyFirstReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name"/>
        </intent-filter>
    </receiver>

    <receiver android:name=".MySecondReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name"/>
        </intent-filter>
    </receiver>

    <receiver android:name=".MainActivity$MyThirdReceiverInner">
        <intent-filter>
            <action android:name=".my.custom.action.name" />
        </intent-filter>
    </receiver>
```

We are going to use the exact intent-filter and action as the other two receivers as shown so just copy the code from the other receiver and paste it:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'AndroidManifest.xml' file is selected in the editor. The manifest contains three receiver declarations, each with a specific intent filter and action:

```
<manifest>
    <application>
        <activity android:name=".MainActivity" />
        <category android:name="android.intent.category.LAUNCHER" />
    </activity>
    <receiver android:name=".MyFirstReceiver">
        <intent-filter>
            <action android:name="my.custom.action.name"/>
        </intent-filter>
    </receiver>
    <receiver android:name=".MySecondReceiver">
        <intent-filter>
            <action android:name="my.custom.action.name"/>
        </intent-filter>
    </receiver>
    <receiver android:name=".MainActivity$MyThirdReceiverInner">
        <intent-filter>
            <action android:name="my.custom.action.name" />
        </intent-filter>
    </receiver>
    </application>

```

Now you have three receivers and all of them have been registered with ".my.custom.action.name" action.

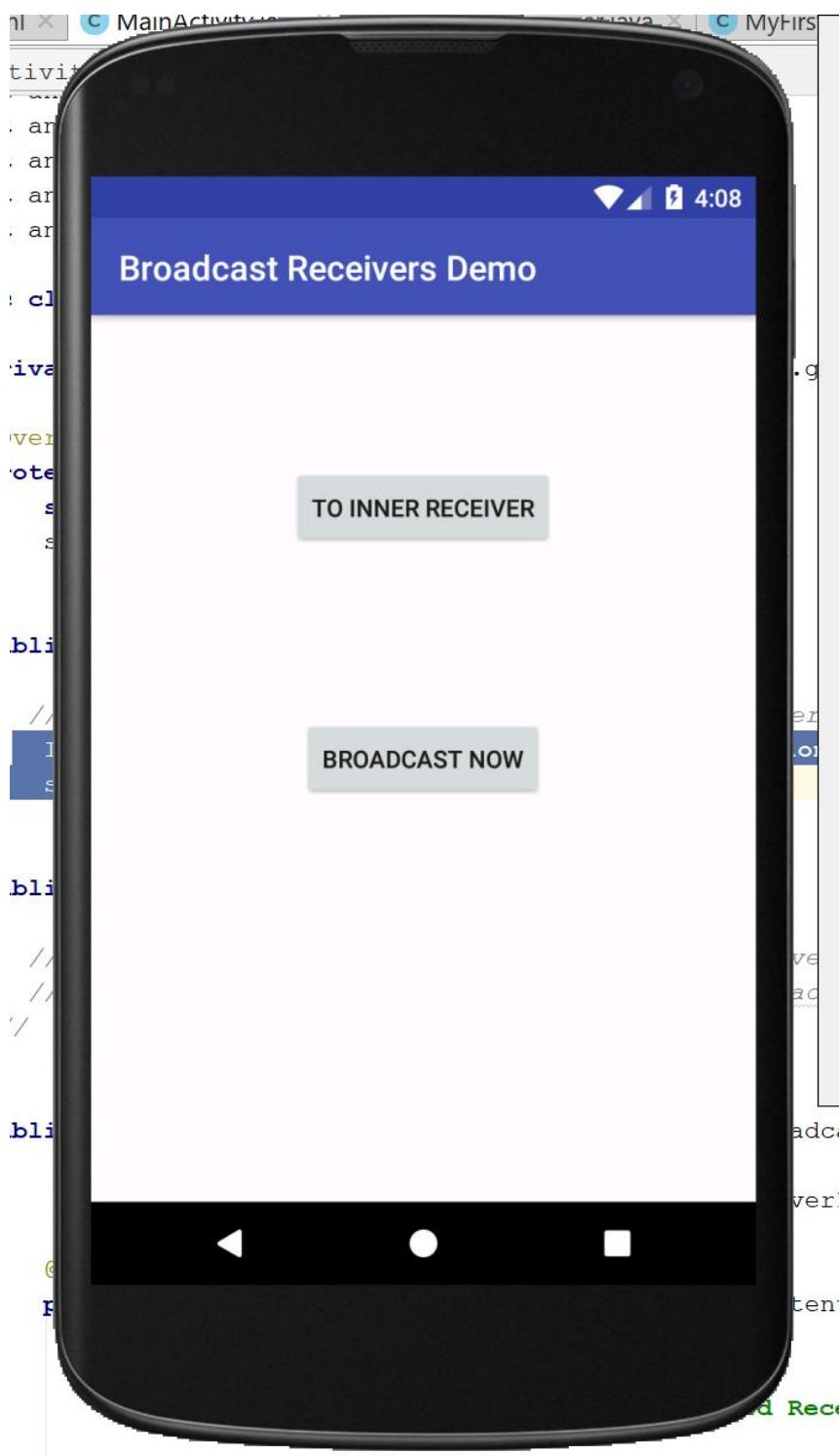
Now when the sendBroadcastMessage() method in the main activity is called and the two highlighted statements in it are executed, then Android operating system sends the broadcast to all of these three receivers:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the `MainActivity.java` file. Two specific lines of code are highlighted with orange circles:

```
private static final String TAG = MainActivity.class.getSimpleName();  
Intent intent = new Intent("com.example.reza.broadcastreceiversdemo.my.custom.action.name");
```

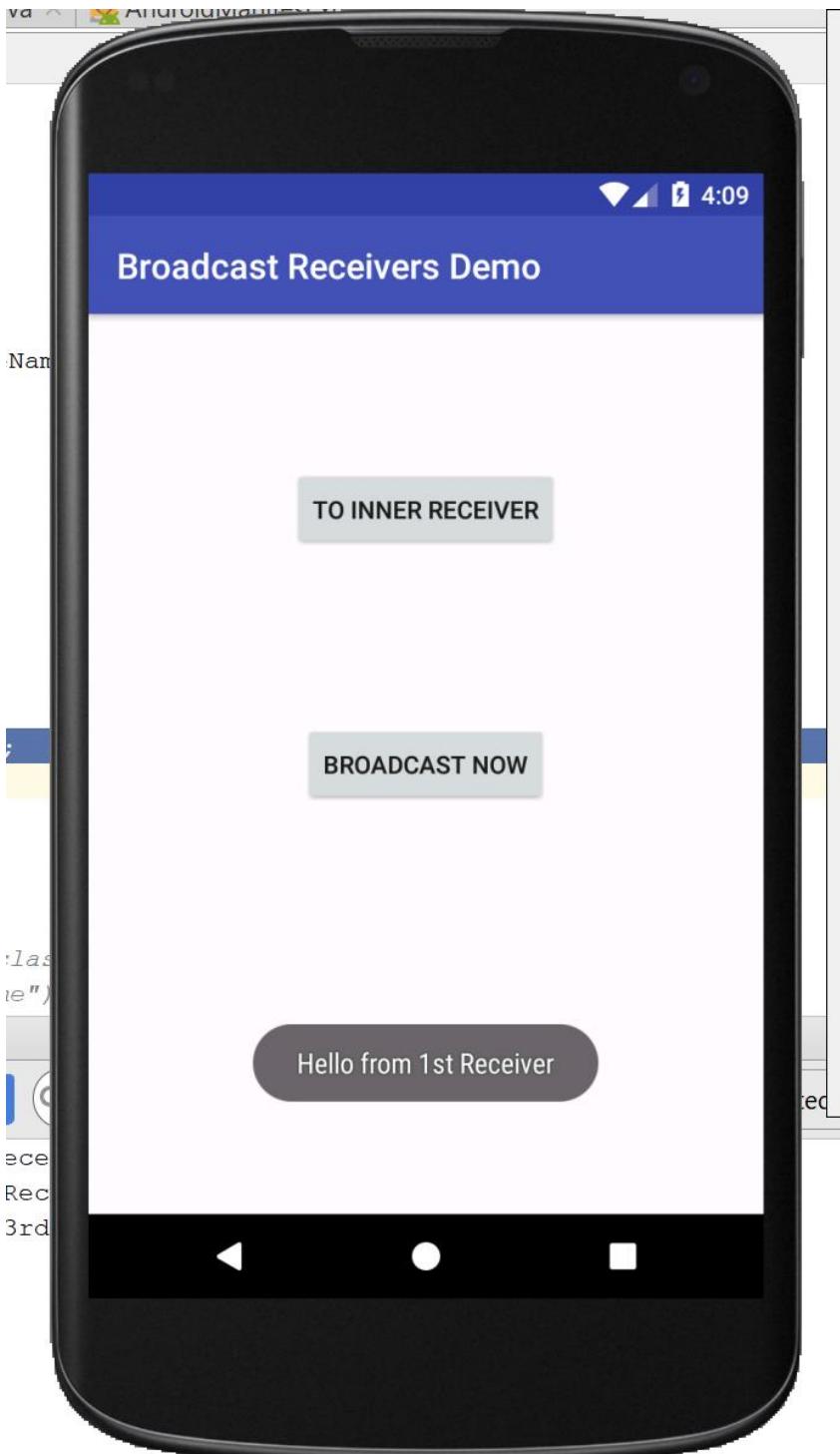
The first highlighted line is located at line 11. The second highlighted line is located at line 26. The code editor has a vertical scrollbar on the right side. The bottom status bar shows the build log: "Gradle build finished in 6s 335ms (32 minutes ago)".

Now run the app and test to see which receiver is called first:

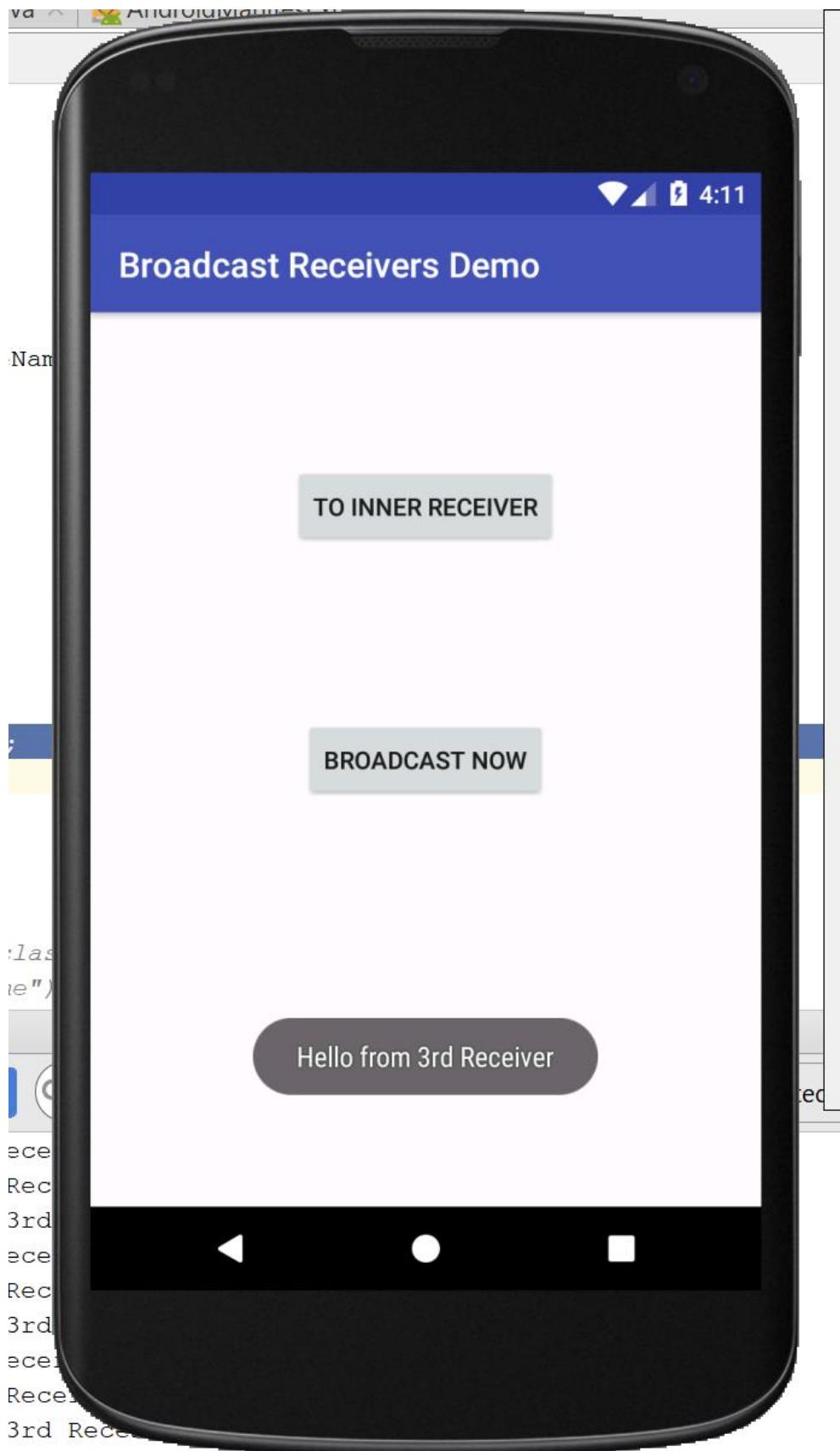


Note that now with clicking the “Broadcast Now” button, we will be sending the broadcast to all of our three receivers so what is the order that these guys are going to receive the broadcast?

Let's try it. Click the “Broadcast Now” button:







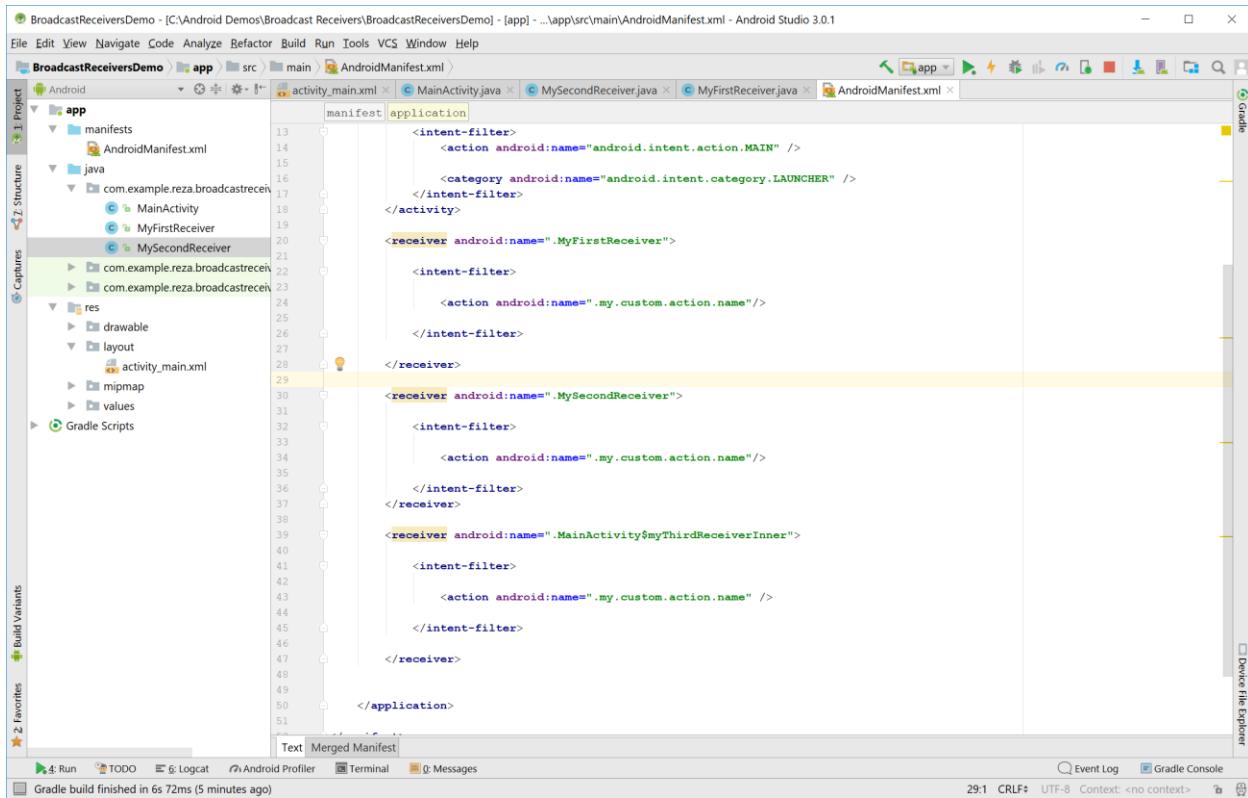
The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "BroadcastReceiversDemo". The `app` module contains the `AndroidManifest.xml`, `src/main/java/com/example/reza/broadcastreceiversdemo` directory which includes `MainActivity.java`, `MyFirstReceiver.java`, and `MySecondReceiver.java`, and the `layout` directory which contains `activity_main.xml`.
- Code Editor:** The `MainActivity.java` file is open. It contains Java code for sending broadcast messages. A specific line of code is highlighted: 

```
Intent intent = new Intent("my.custom.action.name");
```
- Logcat:** The Logcat tab shows the following log entries:

```
03-24 04:09:54.067 6350-6350/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
03-24 04:09:54.130 6350-6350/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 04:09:54.341 6350-6350/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
```
- Bottom Status Bar:** The status bar displays: "Gradle build finished in 6s 72ms (3 minutes ago)", "86 chars, 1 line break", "26:31 CRLF+", "UTF-8", "Context: <no context>".

So our receivers received the broadcast in the same order as they were defined in the manifest file as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The left sidebar shows the project structure with 'app' selected. In the main editor area, the 'AndroidManifest.xml' file is displayed. The code shows three receiver declarations within the application tag:

```
<application>
    <activity android:name=".MainActivity" />
    <category android:name="android.intent.category.LAUNCHER" />
</activity>

<receiver android:name=".MyFirstReceiver">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</receiver>

<receiver android:name=".MySecondReceiver">
    <intent-filter>
        <action android:name=".my.custom.action.name"/>
    </intent-filter>
</receiver>

<receiver android:name=".MainActivity$MyThirdReceiverInner">
    <intent-filter>
        <action android:name=".my.custom.action.name" />
    </intent-filter>
</receiver>

```

The 'Text' tab of the code editor is selected. At the bottom of the screen, the status bar shows 'Gradle build finished in 6s 72ms (5 minutes ago)'.

Let's change the order. Cut the code for the second receiver and put it before the first receiver as shown:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'AndroidManifest.xml' file is selected in the editor. The code is as follows:

```
manifest application receiver intent-filter
    android:supportRtl="true"
    android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MySecondReceiver">
            <intent-filter>
                <action android:name=".my.custom.action.name"/>
            </intent-filter>
        </receiver>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name=".my.custom.action.name"/>
            </intent-filter>
        </receiver>

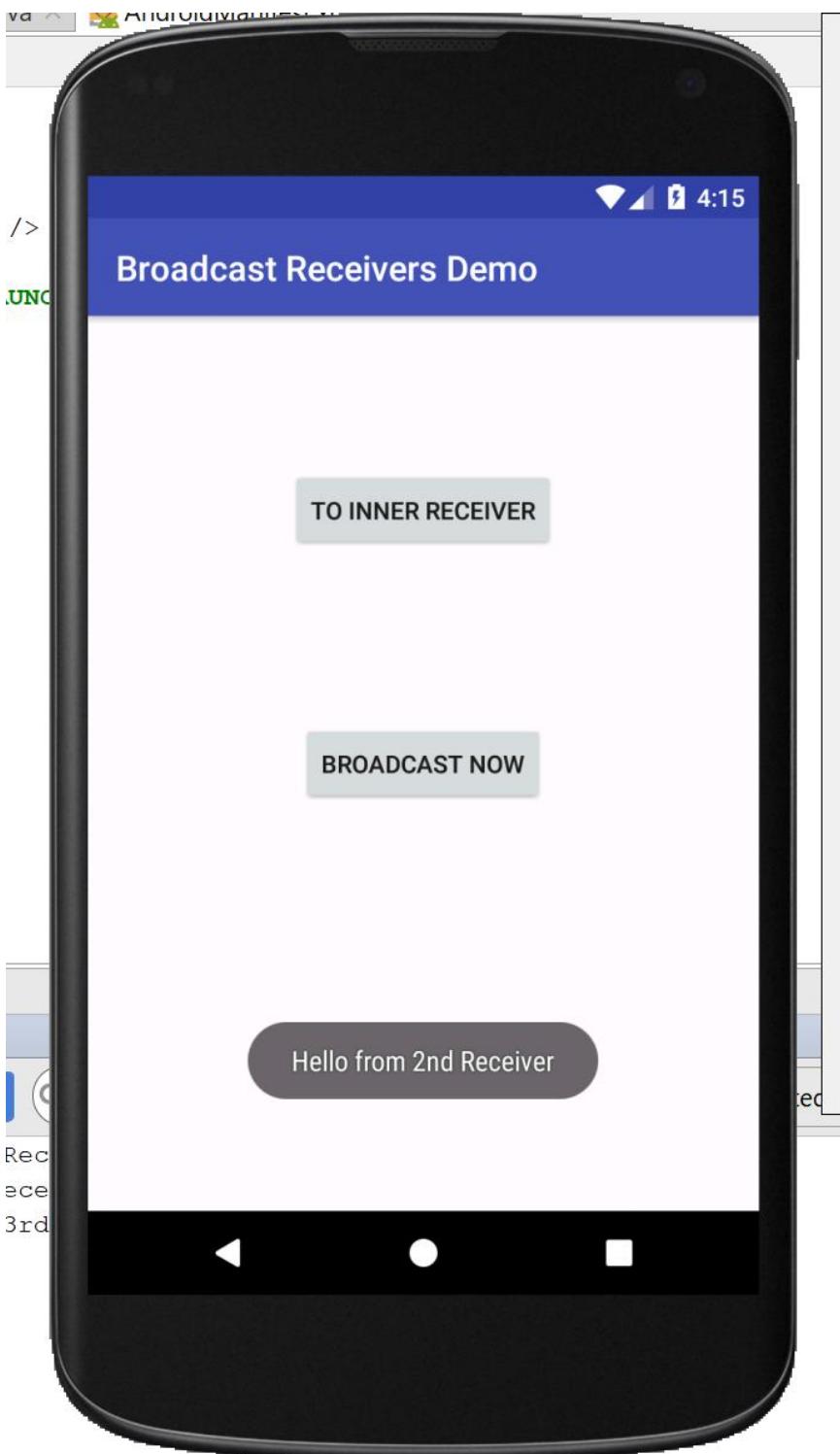
        <receiver android:name=".MainActivity$MyThirdReceiverInner">
            <intent-filter>
                <action android:name=".my.custom.action.name" />
            </intent-filter>
        </receiver>
    
```

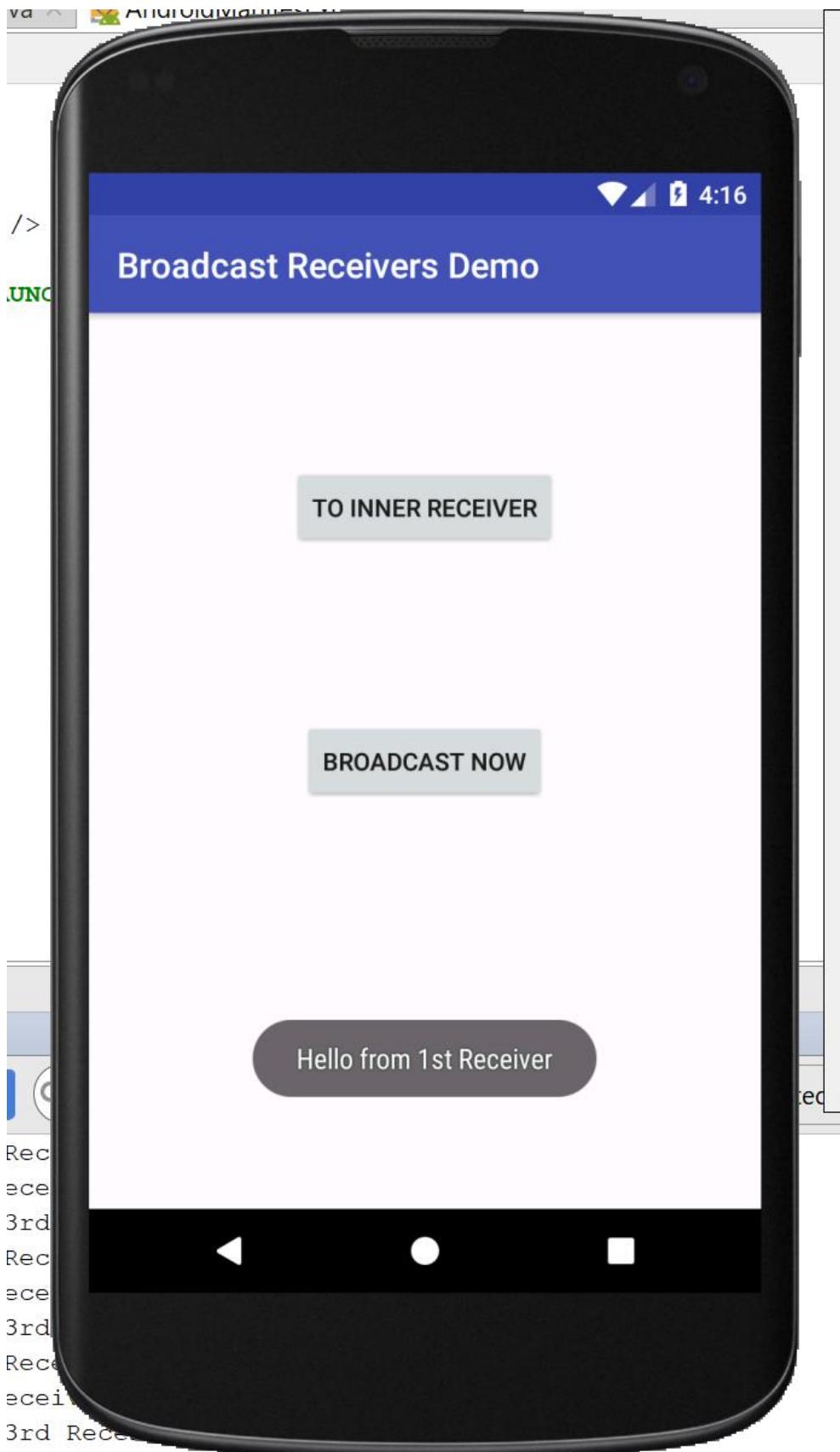
The line containing the code for 'MySecondReceiver' is highlighted with a yellow selection bar. The code for 'MyFirstReceiver' is positioned below it. The bottom status bar indicates 'Gradle build finished in 6s 72ms (6 minutes ago)'.

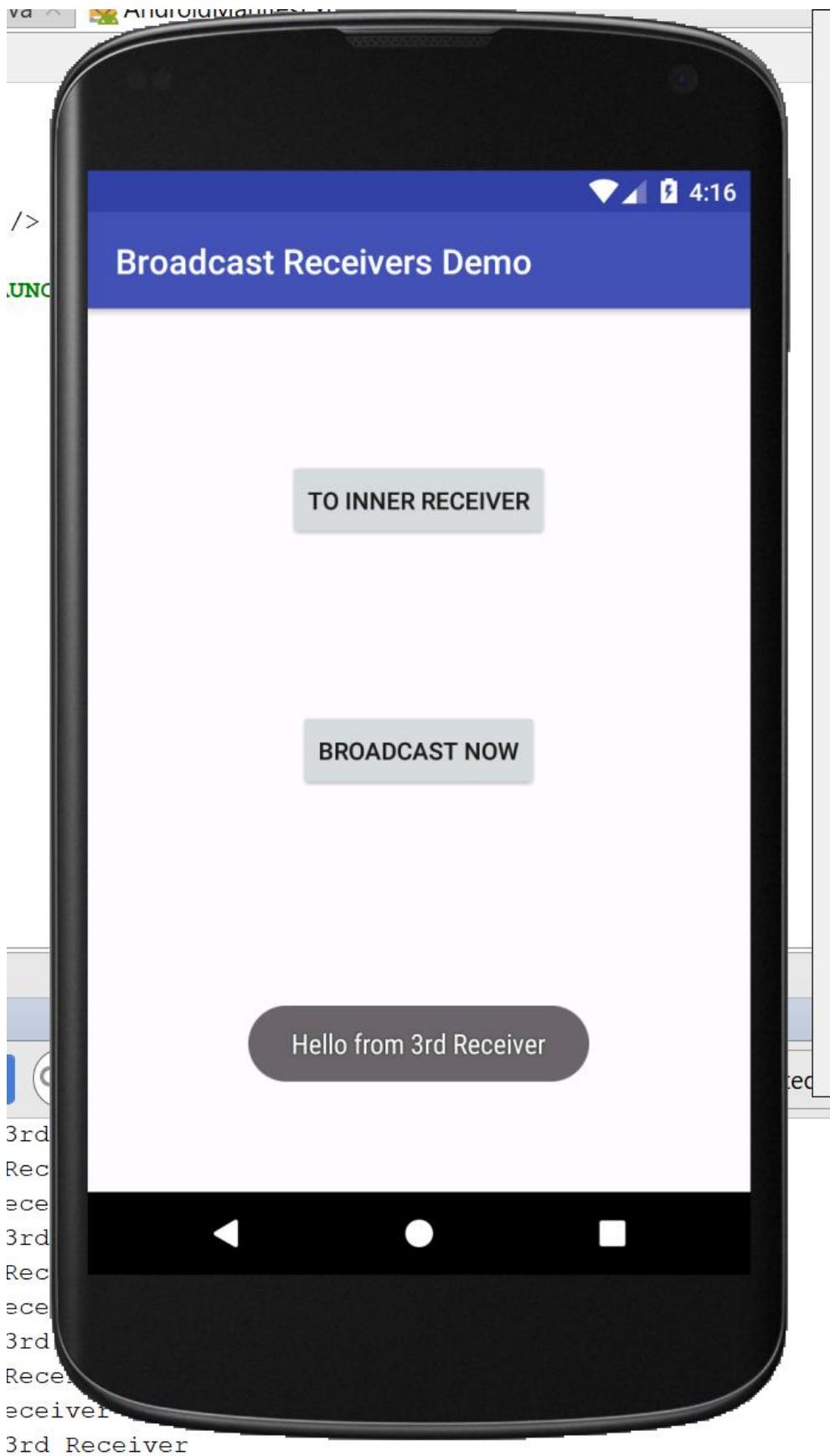
Now let's run the app:



Click the “Broadcast Now” button:







BroadcastReceiversDemo - [C:\Android Demos\Broadcast Receivers\BroadcastReceiversDemo] - [app] - ...\\app\\src\\main\\AndroidManifest.xml - Android Studio 3.0.1

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

BroadcastReceiversDemo > app > src > main > AndroidManifest.xml

Project Structure Captures Gradle

```

<manifest>
    <application>
        <activity android:name=".MainActivity" />
        <activity android:name=".MySecondReceiver" />
        <activity android:name=".MyFirstReceiver" />
    </application>
    <receiver android:name=".MySecondReceiver">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </receiver>
    <receiver android:name=".MyFirstReceiver">
        <intent-filter>
            <action android:name=".my.custom.action.name" />
        </intent-filter>
    </receiver>
</manifest>

```

Logcat

Emulator Nexus\_4 API\_25 Android 7.1.1, API 25 com.example.reza.broadcastreceiversdemo [6523]

Verbose Receiver Regex Show only selected application

03-24 04:15:06.804 6523-6523/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver  
03-24 04:15:06.864 6523-6523/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver  
03-24 04:15:07.042 6523-6523/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver

Build Variants Favorites Device File Explorer

Run TODO Logcat Android Profiler Terminal Messages Event Log Gradle Console

Gradle build finished in 3s 333ms (a minute ago)

So, our three receivers received the broadcast in the exact order that they are declared in the manifest file.

And this order is preserved and is always the same no matter how many times you send the broadcast by clicking the “Broadcast Now” button as the Logcat window shows:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The Project tool window on the left shows the directory structure under 'app': 'manifests' (containing 'AndroidManifest.xml'), 'java' (containing 'com.example.reza.broadcastreceivers' with files 'MainActivity', 'MyFirstReceiver', 'MySecondReceiver', and 'MyThirdReceiver'), and 'res'. The 'activity\_main.xml' file is selected in the top navigation bar. The 'Logcat' tab is active, displaying logs from an 'Emulator Nexus\_4 API\_25' device. The logs show the sequence of broadcasts being received by the three receivers. The 'Text' tab is selected in the bottom-left of the Logcat panel, and the 'Merged Manifest' tab is also visible. The bottom status bar indicates a 'Gradle build finished in 3s 333ms (6 minutes ago)'.

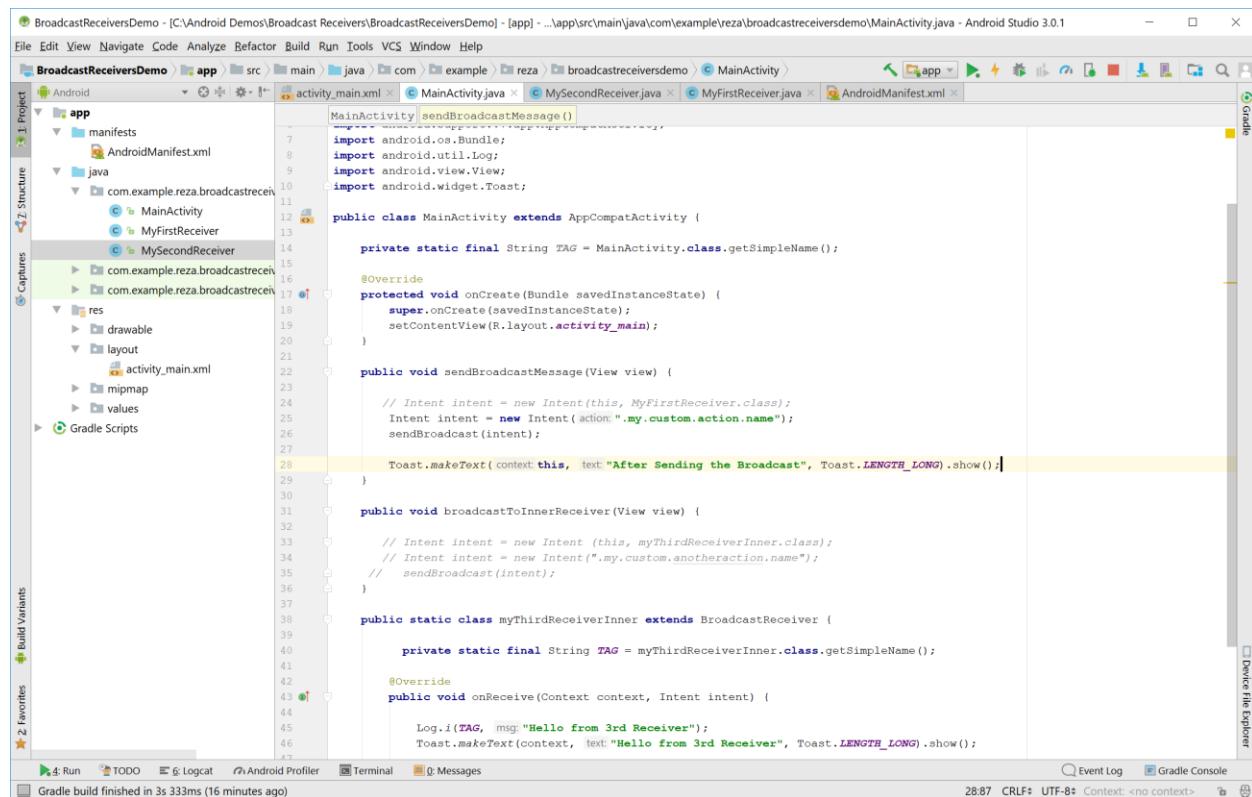
```
03-24 04:15:06.804 6523-6523/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 04:15:06.864 6523-6523/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
03-24 04:15:07.042 6523-6523/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
03-24 04:15:35.734 6523-6523/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 04:15:35.839 6523-6523/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
03-24 04:15:35.963 6523-6523/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
03-24 04:15:59.833 6523-6523/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 04:15:59.965 6523-6523/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver
03-24 04:16:00.259 6523-6523/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
03-24 04:16:20.375 6523-6523/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 04:16:20.394 6523-6523/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
```

Note that all these receivers have the same action name and that's the reason that the sequence in which they are declared inside the Androidmanifest.xml file determines the sequence in which the broadcast will be delivered to these receivers.

## Broadcasts are sent Asynchronously

Anytime a broadcast is sent, it is sent asynchronously, another word in the background. To see this in action you can add a toast statement in the code for sendBroadcastMessage() method right after the sendBroadcast(intent); method and you will see that this toast message appears before any receivers receive the broadcast.

Modify the main activity code as shown (add line 28):



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file. Line 28 has been modified to include a toast message:

```
    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(this, MyFirstReceiver.class);
        intent.setAction("my.custom.action.name");
        sendBroadcast(intent);

        Toast.makeText(this, "After Sending the Broadcast", Toast.LENGTH_LONG).show();
    }
```

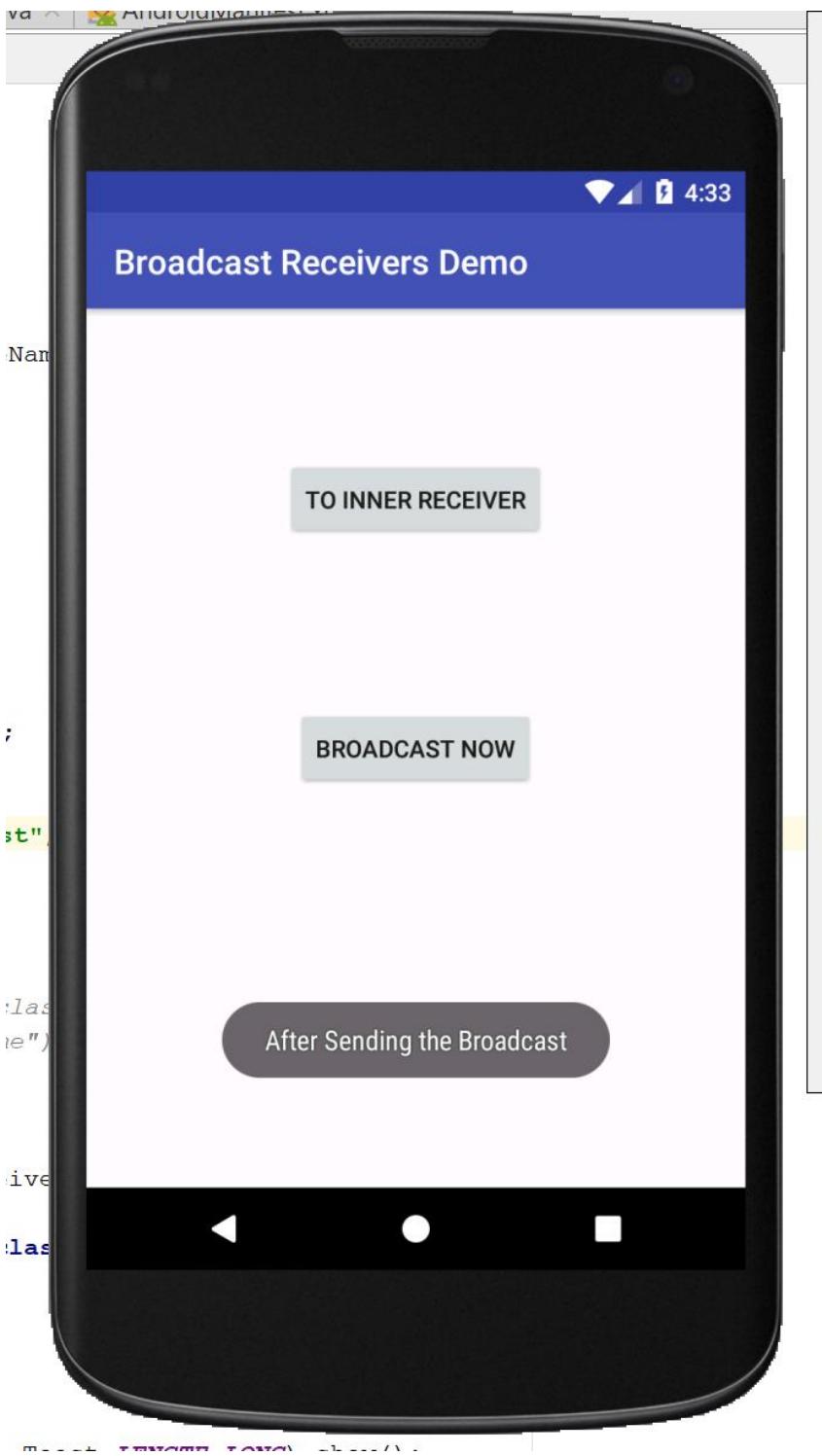
Now line 28 which is going to display a toast will be executed even before line 26 has finished since the program doesn't wait for line 26 to finish hence proving that sending broadcasts is done asynchronously.

Run the app:



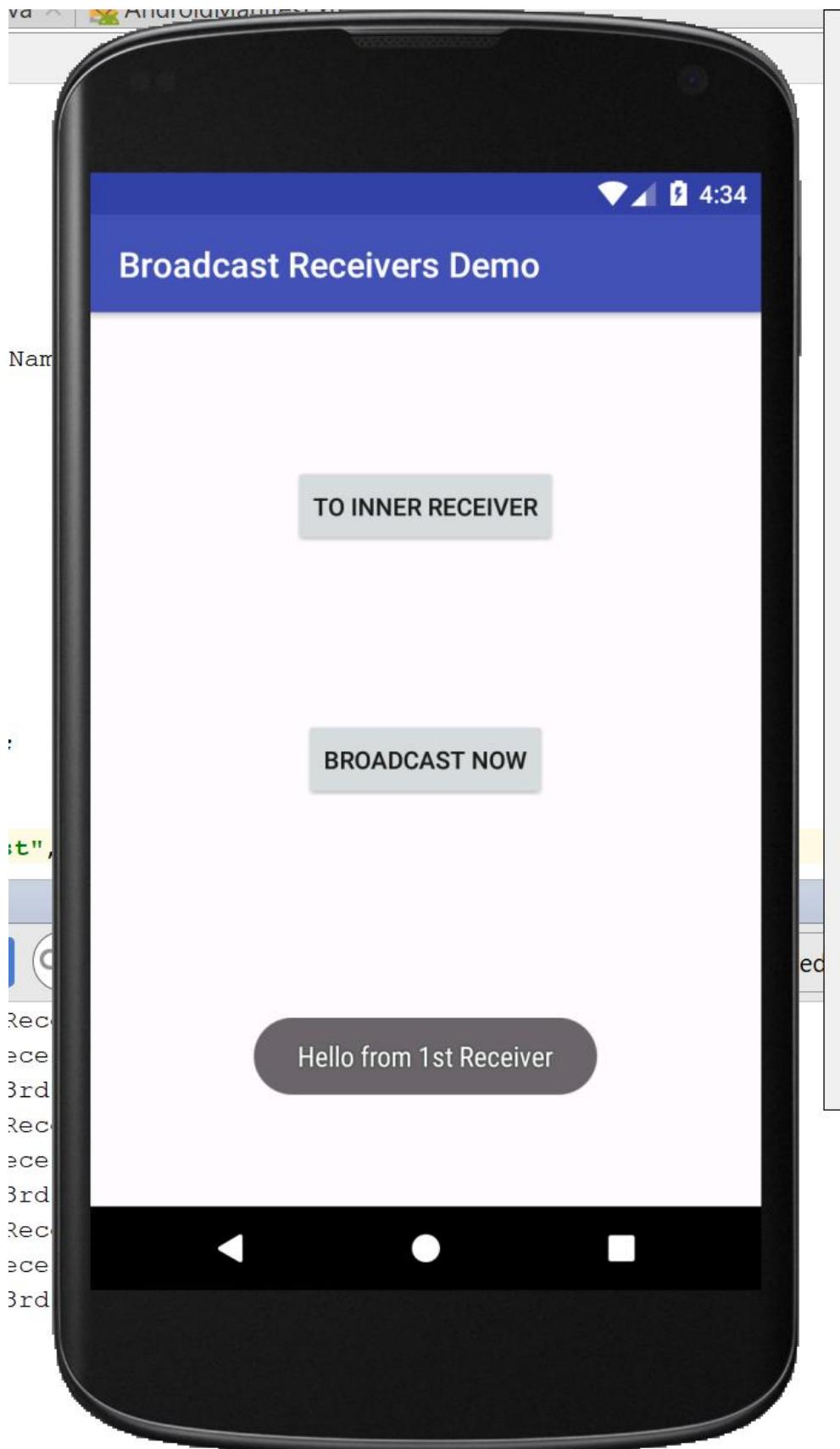
Click the “Broadcast Now” button and here are the toast messages in the order they will be displayed:

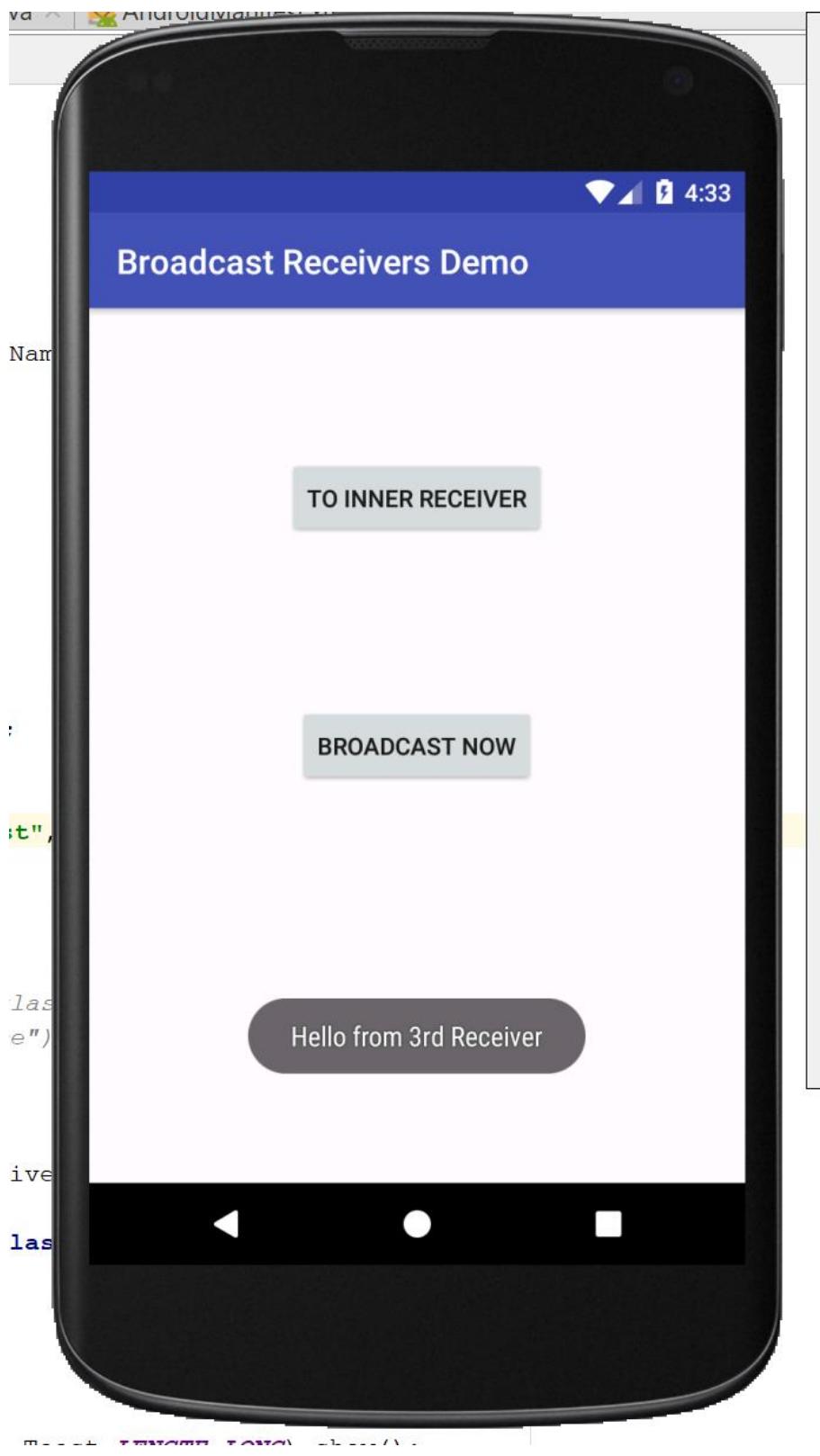
The very first one says “After Sending the Broadcast” when no broadcast has even been sent:



And here come the broadcasts once they have been received in the order they have been received:







The fact that the toast statement got executed right after calling the sendBroadcast(intent); and before returning from this method call, proves that broadcasts are sent asynchronously.

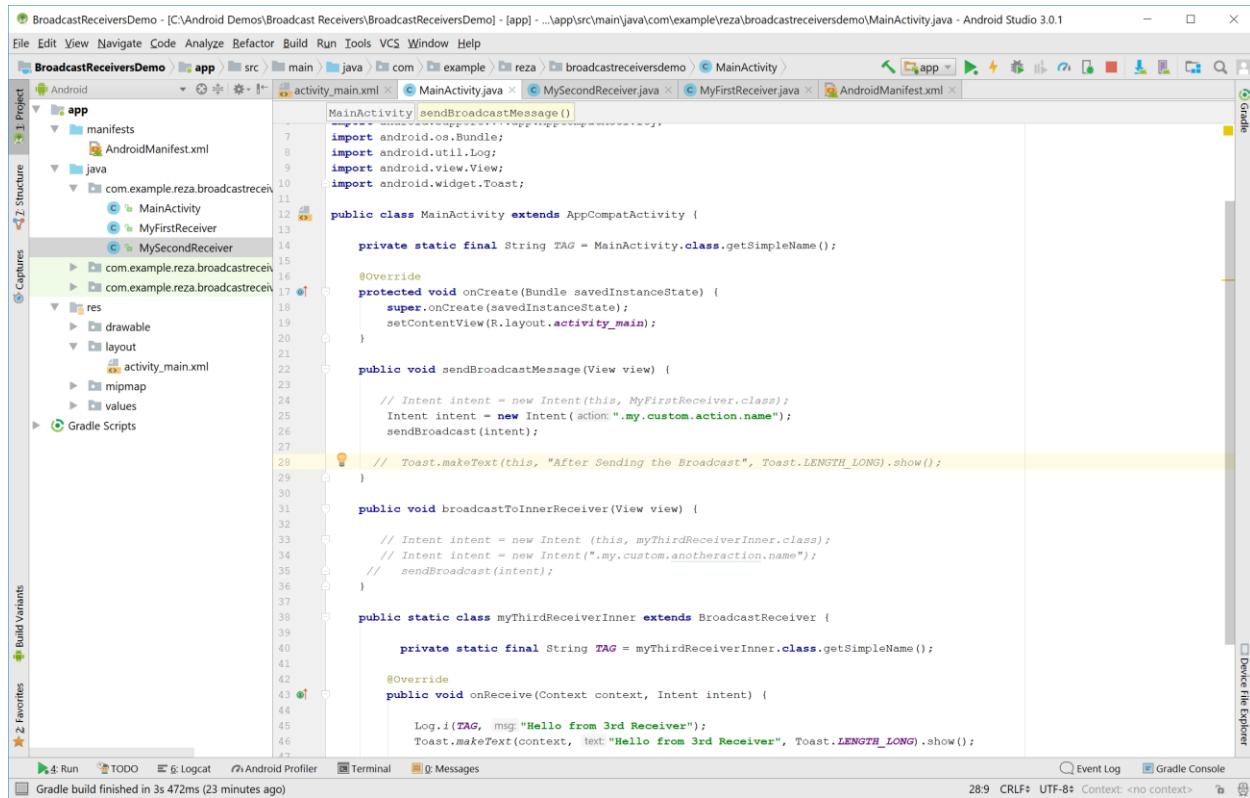
This is an important property of Broadcast Receivers.

Another important property is that the Broadcast Receivers are executed in the main thread which means they can not contain a long running task another word you can not write code to perform a long running task inside the Broadcast Receiver for example you cannot download a file inside the onReceive() method of a Broadcast Receiver.

If you do perform a long running task inside the Broadcast Receiver, then the system will display an ANR (Application Not Responding) dialog and your app will crash. The Android documentation mentions a “Read Timeout of 10 Seconds”, another word if you perform some operation that will take some time around 10 seconds or more, then your app will probably crash.

So, what is the solution to this problem? You should use services for long duration tasks.

Now it is easy to check out if our Broadcast Receiver works in the main thread. To do so first comment out line 28 that we added in the previous step as shown:

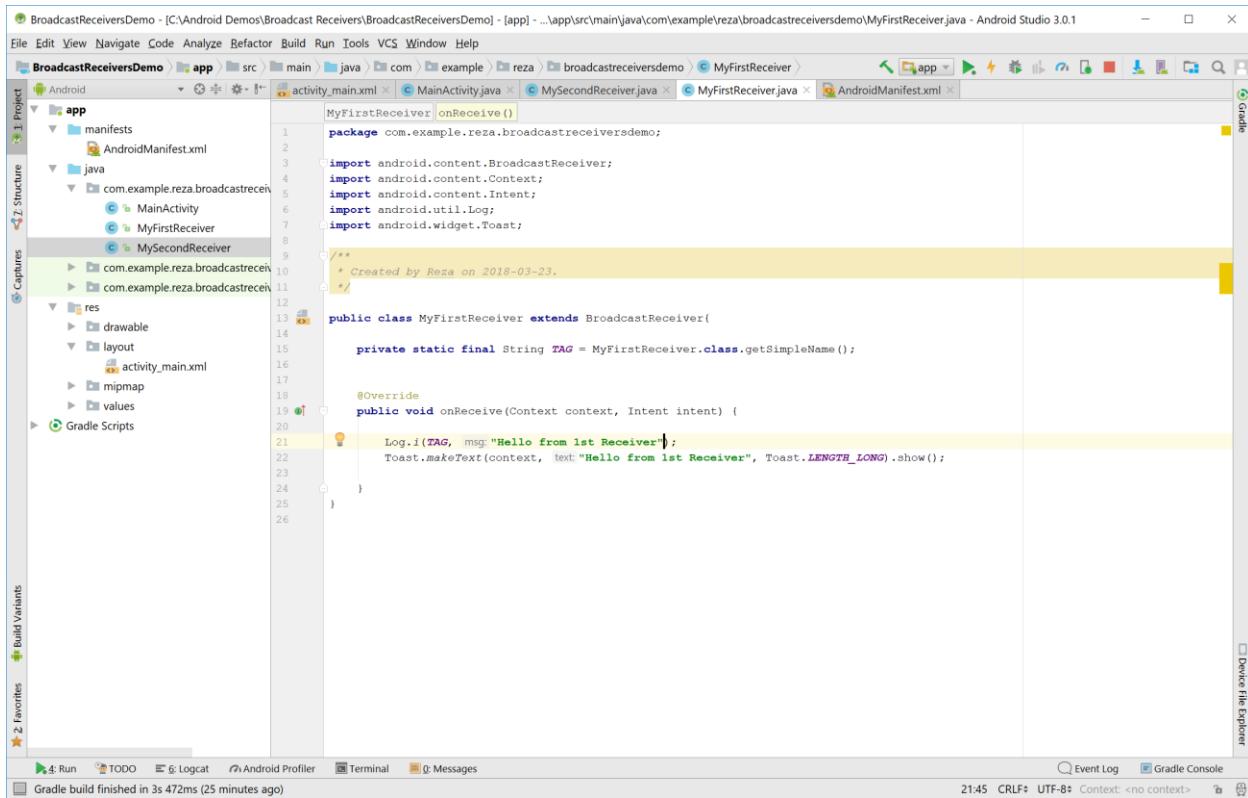


The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file. Line 28, which contains a comment about displaying a toast message after sending a broadcast, is highlighted with a yellow background. The code is as follows:

```
    public void sendBroadcastMessage(View view) {
        // Intent intent = new Intent(this, MyFirstReceiver.class);
        Intent intent = new Intent("my.custom.action.name");
        sendBroadcast(intent);
        // Toast.makeText(this, "After Sending the Broadcast", Toast.LENGTH_LONG).show();
    }
```

The bottom status bar indicates a successful Gradle build: 'Gradle build finished in 3s 472ms (23 minutes ago)'. The bottom right corner shows the page number '28.9 CRLF: UTF-8: Context: <no context>'.

Now go to the MyFirstReceiver.java file and modify it as shown. This is how it is currently looks like:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the 'MyFirstReceiver.java' file under the 'src/main/java/com/example/reza/broadcastreceiversdemo' package. The code is as follows:

```
MyFirstReceiver.onReceive() {
    package com.example.reza.broadcastreceiversdemo;
    import android.content.BroadcastReceiver;
    import android.content.Context;
    import android.content.Intent;
    import android.util.Log;
    import android.widget.Toast;

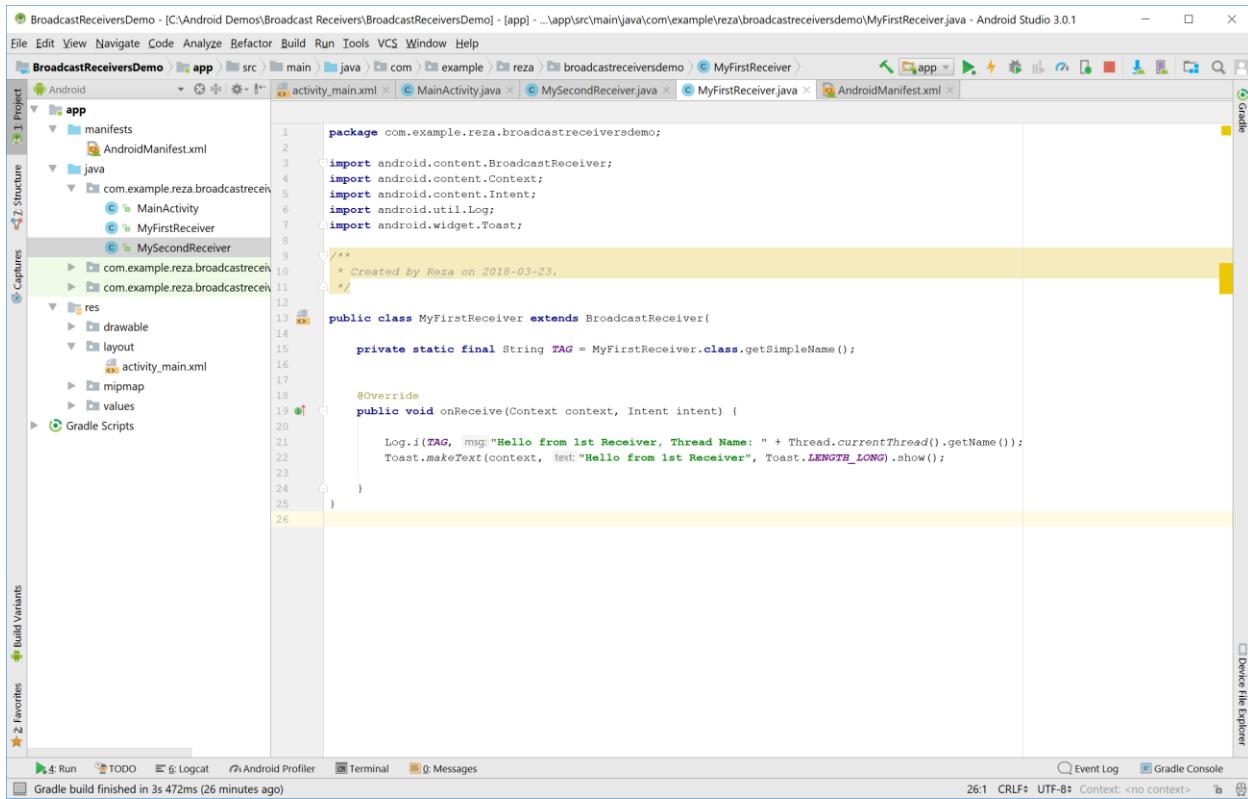
    /**
     * Created by Reza on 2018-03-23.
     */

    public class MyFirstReceiver extends BroadcastReceiver {
        private static final String TAG = MyFirstReceiver.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg: "Hello from 1st Receiver");
            Toast.makeText(context, text: "Hello from 1st Receiver", Toast.LENGTH_LONG).show();
        }
    }
}
```

The code editor highlights the 'msg:' and 'text:' parts of the toast message, indicating they are intended to be replaced.

Modify it with modifying line 21 as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the file 'MyFirstReceiver.java' under the package 'com.example.reza.broadcastreceiversdemo'. The code defines a class 'MyFirstReceiver' that extends 'BroadcastReceiver'. It overrides the 'onReceive' method to log a message and show a toast. A comment at the top of the class indicates it was created by Reza on 2018-03-23.

```
package com.example.reza.broadcastreceiversdemo;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

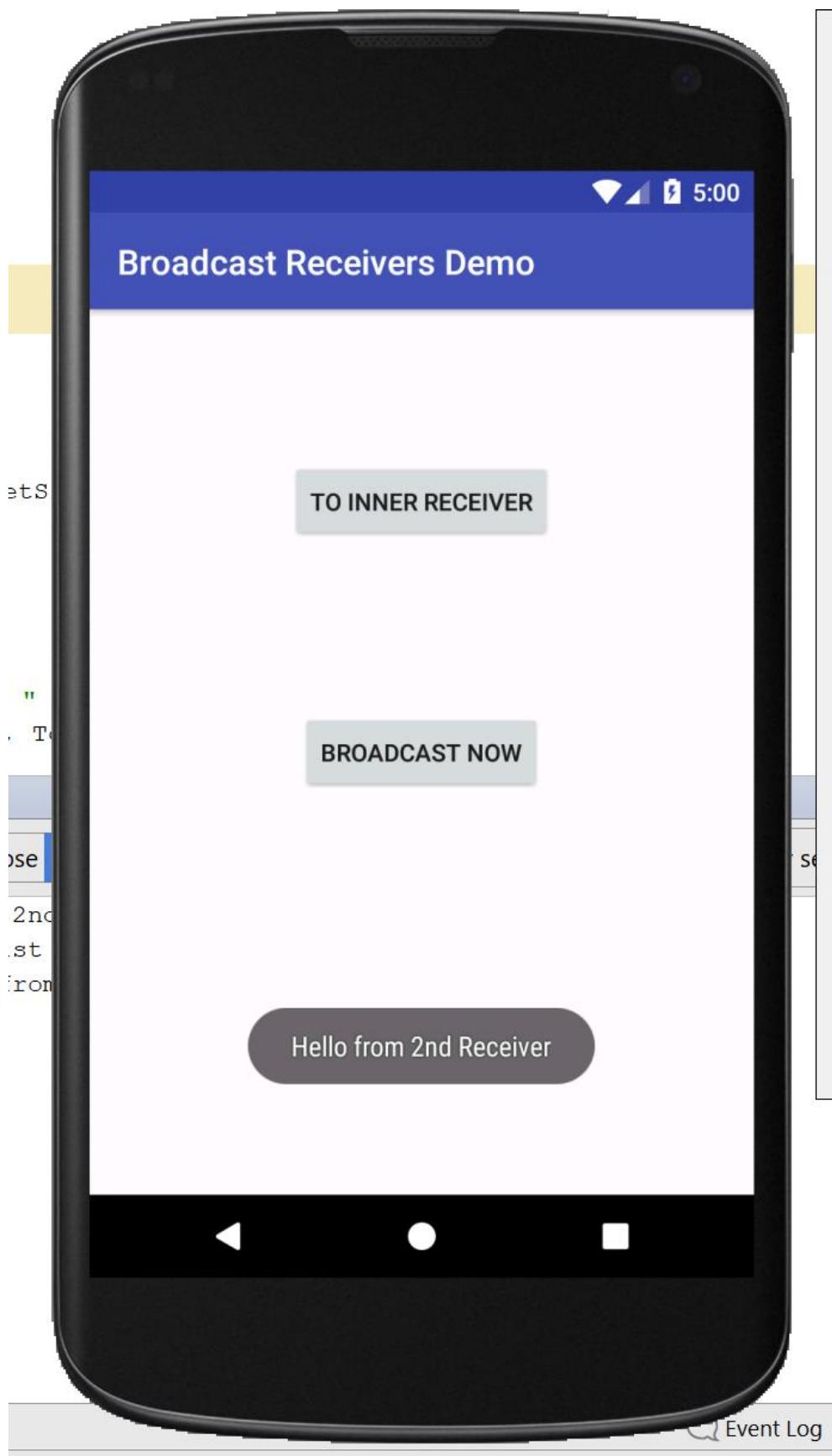
/**
 * Created by Reza on 2018-03-23.
 */
public class MyFirstReceiver extends BroadcastReceiver {

    private static final String TAG = MyFirstReceiver.class.getSimpleName();

    @Override
    public void onReceive(Context context, Intent intent) {
        Log.i(TAG, msg:"Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
        Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();
    }
}
```

Now run the app and click on “Broadcast Now” button:





Now in the screen capture below you see that the thread name is “main”.

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the file `MyFirstReceiver.java`, which contains the following code:

```
package com.example.reza.broadcastreceiversdemo;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

/**
 * Created by Reza on 2018-03-23.
 */
public class MyFirstReceiver extends BroadcastReceiver {

    private static final String TAG = MyFirstReceiver.class.getSimpleName();

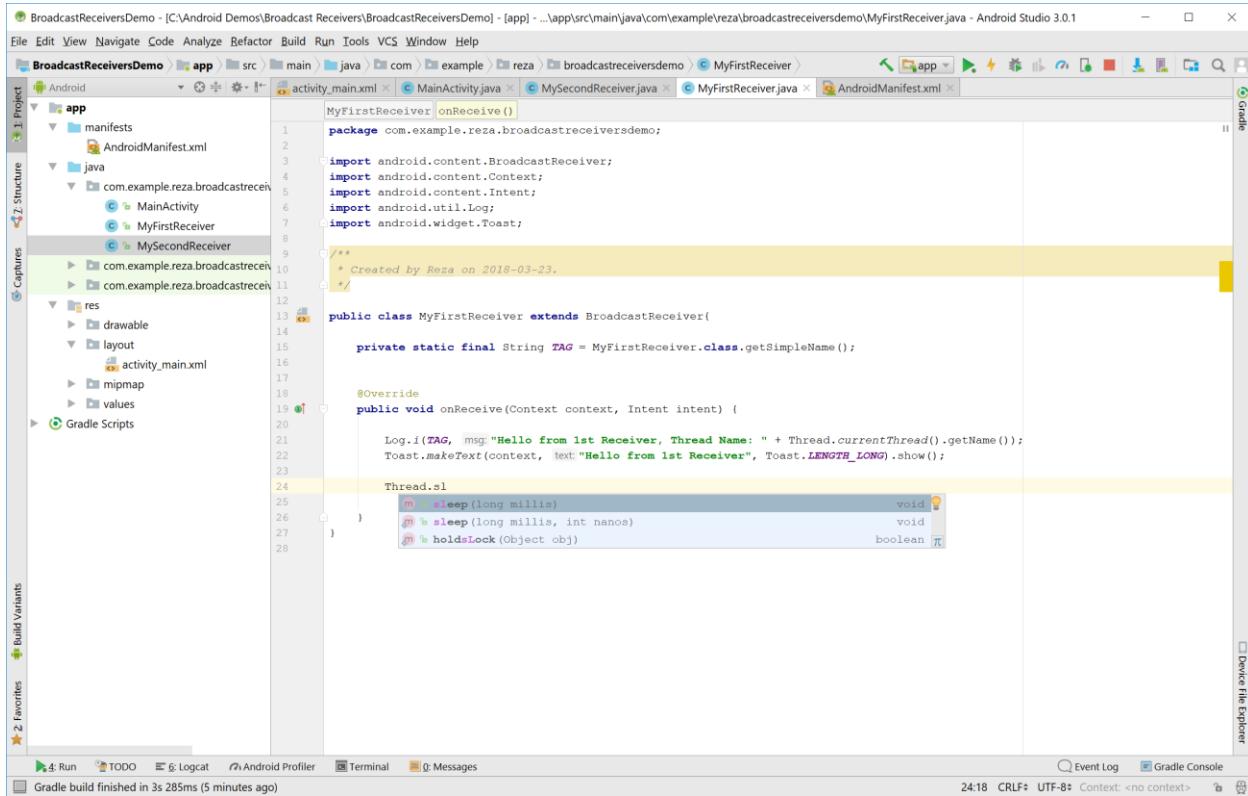
    @Override
    public void onReceive(Context context, Intent intent) {
        Log.i(TAG, msg: "Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
        Toast.makeText(context, text: "Hello from 1st Receiver", Toast.LENGTH_LONG).show();
    }
}
```

The Logcat tab shows the following log entries:

```
03-24 05:00:29.706 7076-7076/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 05:00:29.812 7076-7076/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver, Thread Name: main
03-24 05:00:29.972 7076-7076/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
```

Now if you perform some heavy operation inside the `onReceive()` method, then your application will display an ANR dialog and it will crash and during that process your app user interface will be blocked.

To demonstrate let's modify the `onReceive()` method in "MyFirstReceiver.java" file as shown:



The screenshot shows the Android Studio interface with the project "BroadcastReceiversDemo" open. The code editor displays the `MyFirstReceiver.java` file. The code defines a `BroadcastReceiver` named `MyFirstReceiver` that overrides the `onReceive()` method. Inside the `onReceive()` method, it logs a message and displays a toast. A code completion dropdown is visible at the bottom of the code editor, showing options like `sleep` and `holdLock`.

```
MyFirstReceiver.onReceive()  
1 package com.example.reza.broadcastreceiversdemo;  
2  
3 import android.content.BroadcastReceiver;  
4 import android.content.Context;  
5 import android.content.Intent;  
6 import android.util.Log;  
7 import android.widget.Toast;  
8  
9 /* * Created by Reza on 2018-03-23. */  
10  
11 public class MyFirstReceiver extends BroadcastReceiver {  
12  
13     private static final String TAG = MyFirstReceiver.class.getSimpleName();  
14  
15     @Override  
16     public void onReceive(Context context, Intent intent) {  
17  
18         Log.i(TAG, msg: "Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());  
19         Toast.makeText(context, text: "Hello from 1st Receiver", Toast.LENGTH_LONG).show();  
20  
21         Thread.sleep(  
22             long millis);  
23         long millis, int nanos);  
24     }  
25  
26     void void;  
27     boolean boolean;
```

The screenshot shows the Android Studio interface with the code editor open to a Java file named `MyFirstReceiver.java`. The code contains a call to `Thread.sleep(10000);`. A tooltip appears over the call, listing three options: "Surround with try/catch", "Add on demand static import for 'java.lang.Thread'", and "Annotate class 'Thread' as @Deprecated".

```

    package com.example.reza.broadcastreceiversdemo;

    import android.content.BroadcastReceiver;
    import android.content.Context;
    import android.content.Intent;
    import android.util.Log;
    import android.widget.Toast;

    /**
     * Created by Reza on 2018-03-23.
     */

    public class MyFirstReceiver extends BroadcastReceiver {
        private static final String TAG = MyFirstReceiver.class.getSimpleName();

        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg:"Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
            Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();
            Thread.sleep(millis: 10000);
        }
    }
  
```

The screenshot shows the same Java file after the user has selected the "Surround with try/catch" option from the tooltip. The code now includes a try-catch block around the `Thread.sleep()` call, with a stack trace printed if an exception occurs.

```

    package com.example.reza.broadcastreceiversdemo;

    import android.content.BroadcastReceiver;
    import android.content.Context;
    import android.content.Intent;
    import android.util.Log;
    import android.widget.Toast;

    /**
     * Created by Reza on 2018-03-23.
     */

    public class MyFirstReceiver extends BroadcastReceiver {
        private static final String TAG = MyFirstReceiver.class.getSimpleName();

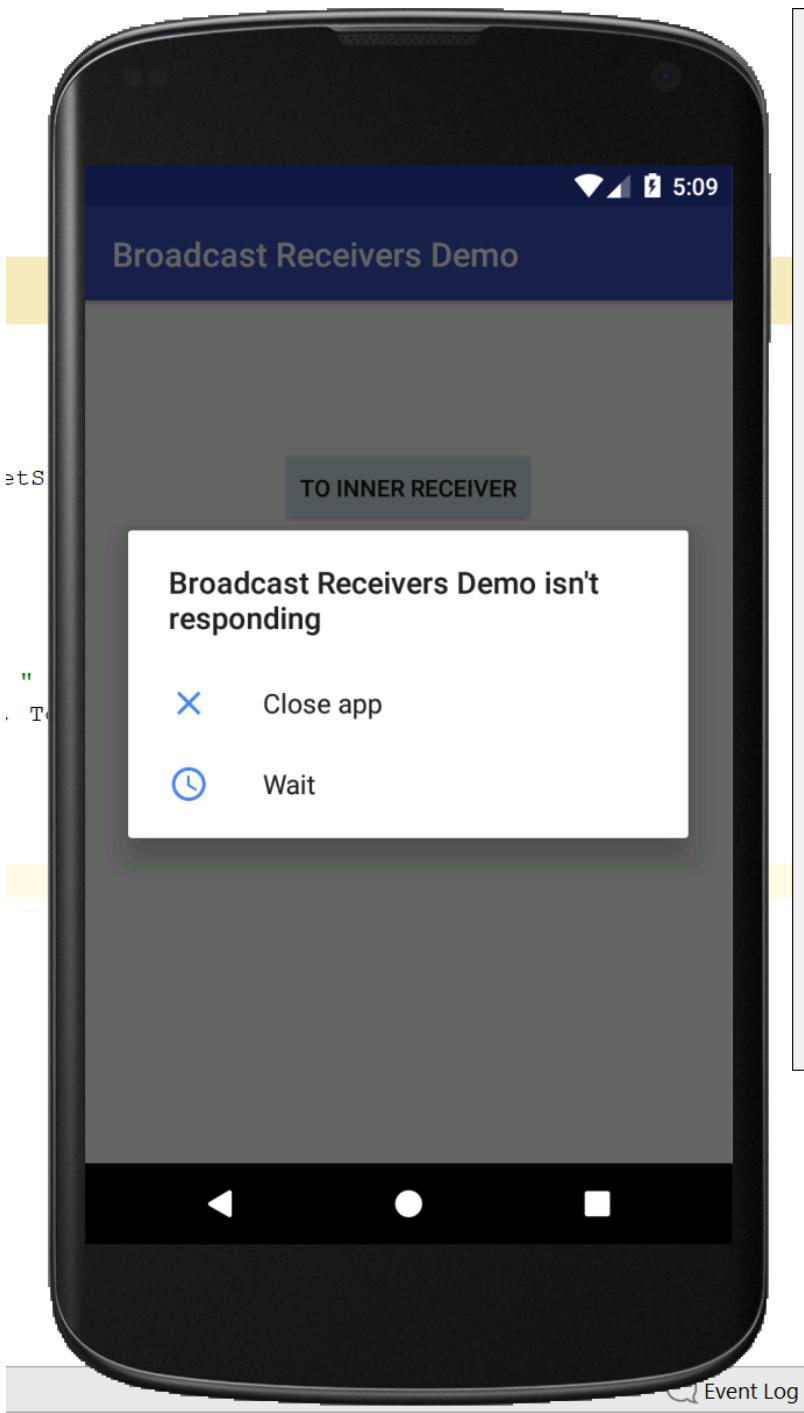
        @Override
        public void onReceive(Context context, Intent intent) {
            Log.i(TAG, msg:"Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
            Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();

            try {
                Thread.sleep(millis: 10000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
  
```

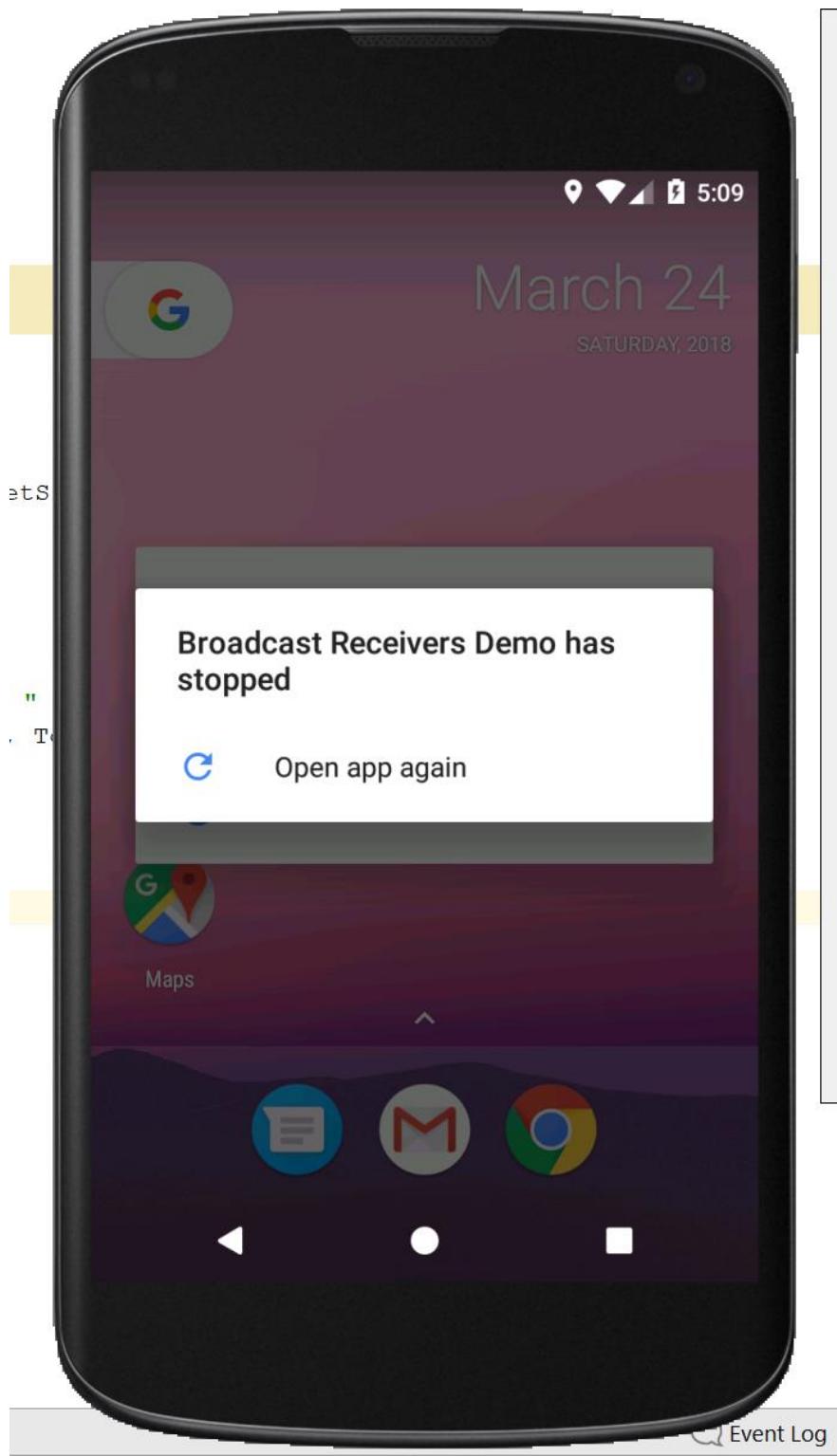
Now run the app:



Click “Broadcast Now” button and then click “To Inner Receiver” button a few times. You will see that the “To Inner Receiver” button is not responsive and after a few seconds you will get the following message from Android operating system:

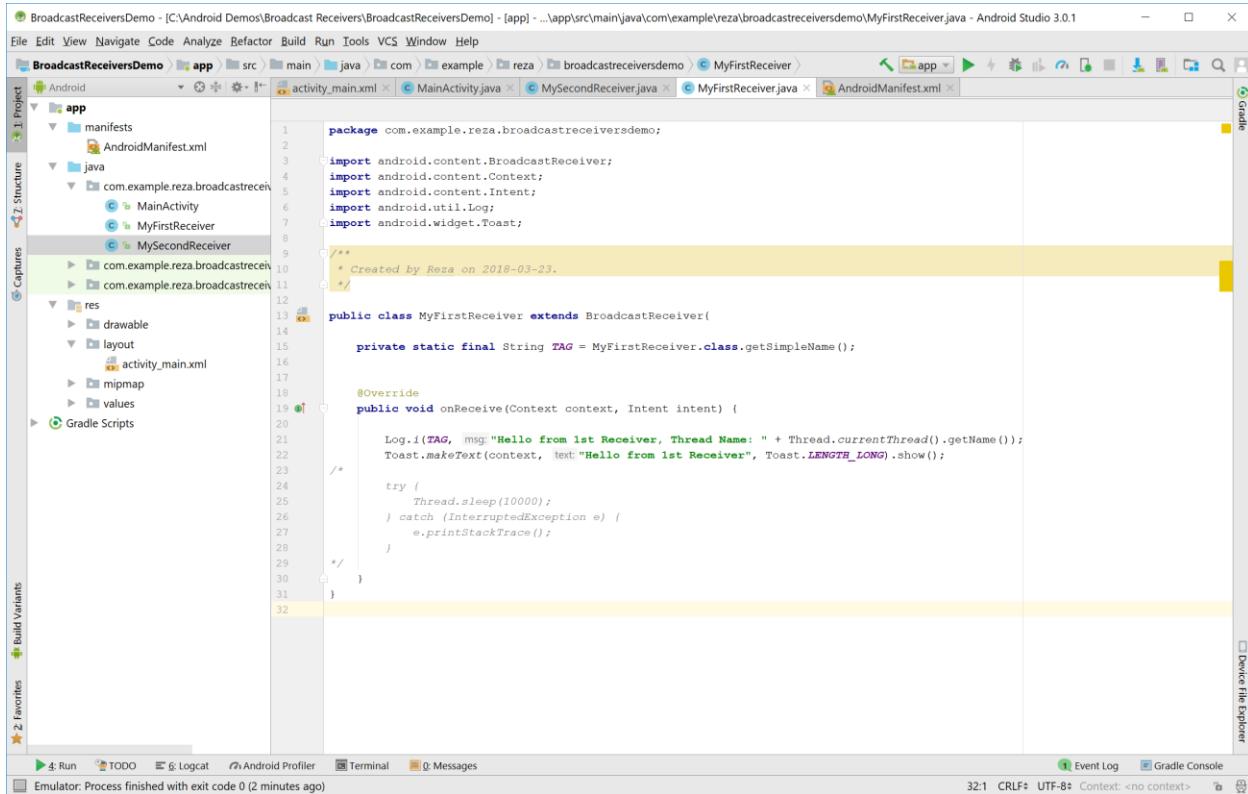


And then you will get the following:



## How to Pass Data from the Activity to a Receiver:

First comment out the code (the long running task) we added to illustrate the previous concept in your “MyFirstReceiver.java” file as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays 'MyFirstReceiver.java' under the package 'com.example.reza.broadcastreceiversdemo'. The code implements a BroadcastReceiver with a sleep operation commented out.

```
package com.example.reza.broadcastreceiversdemo;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

/**
 * Created by Reza on 2018-03-23.
 */

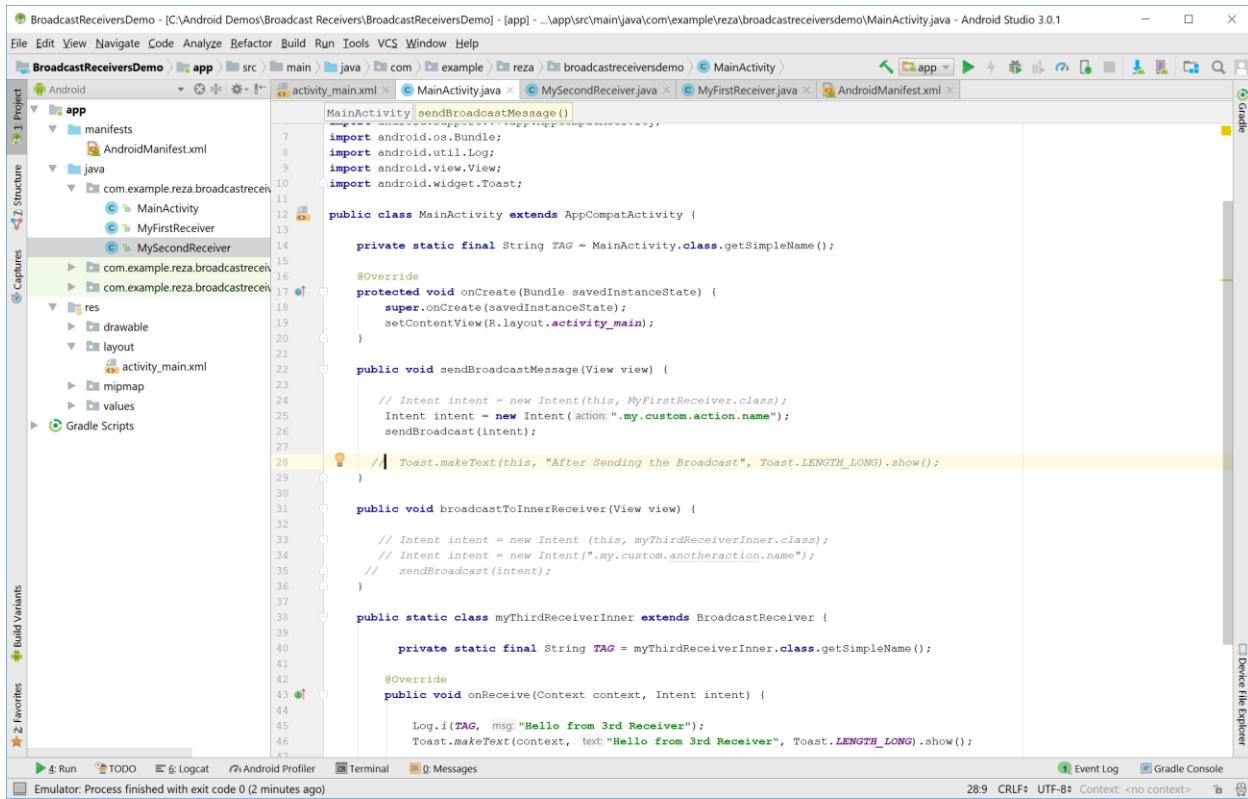
public class MyFirstReceiver extends BroadcastReceiver {

    private static final String TAG = MyFirstReceiver.class.getSimpleName();

    @Override
    public void onReceive(Context context, Intent intent) {
        Log.i(TAG, msg:"Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
        Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();

        /*
        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        */
    }
}
```

Then go to MainActivity.java. It is currently looks like as follows:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file, which contains the following code:

```
MainActivity sendBroadcastMessage()
{
    import android.os.Bundle;
    import android.util.Log;
    import android.view.View;
    import android.widget.Toast;

    public class MainActivity extends AppCompatActivity {
        private static final String TAG = MainActivity.class.getSimpleName();

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
        }

        public void sendBroadcastMessage(View view) {
            Intent intent = new Intent(this, MyFirstReceiver.class);
            intent.setAction("my.custom.action.name");
            sendBroadcast(intent);

            // Toast.makeText(this, "After Sending the Broadcast", Toast.LENGTH_LONG).show();
        }

        public void broadcastToInnerReceiver(View view) {
            Intent intent = new Intent(this, myThirdReceiverInner.class);
            intent.setAction("my.custom.anotheraction.name");
            sendBroadcast(intent);
        }

        public static class myThirdReceiverInner extends BroadcastReceiver {
            private static final String TAG = myThirdReceiverInner.class.getSimpleName();

            @Override
            public void onReceive(Context context, Intent intent) {
                Log.i(TAG, msg:"Hello from 3rd Receiver");
                Toast.makeText(context, text:"Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
            }
        }
    }
}
```

The code editor highlights the line `// Toast.makeText(this, "After Sending the Broadcast", Toast.LENGTH_LONG).show();` with a yellow background. The bottom status bar shows the message "Emulator: Process finished with exit code 0 (2 minutes ago)".

Uncomment lines 34 and 35 as shown:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file. Lines 34 and 35 are highlighted with a blue selection bar. The code is as follows:

```
private static final String TAG = MainActivity.class.getSimpleName();  
Intent intent = new Intent(action ".my.custom.anotheraction.name");  
sendBroadcast(intent);
```

Now copy the string ".my.custom.anotheraction.name" from line 34 and copy that as the action for your third receiver in the manifest file as shown:

This is how our manifest file currently looks like:

The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The Project tool window on the left shows the app module structure, including the manifests folder containing the AndroidManifest.xml file, and the java folder containing MainActivity, MyFirstReceiver, and MySecondReceiver. The main editor window displays the XML code of the AndroidManifest.xml file. The code defines an application with three receiver components: MySecondReceiver, MyFirstReceiver, and MainActivity\$MyThirdReceiverInner. Each receiver has an intent filter with a specific action. The bottom of the screen shows the Android Studio navigation bar and status bar.

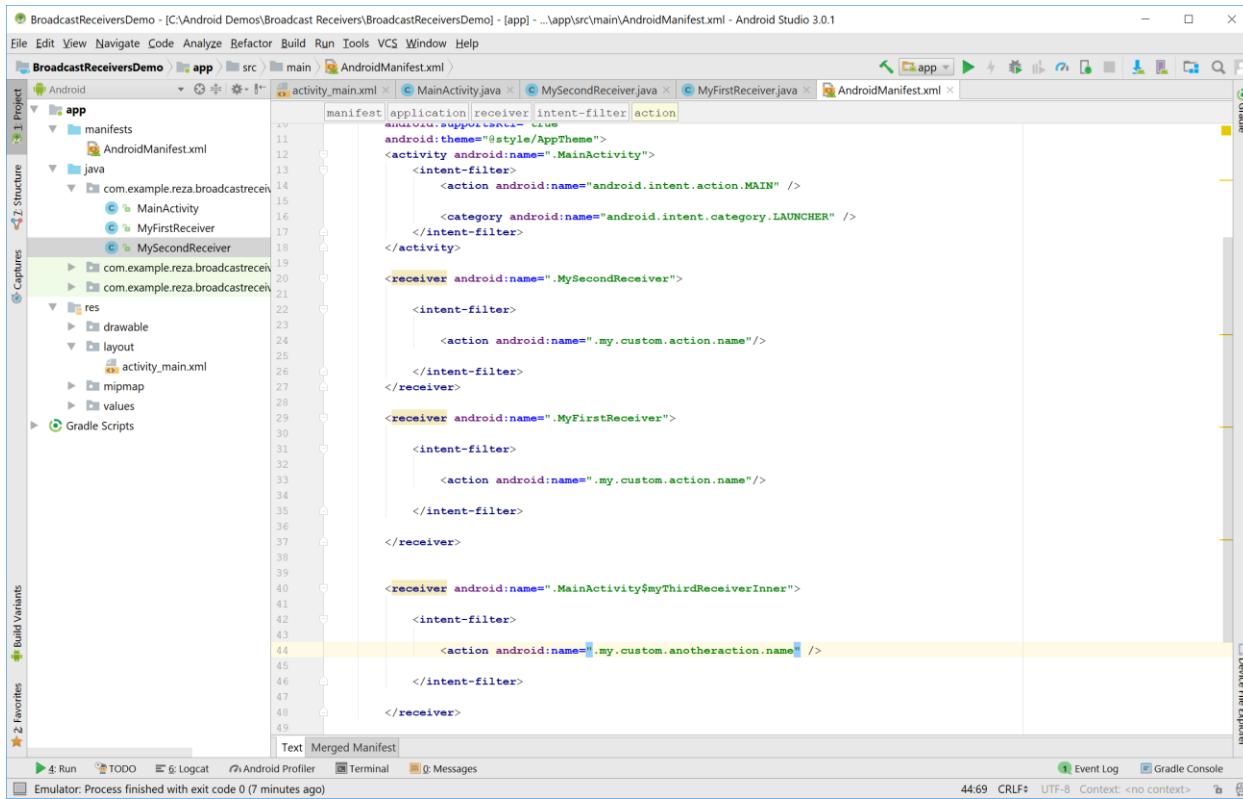
```
<manifest>
    <application android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <receiver android:name=".MySecondReceiver">
            <intent-filter>
                <action android:name=".my.custom.action.name"/>
            </intent-filter>
        </receiver>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name=".my.custom.action.name"/>
            </intent-filter>
        </receiver>

        <receiver android:name=".MainActivity$MyThirdReceiverInner">
            <intent-filter>
                <action android:name=".my.custom.action.name" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

Replace the action in line 44 as shown:

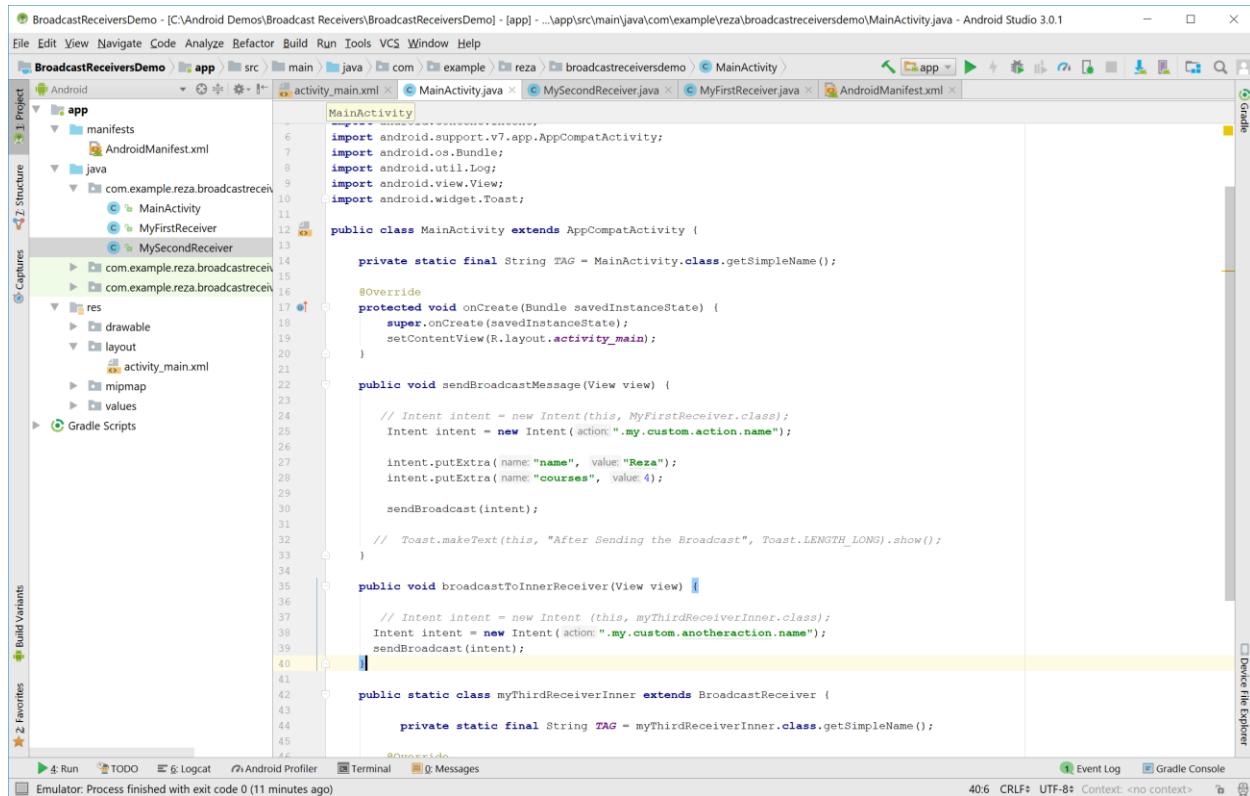


The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The left sidebar displays the project structure, including the app module with its manifest, Java files (MainActivity, MyFirstReceiver, MySecondReceiver), and layout files. The main editor window shows the XML code for the manifest. A yellow highlight is placed over the line containing the action attribute in the receiver declaration at line 44. The code snippet below illustrates the target modification.

```
1<manifest>
2    <application>
3        <activity android:name=".MainActivity">
4            <intent-filter>
5                <action android:name="android.intent.action.MAIN" />
6            </intent-filter>
7        </activity>
8
9        <receiver android:name=".MySecondReceiver">
10            <intent-filter>
11                <action android:name=".my.custom.action.name"/>
12            </intent-filter>
13        </receiver>
14
15        <receiver android:name=".MyFirstReceiver">
16            <intent-filter>
17                <action android:name=".my.custom.action.name"/>
18            </intent-filter>
19        </receiver>
20
21        <receiver android:name=".MainActivity$myThirdReceiverInner">
22            <intent-filter>
23                <action android:name=".my.custom.anotheraction.name" />
24            </intent-filter>
25        </receiver>
26
27    </application>
28
29</manifest>
```

For the purpose of our demonstration in this part, we will be using our first and third receivers.

Now in the MainActivity.java let's modify our sendBroadcastMessage() method and send some data as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the MainActivity.java file, which contains Java code for sending broadcast messages. The code includes imports for AppCompatActivity, Bundle, Log, View, and Toast. It defines a static final String TAG and overrides the onCreate method. The sendBroadcastMessage method creates an Intent with actions for MyFirstReceiver and MySecondReceiver, putting extra data for name ("Reza") and courses (4). The broadcastToInnerReceiver method sends an intent to myThirdReceiverInner. The code editor has syntax highlighting and code completion features.

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private static final String TAG = MainActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void sendBroadcastMessage(View view) {
        Intent intent = new Intent(this, MyFirstReceiver.class);
        intent.setAction("my.custom.action.name");

        intent.putExtra("name", "Reza");
        intent.putExtra("courses", 4);

        sendBroadcast(intent);

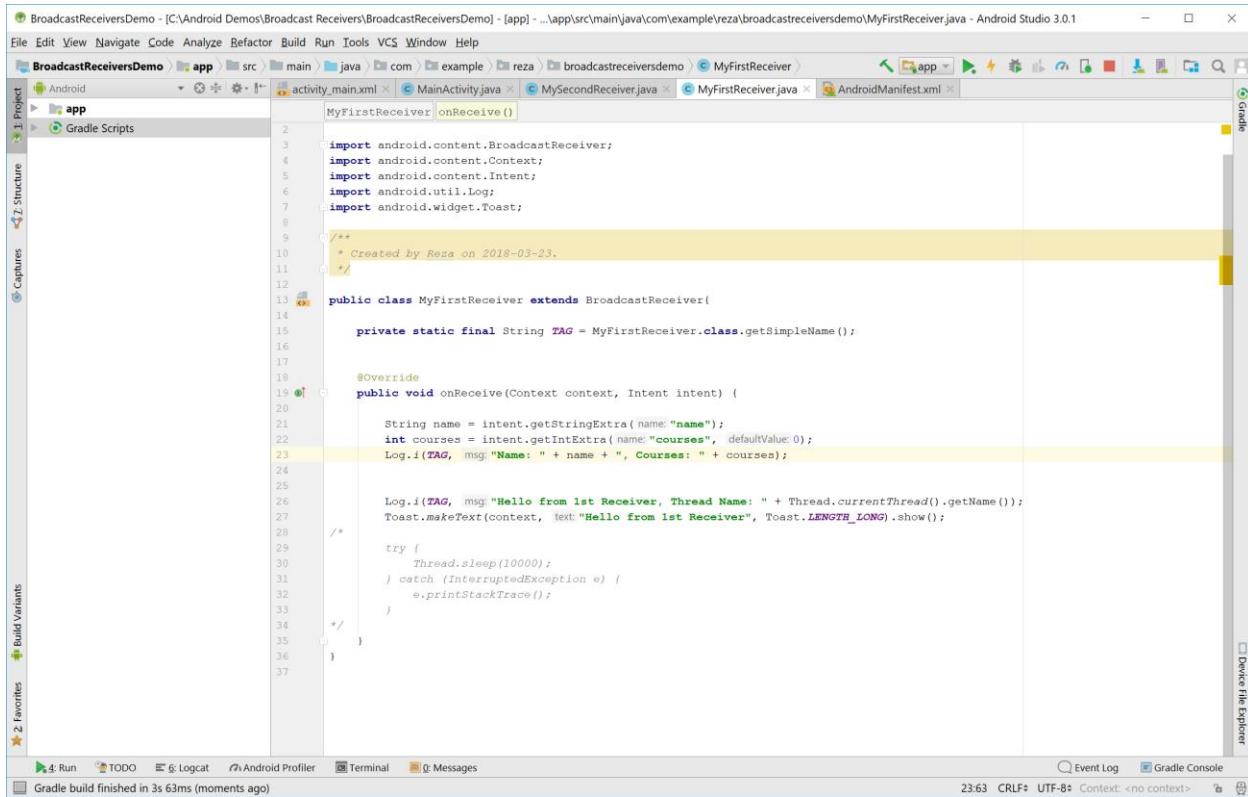
        // Toast.makeText(this, "After Sending the Broadcast", Toast.LENGTH_LONG).show();
    }

    public void broadcastToInnerReceiver(View view) {
        Intent intent = new Intent(this, myThirdReceiverInner.class);
        intent.setAction("my.custom.anotheraction.name");
        sendBroadcast(intent);
    }

    public static class myThirdReceiverInner extends BroadcastReceiver {

        private static final String TAG = myThirdReceiverInner.class.getSimpleName();
    }
}
```

Note that lines 27 and 28 were added. Now let's go to our first receiver (go to MyFirstReceiver.java file) and retrieve this data as shown (lines 21, 22, and 23 were added):



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the 'MyFirstReceiver.java' file, which contains Java code for a BroadcastReceiver. The code includes imports for android.content.BroadcastReceiver, Context, Intent, Log, and Toast. It features an overridden 'onReceive' method that logs the name and courses from the intent extras and displays a toast message. Lines 21, 22, and 23 have been highlighted in yellow, indicating they are new additions.

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

/**
 * Created by Reza on 2018-03-23.
 */

public class MyFirstReceiver extends BroadcastReceiver {

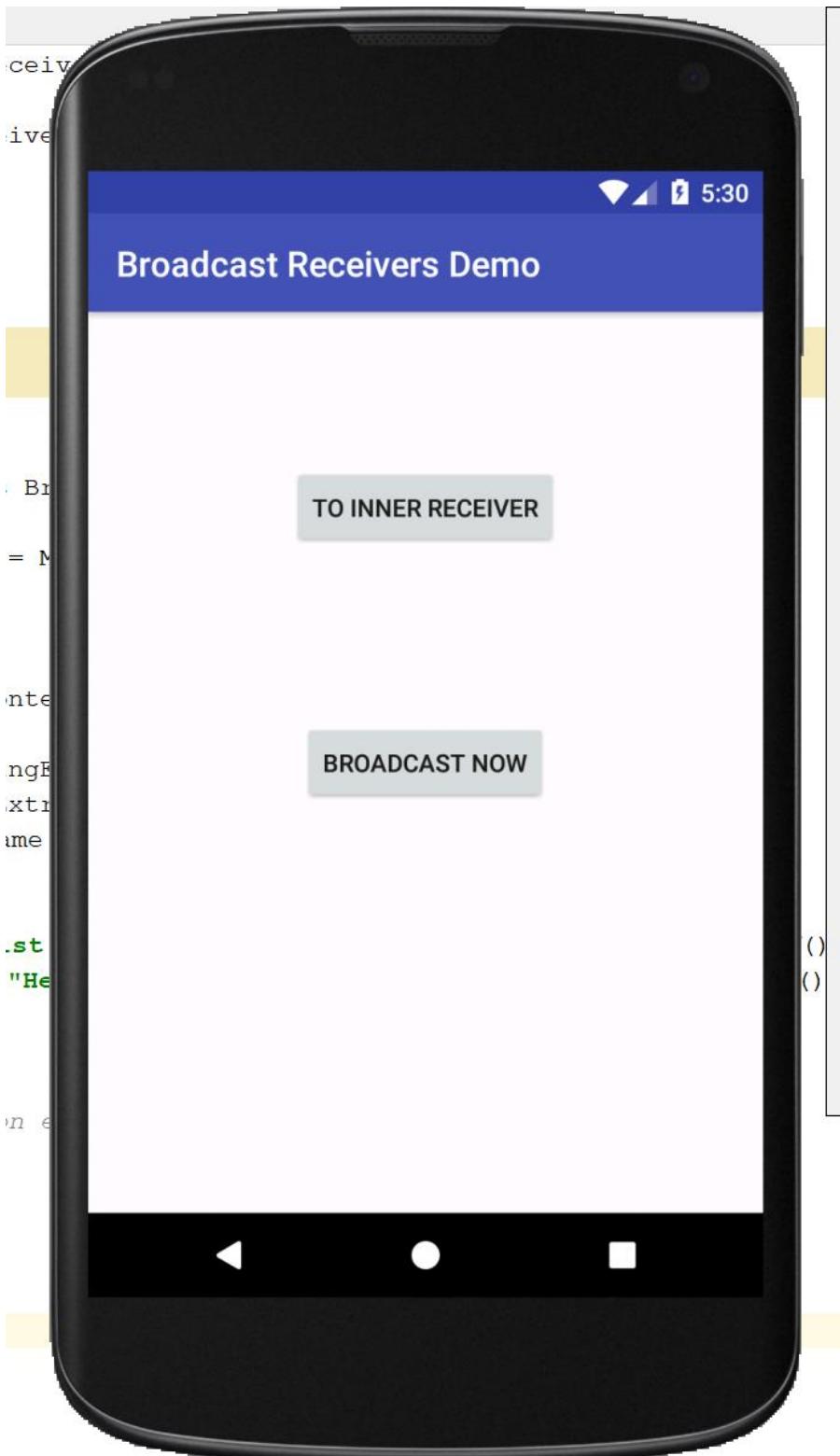
    private static final String TAG = MyFirstReceiver.class.getSimpleName();

    @Override
    public void onReceive(Context context, Intent intent) {
        String name = intent.getStringExtra("name");
        int courses = intent.getIntExtra("courses", 0);
        Log.i(TAG, msg:"Name: " + name + ", Courses: " + courses);

        Log.i(TAG, msg:"Hello from 1st Receiver, Thread Name: " + Thread.currentThread().getName());
        Toast.makeText(context, text:"Hello from 1st Receiver", Toast.LENGTH_LONG).show();

        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Now let's run our app and see if we can pass data from our main activity to our first receiver so run the app:



Click “Broadcast Now” button and here is the Logcat window once you do the broadcast:

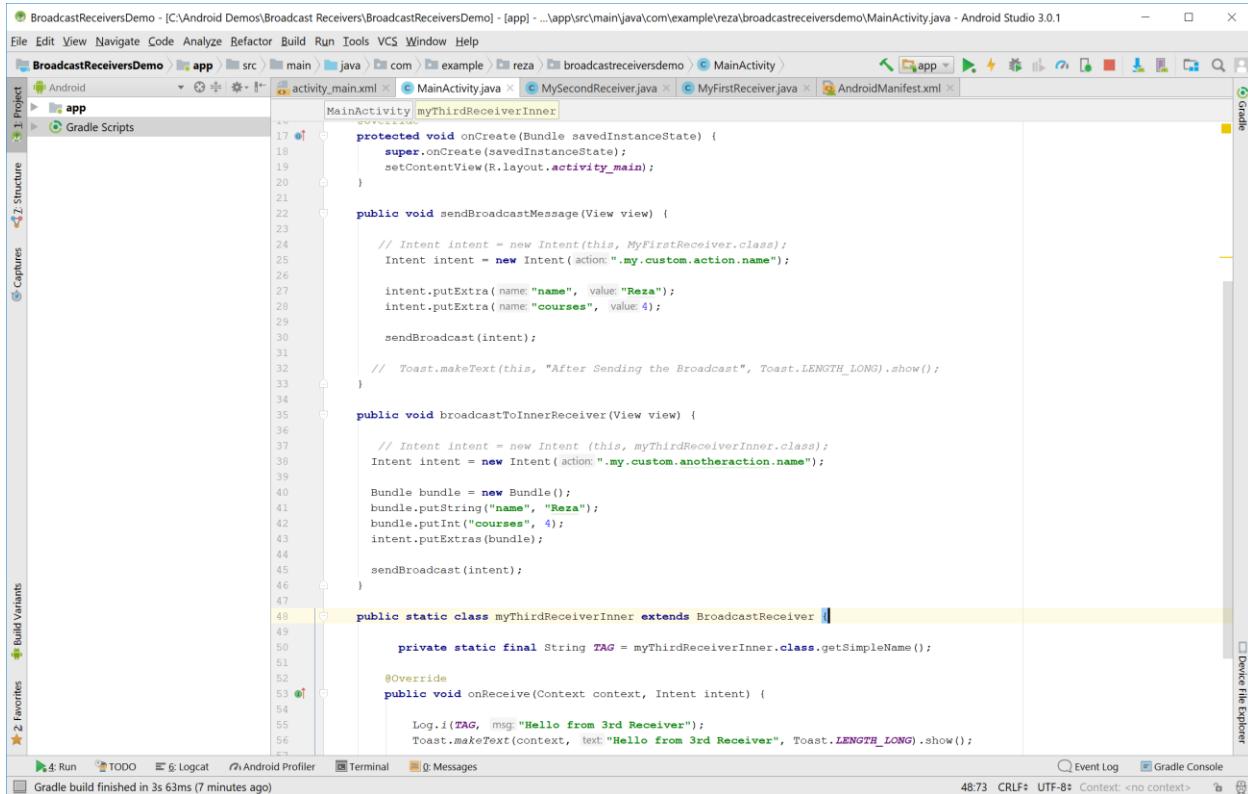
The screenshot shows the Android Studio interface. The top navigation bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, and Help. Below the toolbar, the Project tab is selected, showing the project structure with files like activity\_main.xml, MainActivity.java, MySecondReceiver.java, MyFirstReceiver.java, and AndroidManifest.xml. The Logcat tab is also selected, showing logs for the Emulator Nexus\_4\_API\_25. The logs show two entries:

```
03-24 05:39:34.680 5058-5058/com.example.reza.broadcastreceiversdemo I/MySecondReceiver: Hello from 2nd Receiver
03-24 05:39:34.955 5058-5058/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Name: Reza, Courses: 4
03-24 05:39:34.955 5058-5058/com.example.reza.broadcastreceiversdemo I/MyFirstReceiver: Hello from 1st Receiver, Thread Name: main
```

The bottom status bar indicates a Gradle build finished in 3s 63ms (a minute ago), 110 chars, 2:1 CRLF, UTF-8, and Context: <no context>.

So we were able to send data from our main activity to our first receiver. Now here we used `putExtra()` method which behind the scene used the `Bundle` object.

You can use the `Bundle` object directly and use it to send data. Let's try that to send data to our third receiver so in `MainActivity.java`, modify the method `broadcastToInnnerReceiver()` (which sends data to the third receiver) as shown:



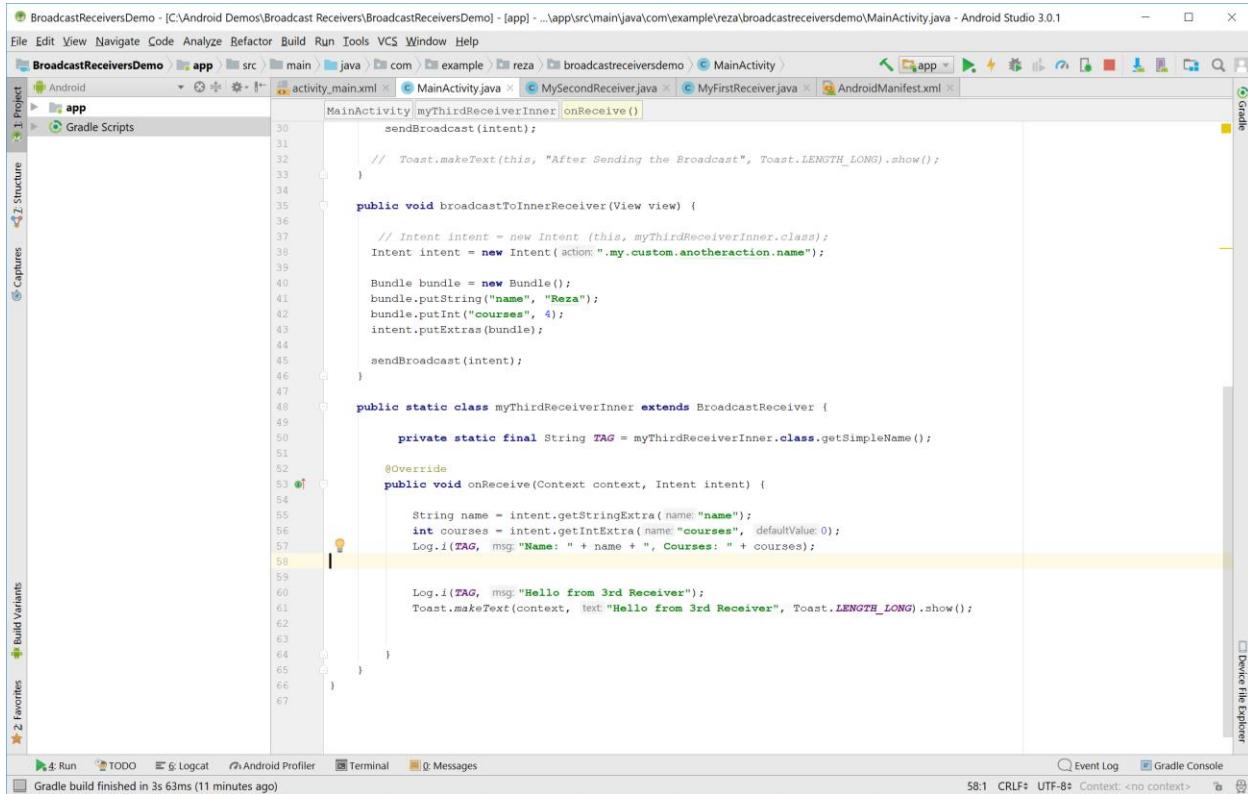
The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays `MainActivity.java`. The `broadcastToInnnerReceiver()` method has been modified to use a `Bundle` instead of `Intent.putExtra()`. The new code is as follows:

```
protected void broadcastToInnnerReceiver(View view) {
    Intent intent = new Intent(action ".my.custom.anotheraction.name");
    Bundle bundle = new Bundle();
    bundle.putString("name", "Reza");
    bundle.putInt("courses", 4);
    intent.putExtras(bundle);
    sendBroadcast(intent);
}
```

The code editor highlights the `intent.putExtras(bundle);` line. The bottom status bar shows 'Gradle build finished in 3s 63ms (7 minutes ago)' and '48/73 CRLF: UTF-8: Context: <no context>'.

Lines 40, 41, 42, and 43 were added.

Since we are sending data to our third receiver which is actually an inner class inside the main activity then we go to that class and modify the code for its `onReceive()` method as shown so we can retrieve the data we just sent to it as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The code editor displays the `MainActivity.java` file. The specific code being modified is:

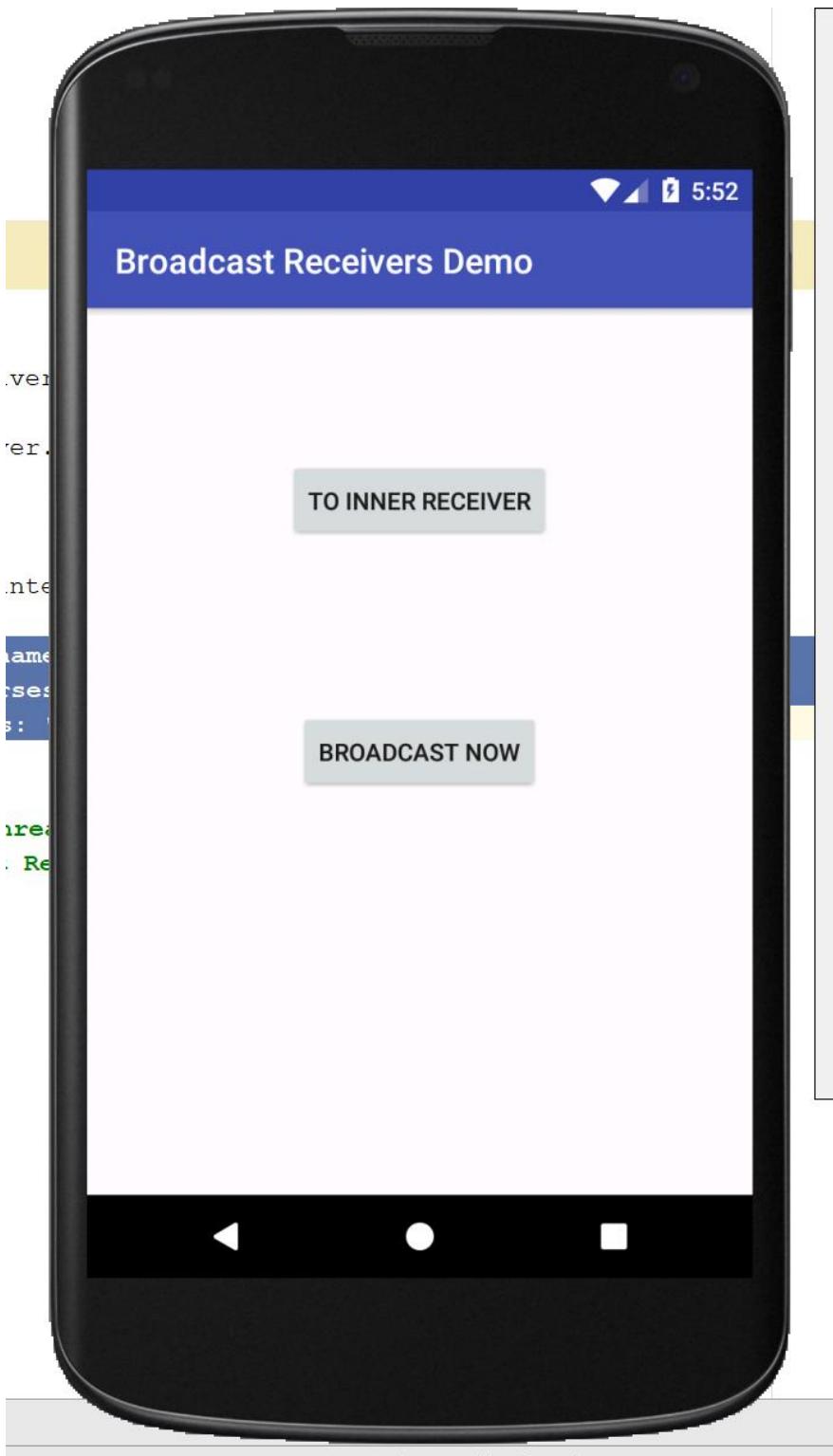
```
    public void broadcastToInnerReceiver(View view) {
        Intent intent = new Intent(action: ".my.custom.anotheraction.name");
        Bundle bundle = new Bundle();
        bundle.putString("name", "Reza");
        bundle.putInt("courses", 4);
        intent.putExtras(bundle);
        sendBroadcast(intent);
    }

    public static class myThirdReceiverInner extends BroadcastReceiver {
        @Override
        public void onReceive(Context context, Intent intent) {
            String name = intent.getStringExtra(name: "name");
            int courses = intent.getIntExtra(name: "courses", defaultValue: 0);
            Log.i(TAG, msg: "Name: " + name + ", Courses: " + courses);

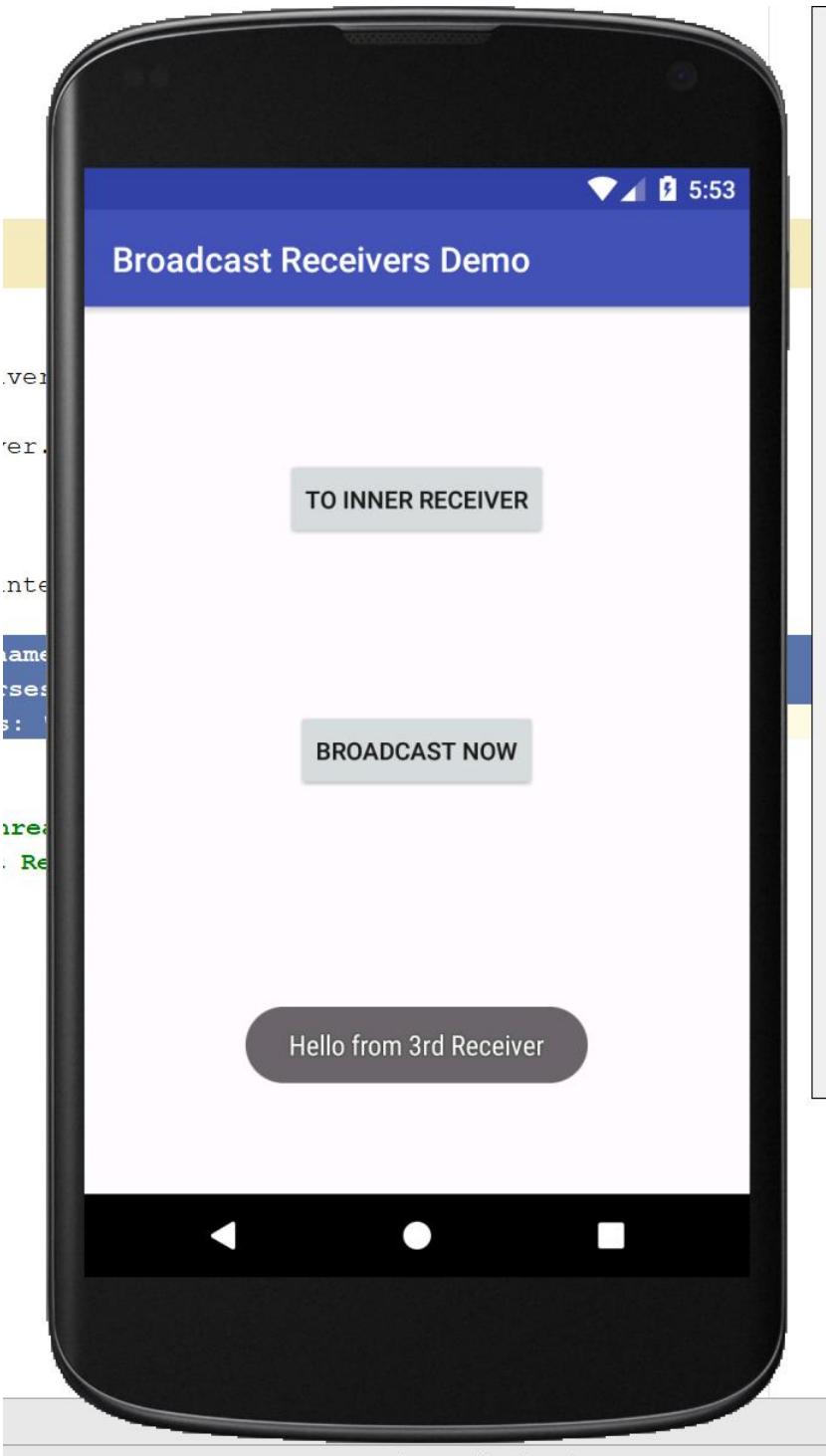
            Log.i(TAG, msg: "Hello from 3rd Receiver");
            Toast.makeText(context, text: "Hello from 3rd Receiver", Toast.LENGTH_LONG).show();
        }
    }
}
```

Lines 55, 56, and 57 were added. Note that they are identical to lines 21, 22, and 23 which we added to the `onReceive()` method of our first receiver.

Now let's run the app to test our newly added code.



Click “To Inner Receiver” button to send broadcast to the third receiver:



The screenshot shows the Android Studio interface. The top navigation bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, and Help. Below the toolbar, the project structure is visible with files like activity\_main.xml, MainActivity.java, MySecondReceiver.java, MyFirstReceiver.java, and AndroidManifest.xml. The main code editor displays MyFirstReceiver.java:

```
2 import android.content.BroadcastReceiver;
3 import android.content.Context;
4 import android.content.Intent;
5 import android.util.Log;
6 import android.widget.Toast;
7
8 /**
9  * Created by Reza on 2018-03-23.
10 */
11
12 public class MyFirstReceiver extends BroadcastReceiver {
13
14     private static final String TAG = MyFirstReceiver.class.getSimpleName();
15
16     @Override
17     public void onReceive(Context context, Intent intent) {
18
19         String name = intent.getStringExtra("name");
20         int courses = intent.getIntExtra("courses", defaultValue 0);
21         Log.i(TAG, msg "Name: " + name + ", Courses: " + courses);
22
23     }
24
25
26 }
```

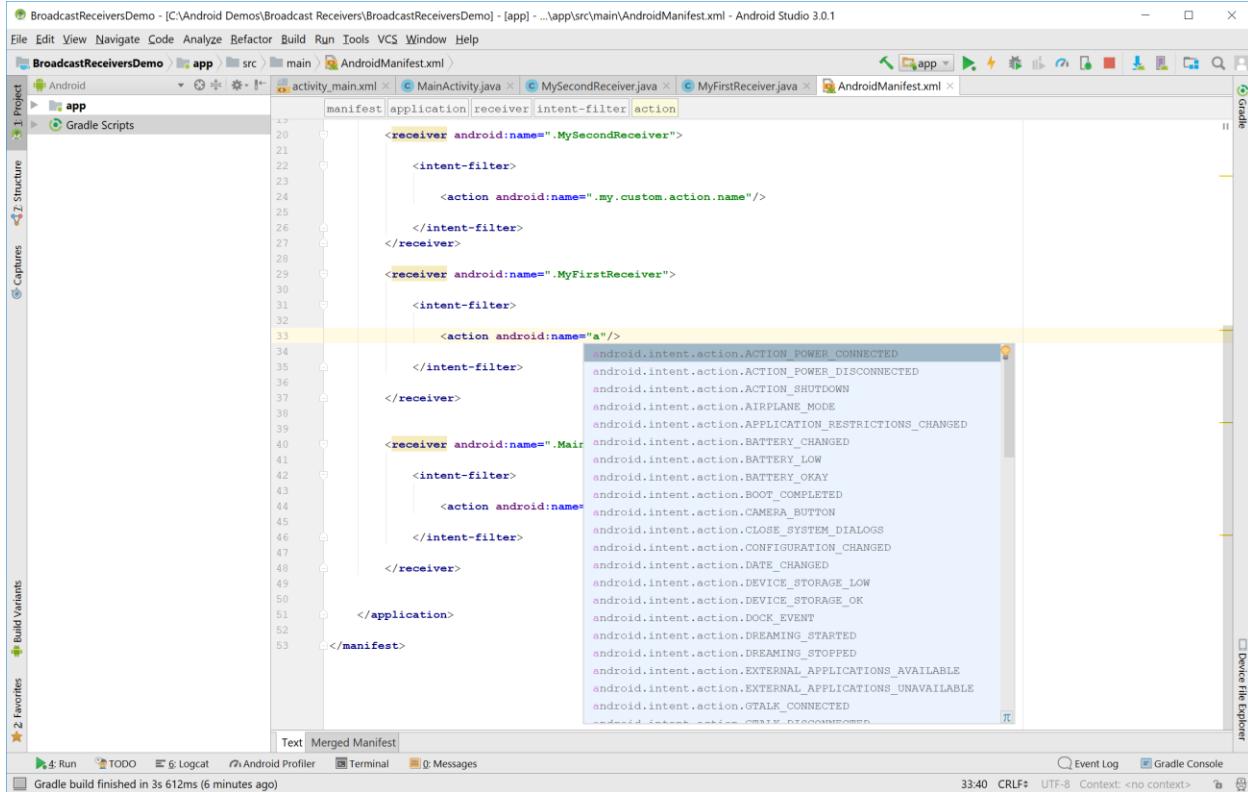
The Logcat tab at the bottom shows the following output:

```
Emulator Nexus_4 API_25 Android 7.1.1, API 25 com.example.reza.broadcastreceiversdemo (5273)
03-24 05:53:30.315 5273-5273/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Name: Reza, Courses: 4
03-24 05:53:30.315 5273-5273/com.example.reza.broadcastreceiversdemo I/myThirdReceiverInner: Hello from 3rd Receiver
```

So once again we were able to send data from our activity to our receiver (third receiver here).

## A few more practical examples:

In your manifest file remove the current action from the action tag and type “a” and you will see a list of many predefined actions in Android as shown:



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'AndroidManifest.xml' file is selected in the editor. A code completion dropdown is displayed over the word 'action' in the XML code, listing various predefined Intent actions such as ACTION\_POWER\_CONNECTED, ACTION\_POWER\_DISCONNECTED, ACTION\_SHUTDOWN, etc.

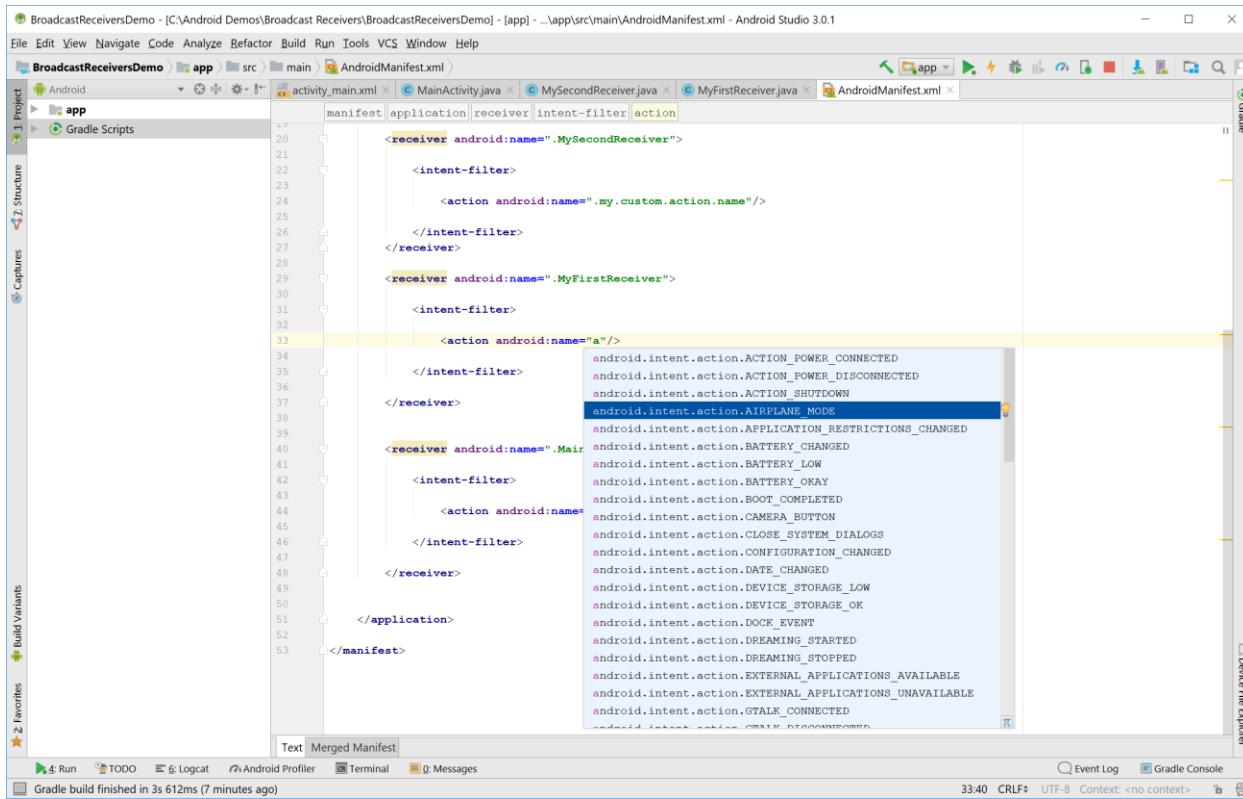
```
<receiver android:name=".MySecondReceiver">
    <intent-filter>
        <action android:name=".my.custom.action.name"/>
    </intent-filter>
</receiver>

<receiver android:name=".MyFirstReceiver">
    <intent-filter>
        <action android:name="a"/>
    </intent-filter>
</receiver>

<receiver android:name=".MainReceiver">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
    </intent-filter>
</receiver>

</application>
</manifest>
```

Pick the action airplane mode as shown:

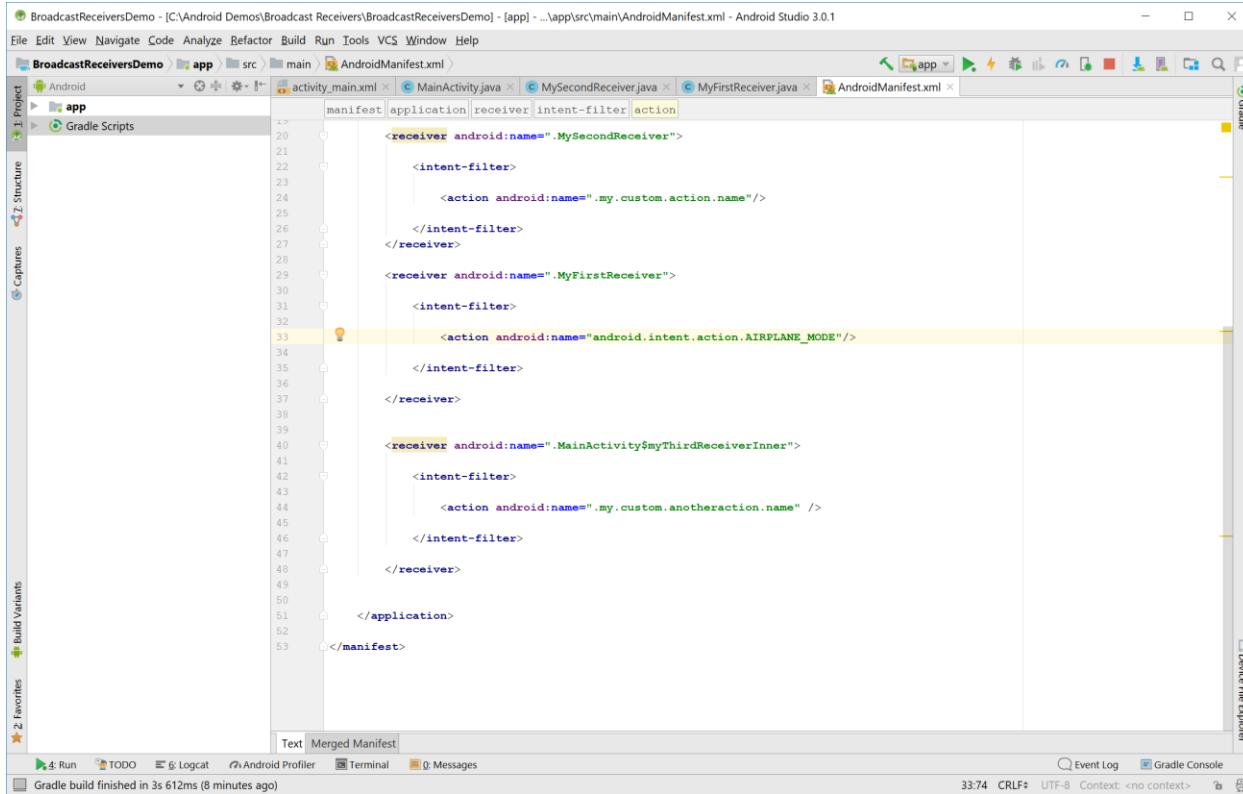


The screenshot shows the AndroidManifest.xml file in the Android Studio editor. The cursor is positioned over the 'action' attribute of a receiver element. A code completion dropdown is open, listing various Android intent actions. The 'android.intent.action.AIRPLANE\_MODE' option is highlighted.

```
<receiver android:name=".MySecondReceiver">
    <intent-filter>
        <action android:name=".my.custom.action.name"/>
    </intent-filter>
</receiver>

<receiver android:name=".MyFirstReceiver">
    <intent-filter>
        <action android:name="a"/>
    </intent-filter>
</receiver>

<receiver android:name=".MainActivity$MyThirdReceiverInner">
    <intent-filter>
        <action android:name="android.intent.action.AIRPLANE_MODE"/>
    </intent-filter>
</receiver>
```



The screenshot shows the same AndroidManifest.xml file in the Android Studio editor. The 'action android:name="android.intent.action.AIRPLANE\_MODE"' line has been selected, indicated by a yellow highlight and a small orange lightbulb icon.

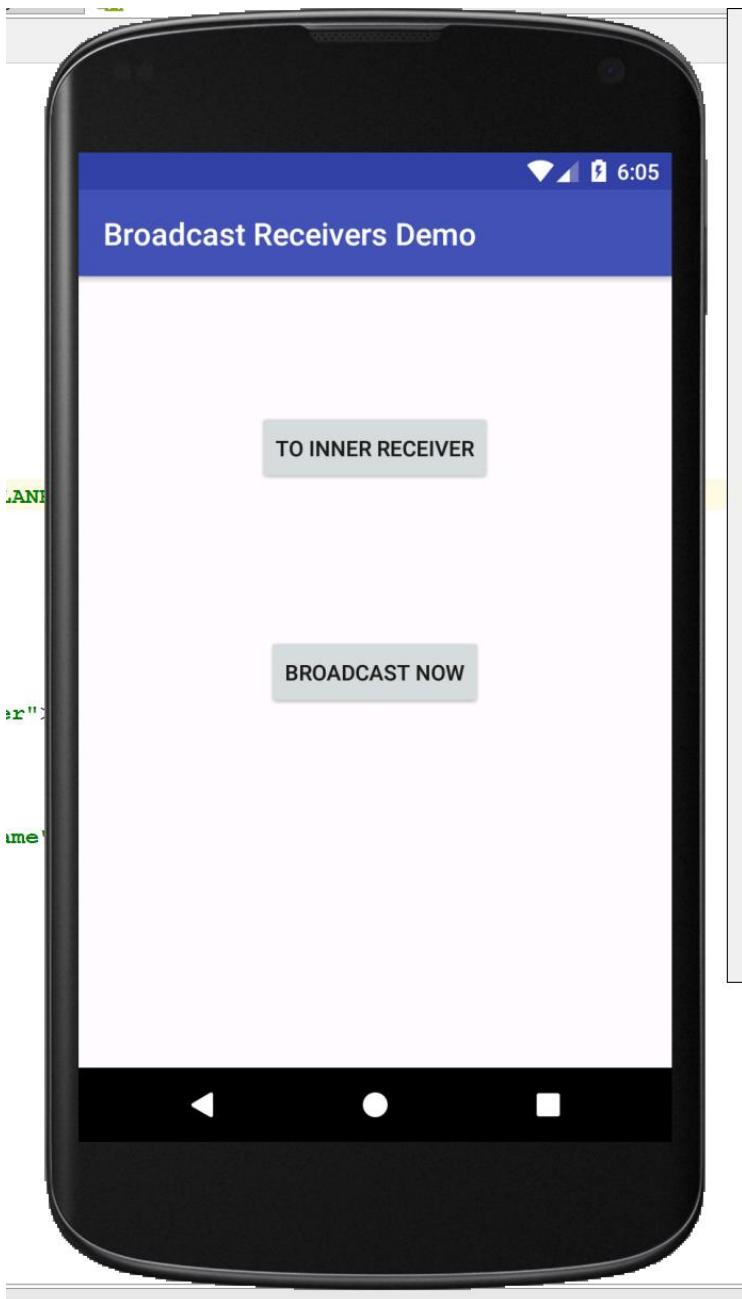
```
<receiver android:name=".MySecondReceiver">
    <intent-filter>
        <action android:name=".my.custom.action.name"/>
    </intent-filter>
</receiver>

<receiver android:name=".MyFirstReceiver">
    <intent-filter>
        <action android:name="a"/>
    </intent-filter>
</receiver>

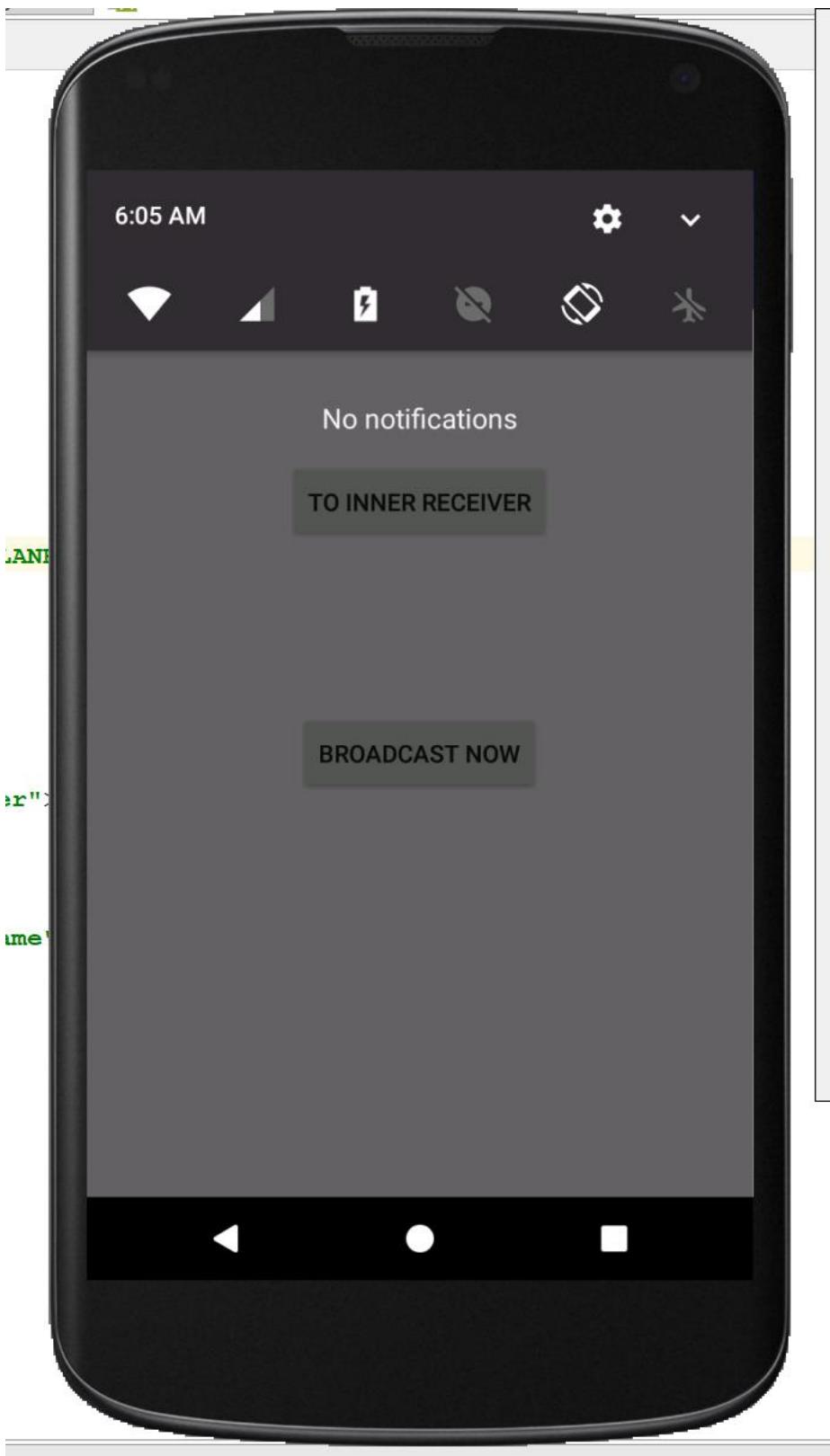
<receiver android:name=".MainActivity$MyThirdReceiverInner">
    <intent-filter>
        <action android:name="android.intent.action.AIRPLANE_MODE"/>
    </intent-filter>
</receiver>
```

So now our first receiver is registered with the air plane mode action. So anytime you turn on the flight mode of your device, then the Android operating system will broadcast a message to all the applications that the flight mode has been turned on or it has been turned off if you turn it off and any broadcast receiver which is registered for this action will act accordingly.

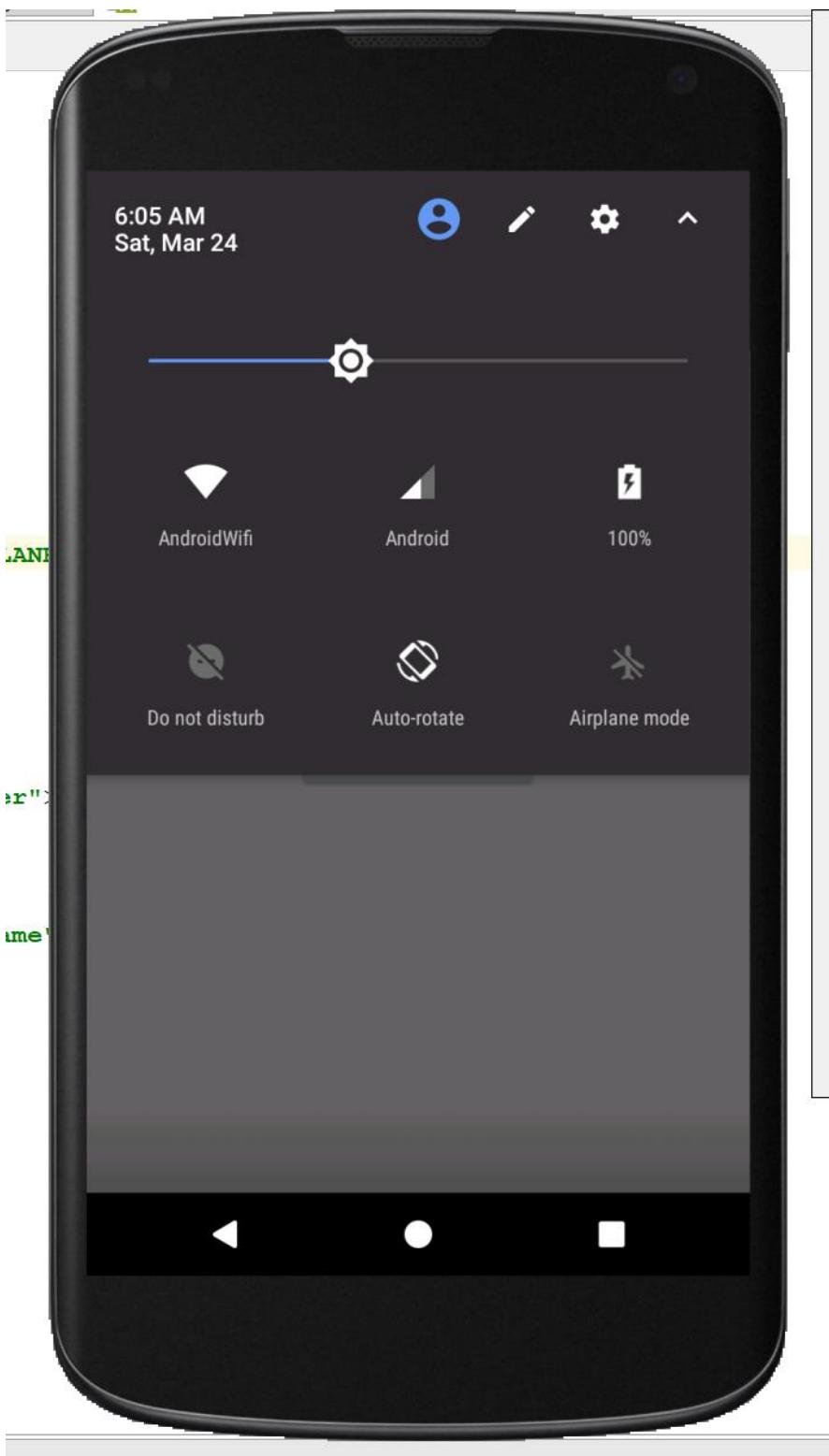
Now let's run our app to test this.



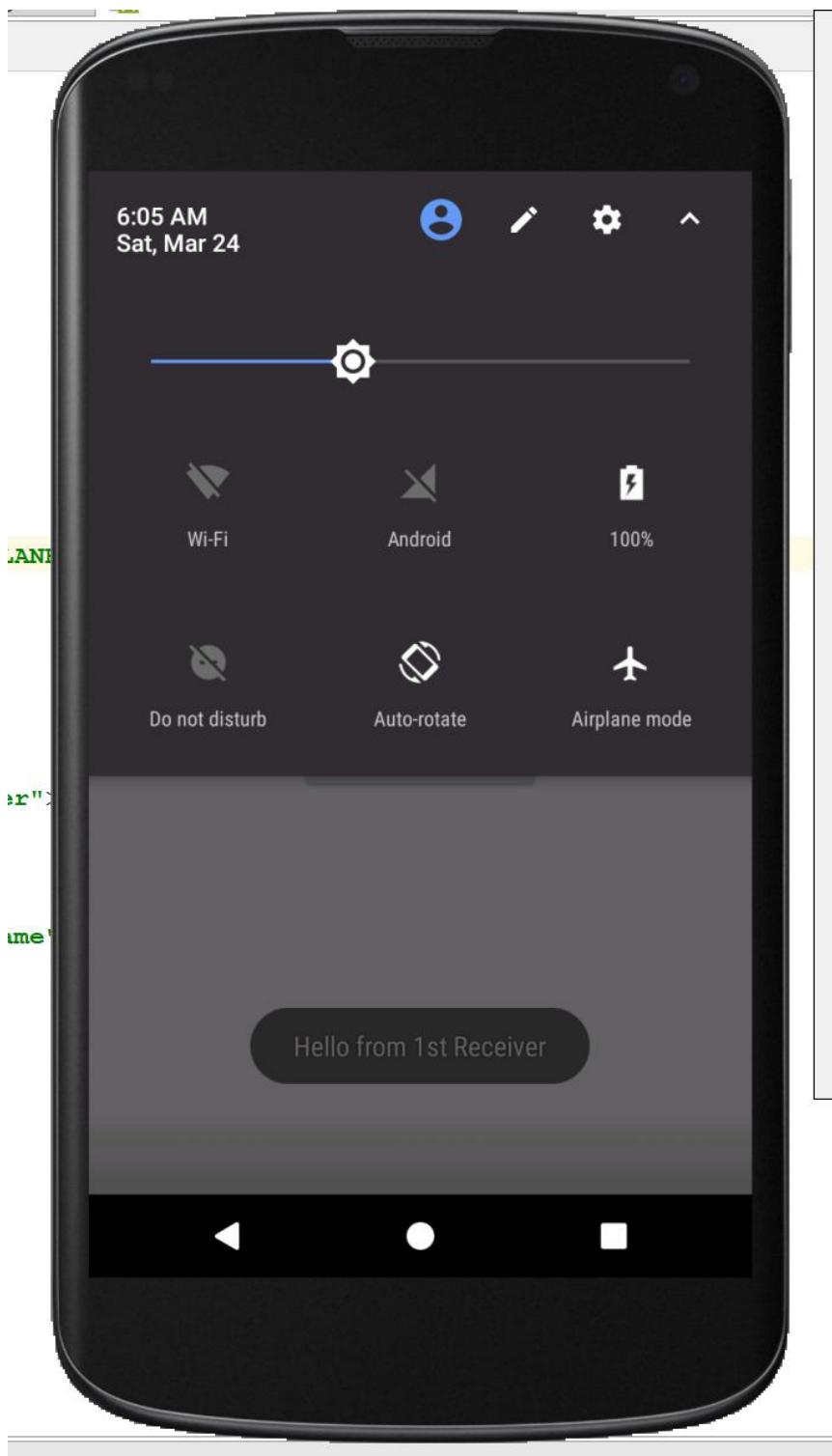
Pull down the notification panel:



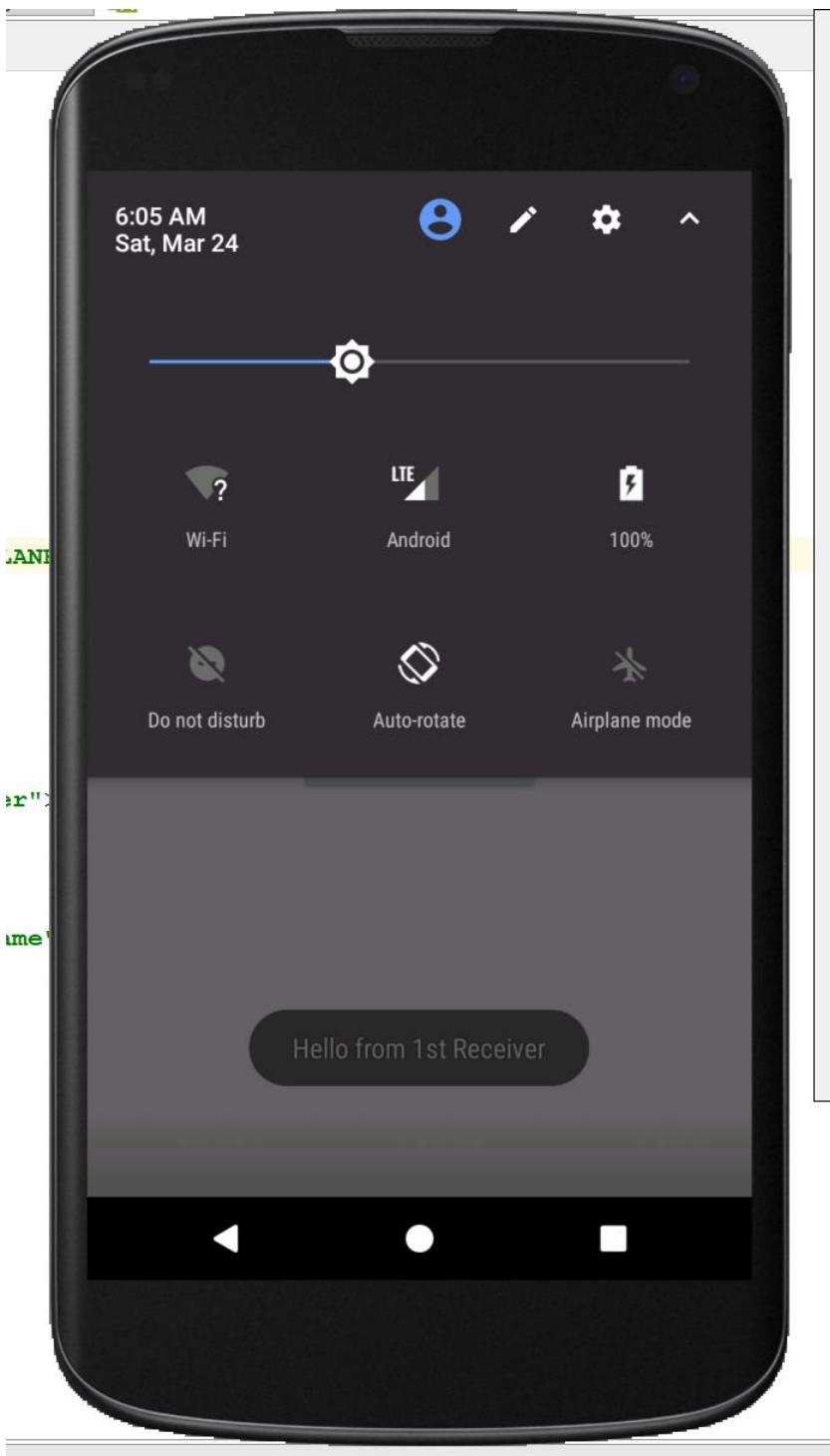
Click on the down arrow:



Click on the “Airplane mode” to turn it on and you see your first receiver responds as shown:

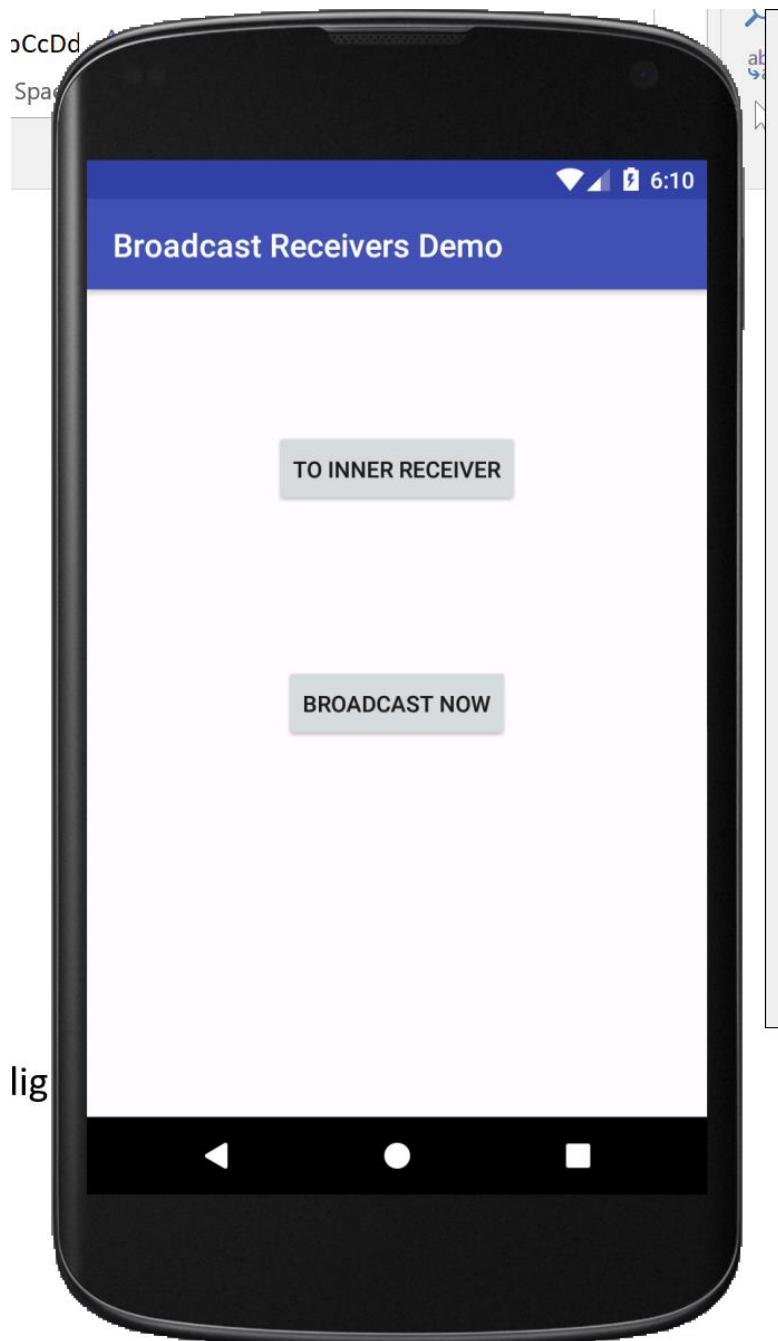


Now click on the “Airplane mode” again to turn the flight mode off and again you have a response from your first receiver as shown:

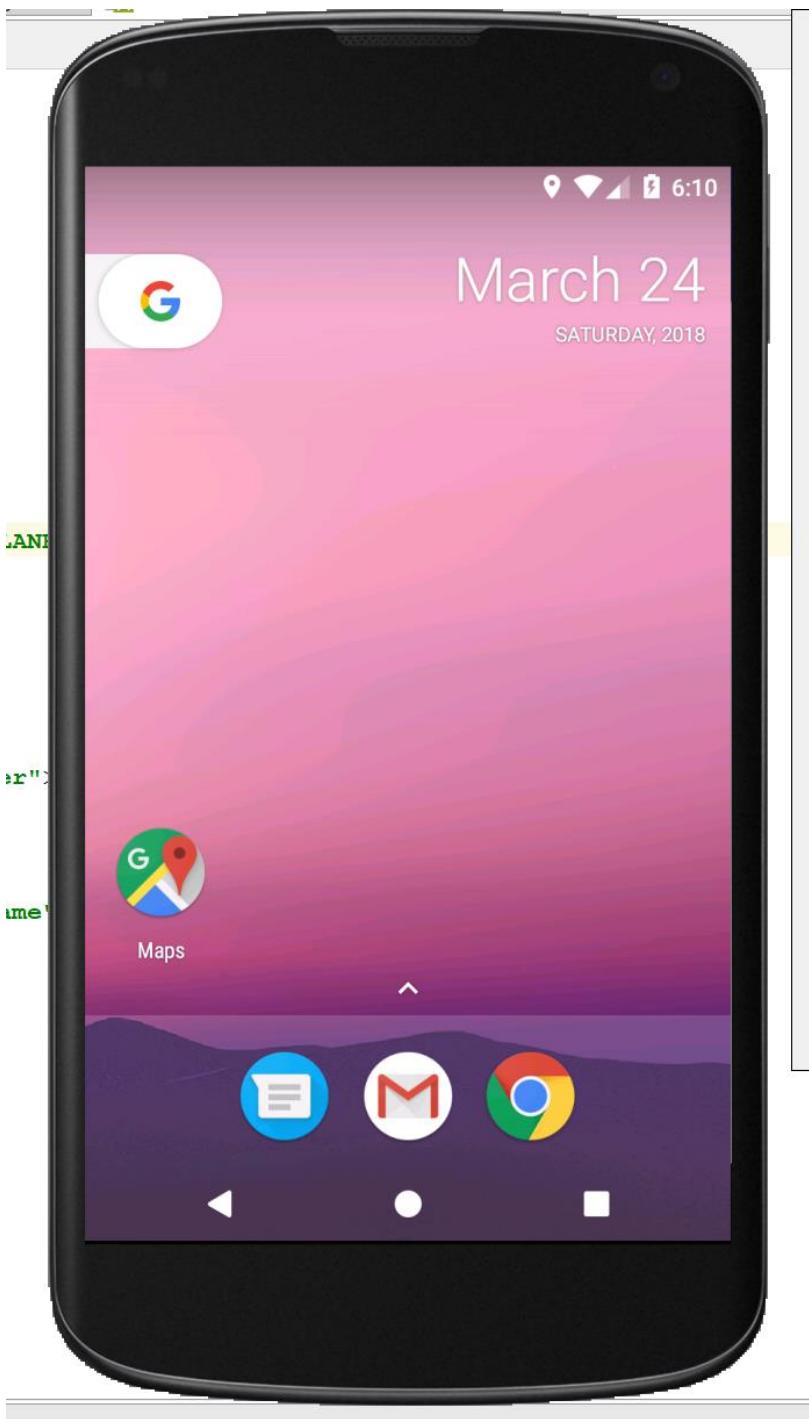


Now note that even if you shut down your application and then turn the flight mode on or off, your receiver still receives the broadcast as shown.

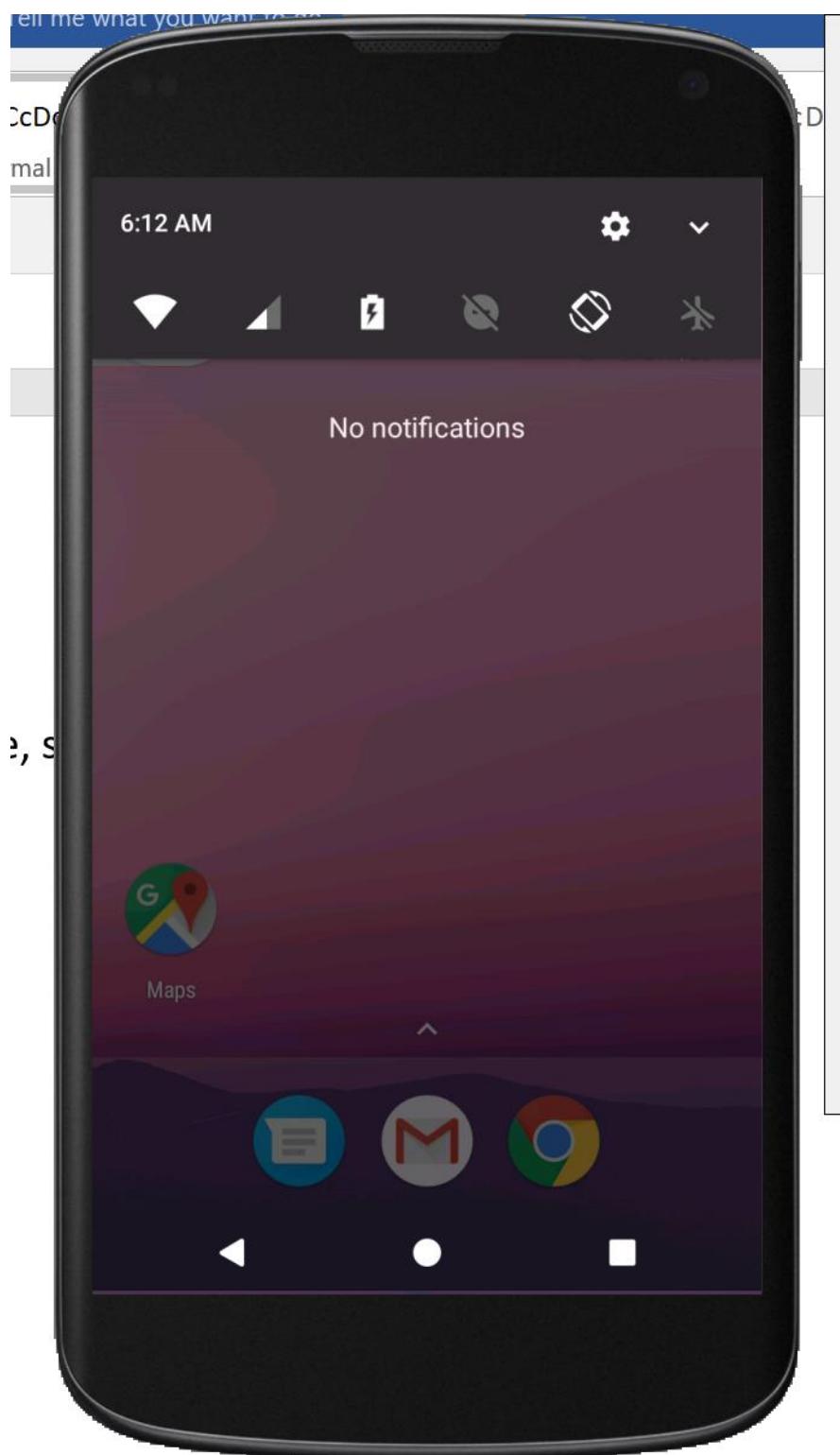
Run the app:

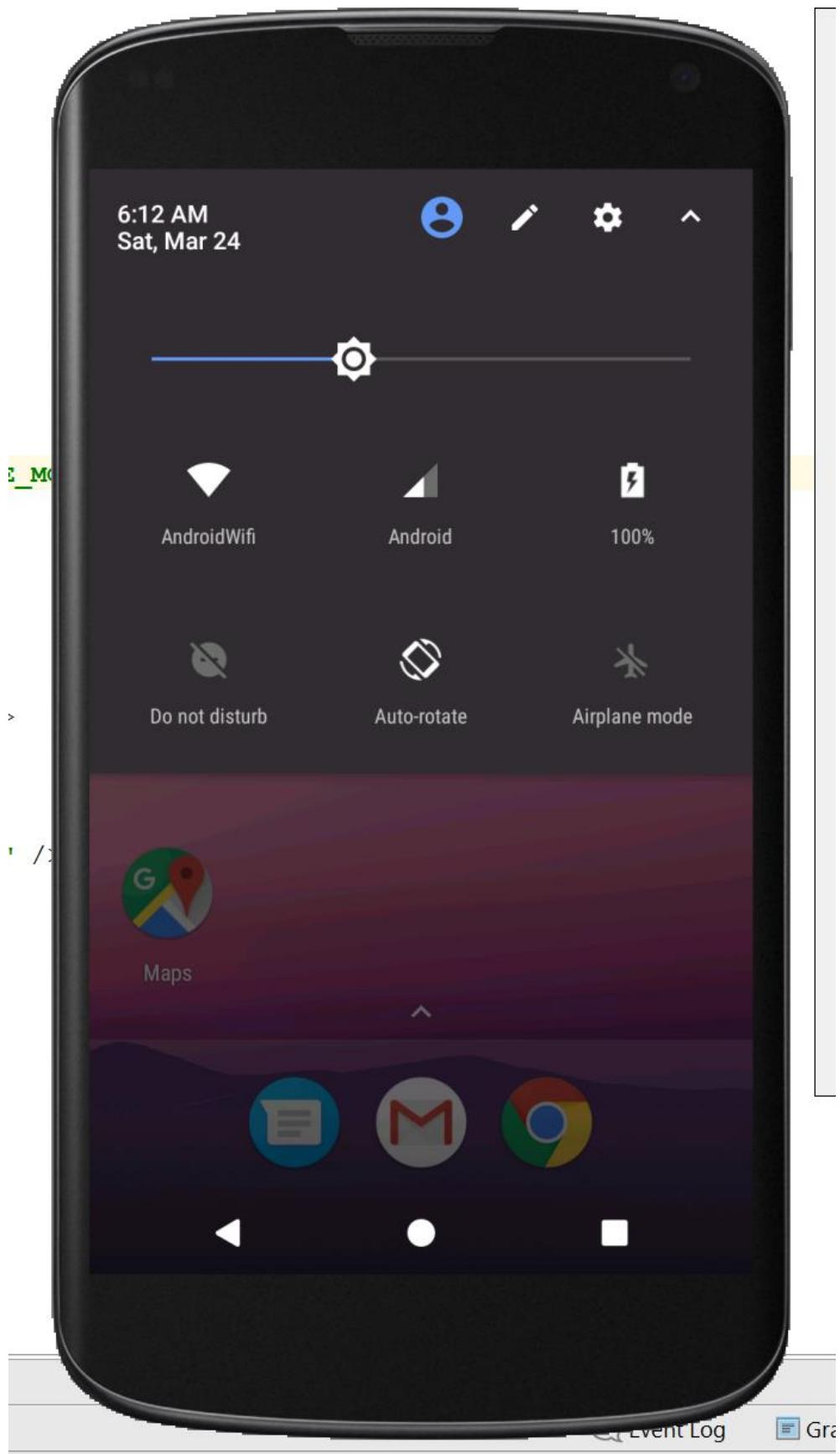


Click on the left arrow at the bottom to shut down the application:

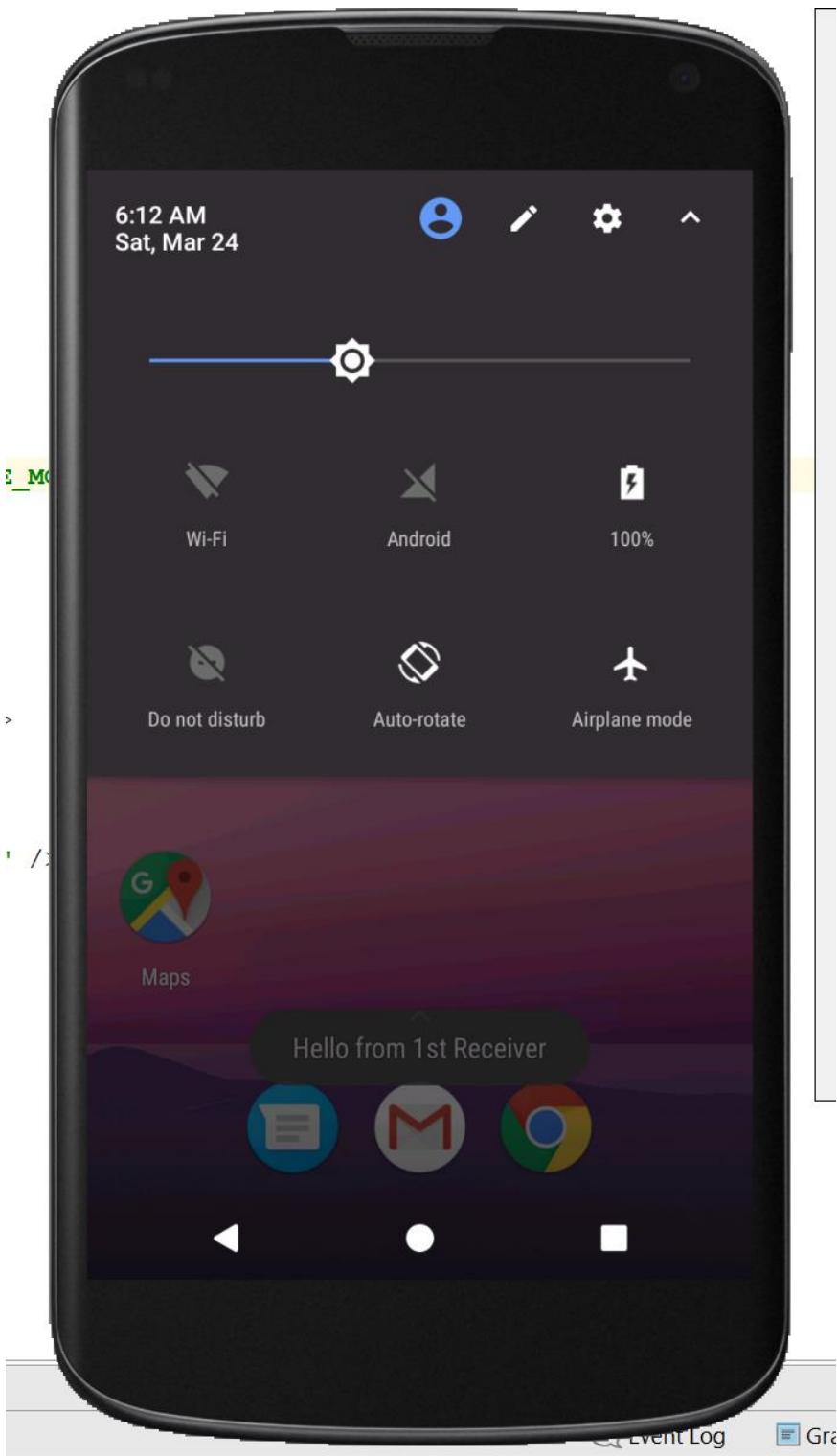


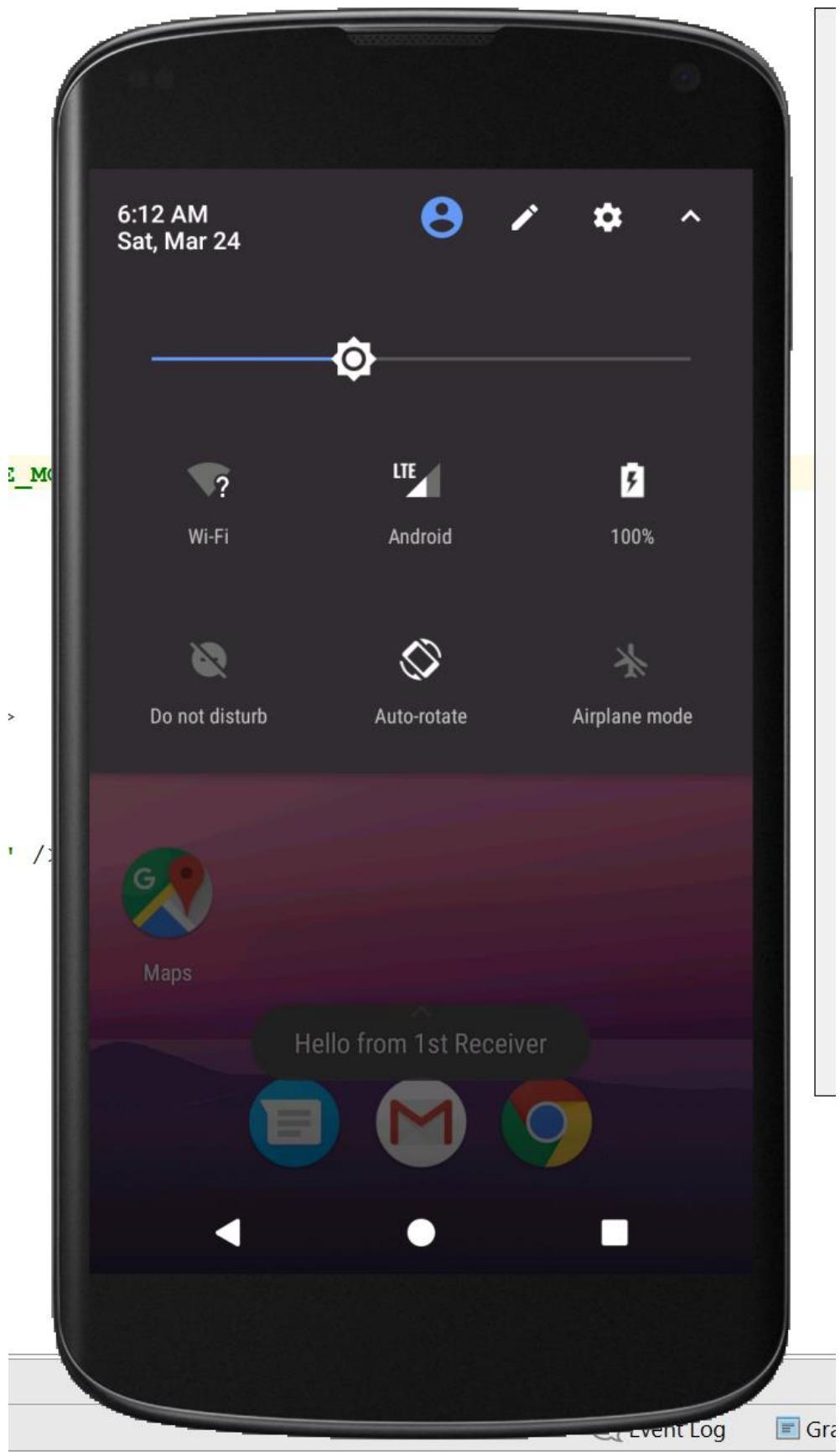
Now pull down your notification panel:





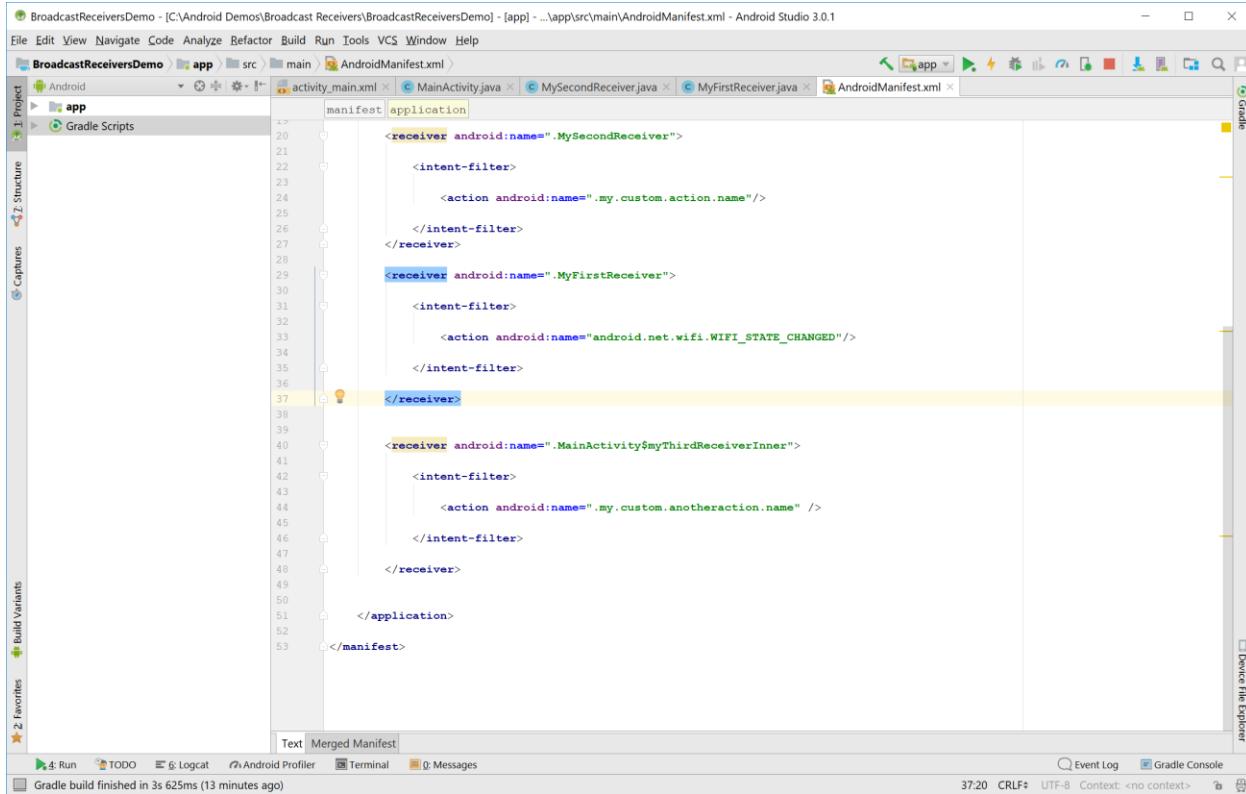
and turn on the flight mode, you will still get the response from your first receiver even if the app is not running as shown:





So even if your application is not open and running, your application is still going to respond to any event to which your broadcast receiver has been registered.

Now let's try a different event. Change the manifest file as shown (line 33):



The screenshot shows the Android Studio interface with the project 'BroadcastReceiversDemo' open. The 'AndroidManifest.xml' file is selected in the editor. The code in the manifest file is as follows:

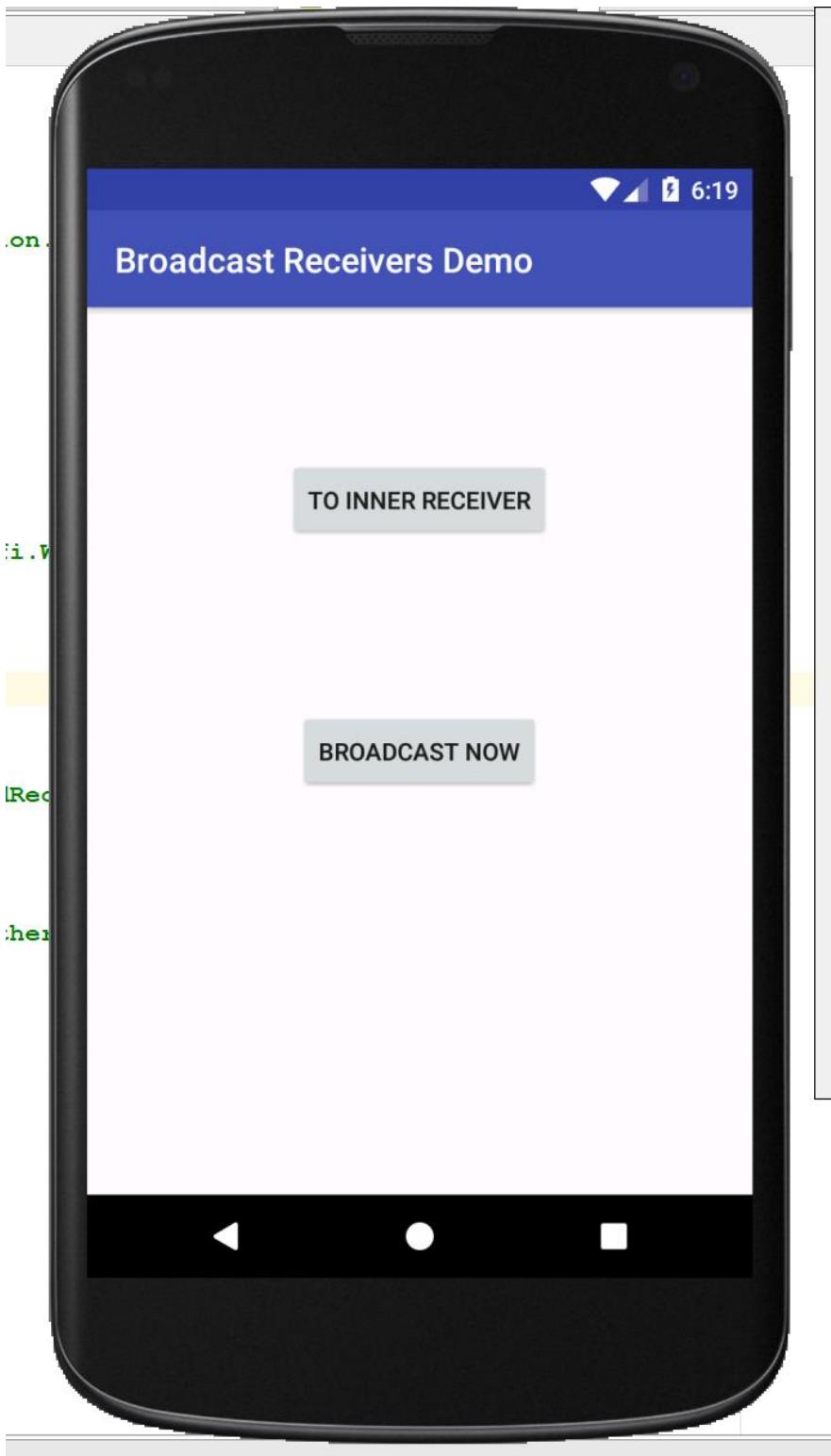
```
<manifest>
    <application>
        <receiver android:name=".MySecondReceiver">
            <intent-filter>
                <action android:name=".my.custom.action.name"/>
            </intent-filter>
        </receiver>

        <receiver android:name=".MyFirstReceiver">
            <intent-filter>
                <action android:name="android.net.wifi.WIFI_STATE_CHANGED"/>
            </intent-filter>
        </receiver>

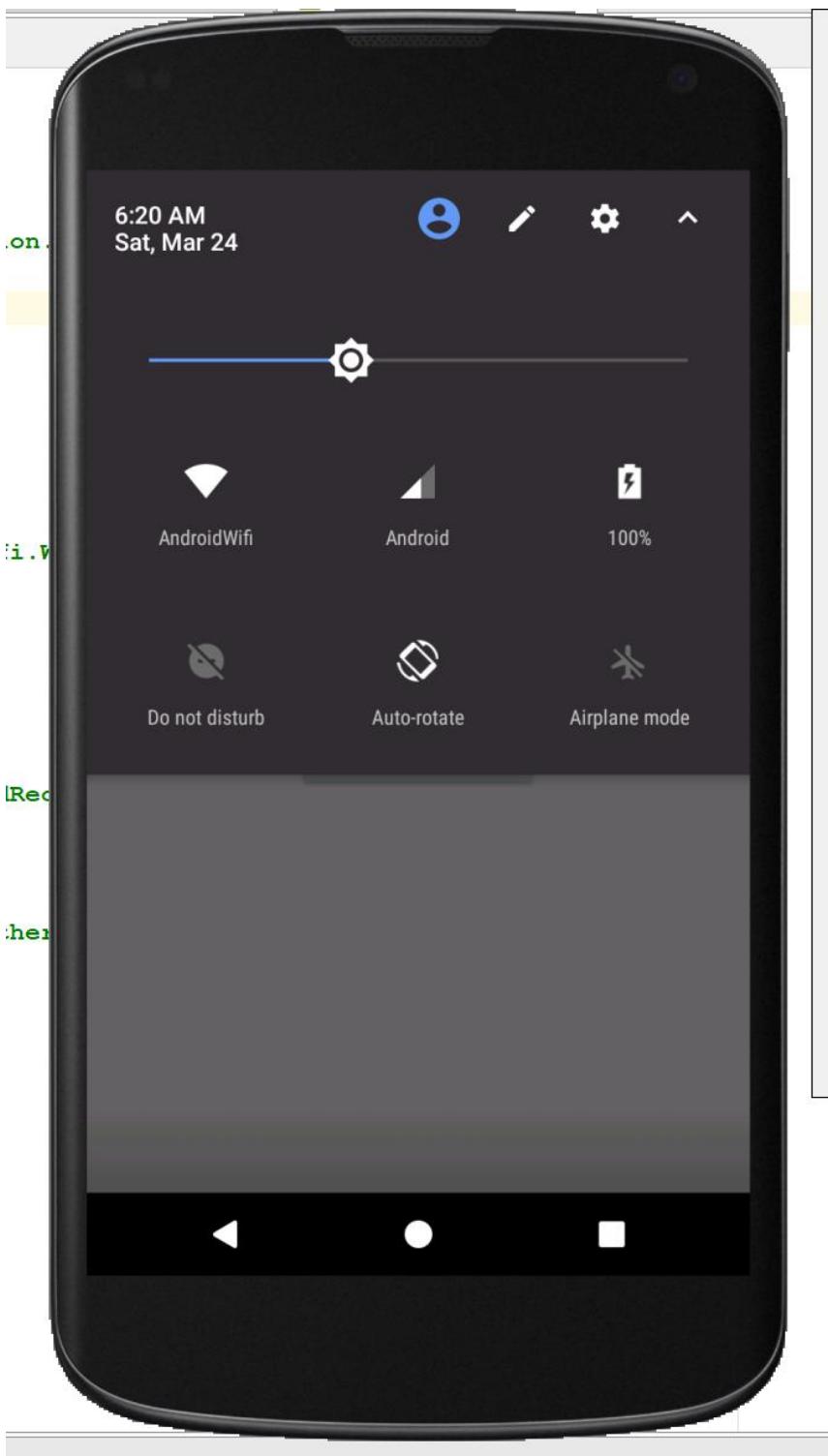
        <receiver android:name=".MainActivity$MyThirdReceiverInner">
            <intent-filter>
                <action android:name=".my.custom.anotheraction.name" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

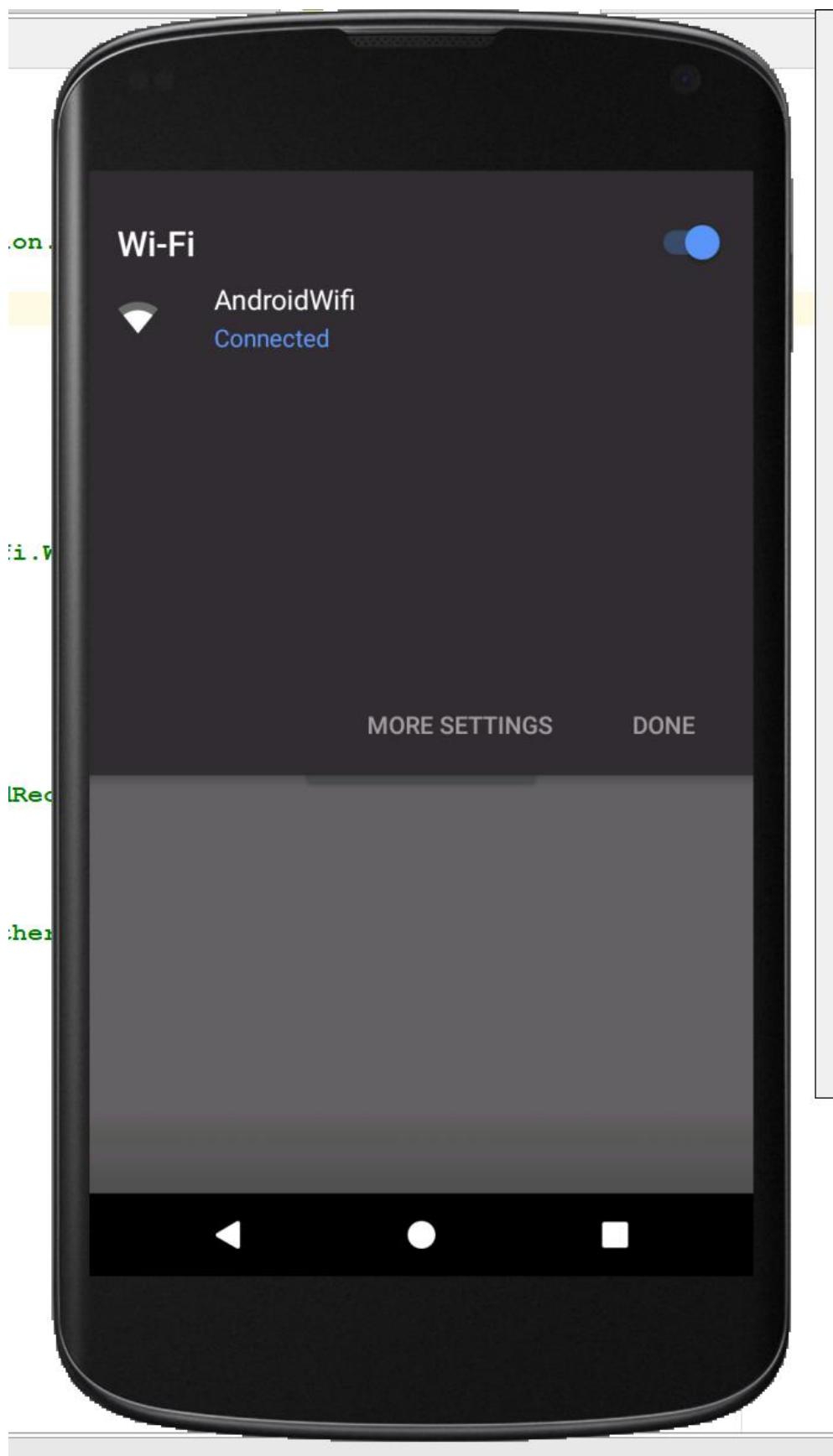
The line 33, which contains the action filter for the MyFirstReceiver, is highlighted with a yellow background. The bottom status bar indicates a successful Gradle build.

Now run the app to check if our receiver responds to the change in the state of Wi-Fi as shown:

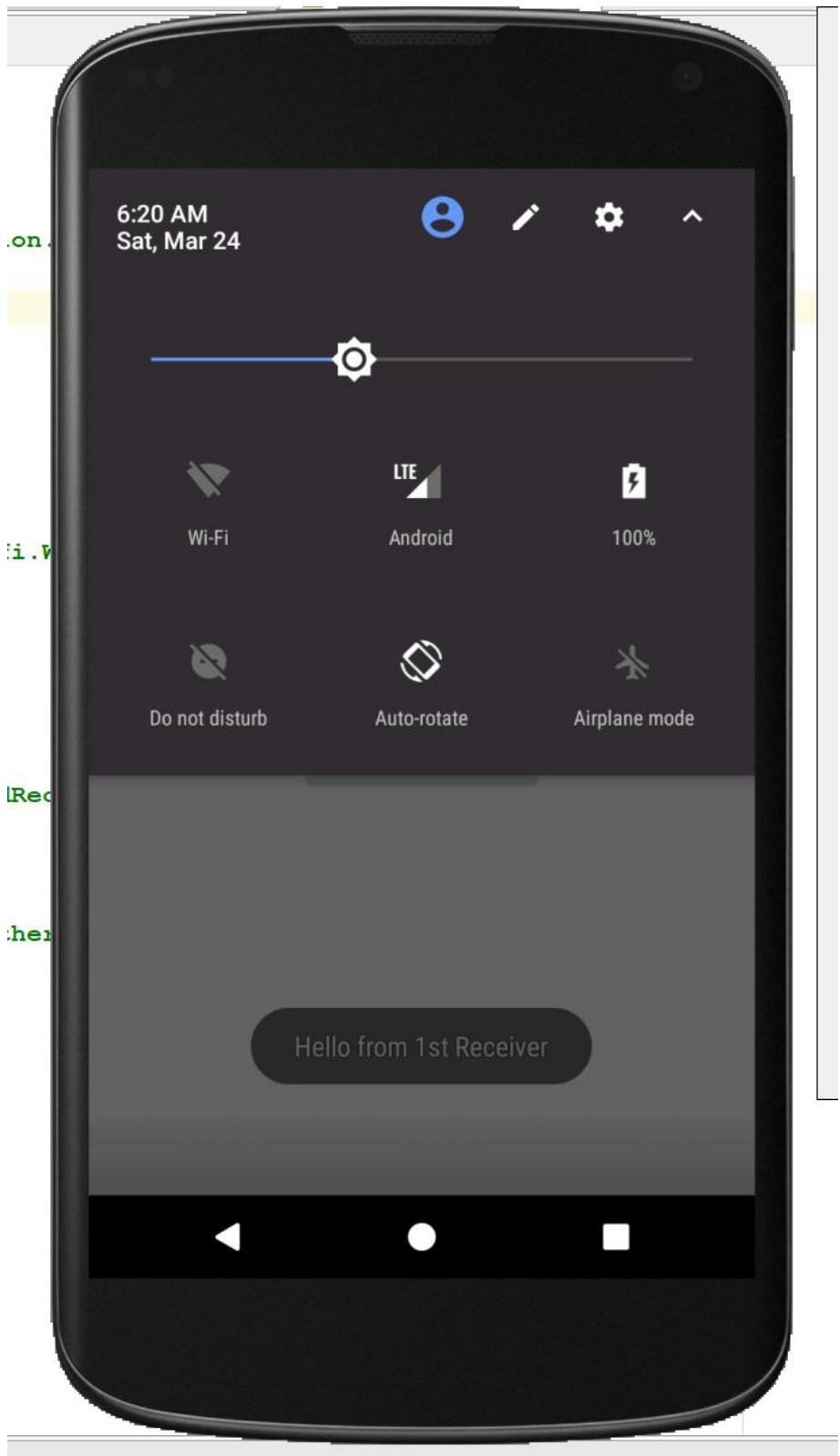


Pull down the menu:

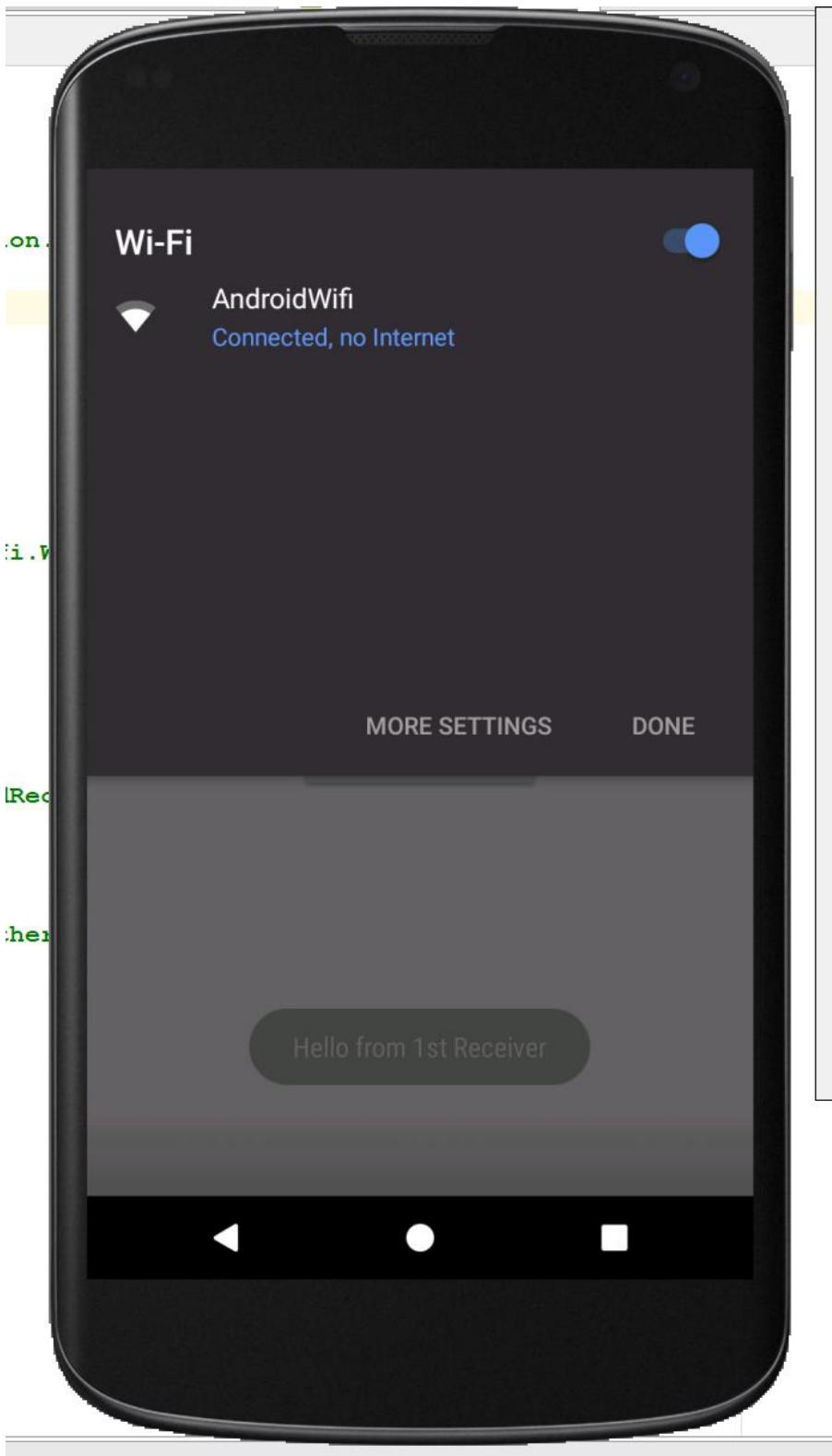


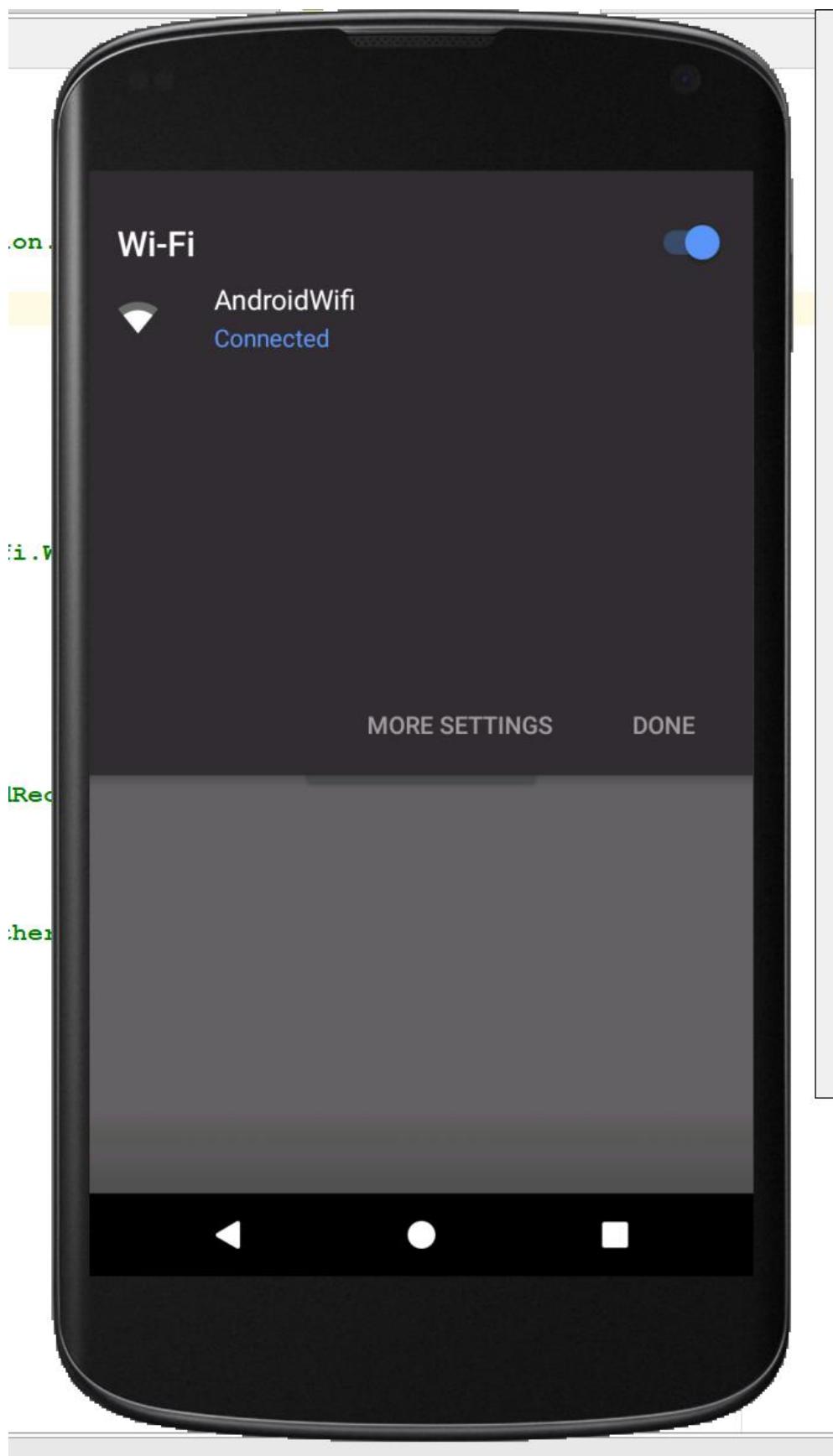


Turn off the Wi-Fi and your receiver responds to the event as shown:

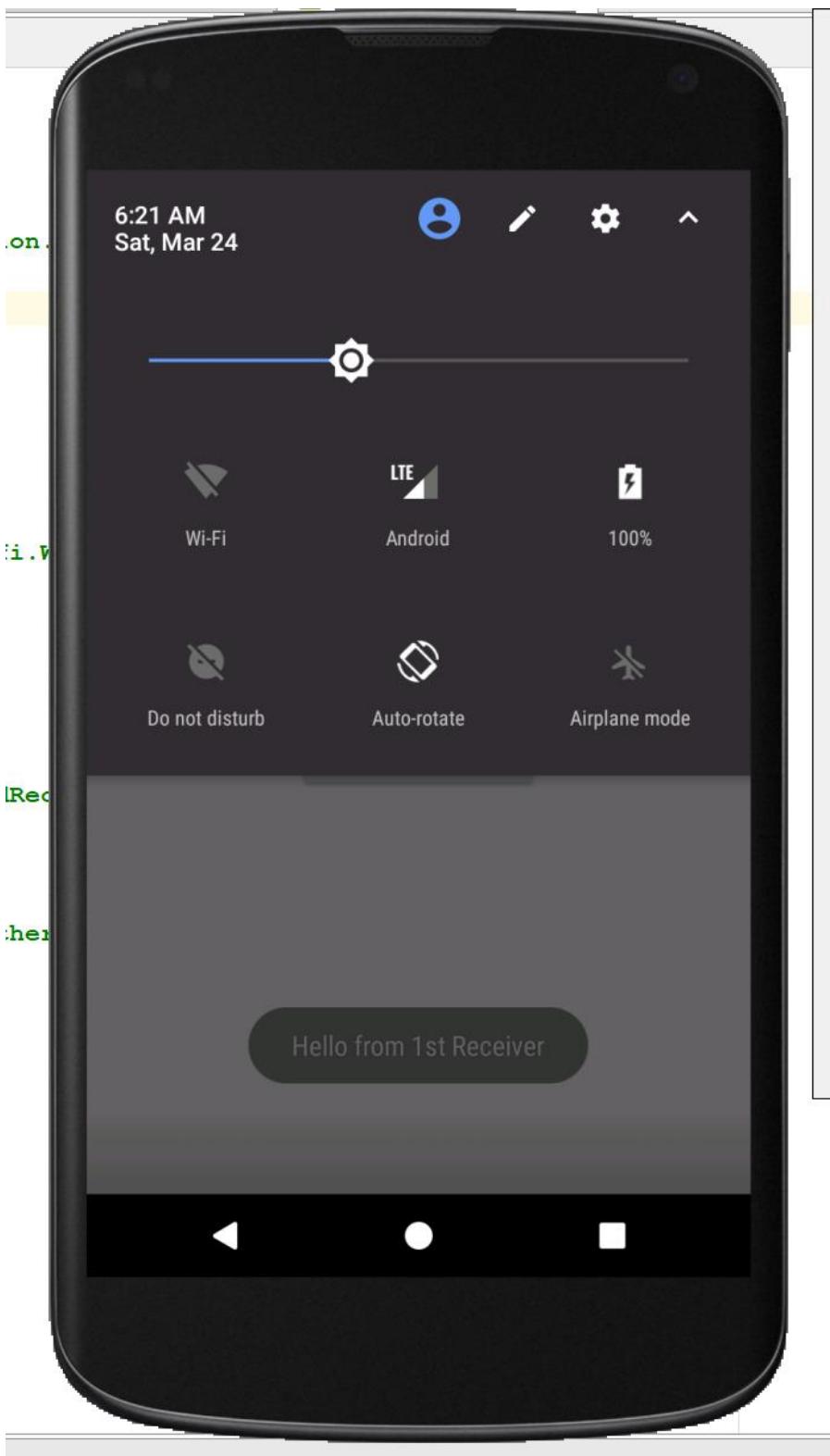


Turn it on again and again your receiver responds to the change in the state of Wi-Fi as shown:

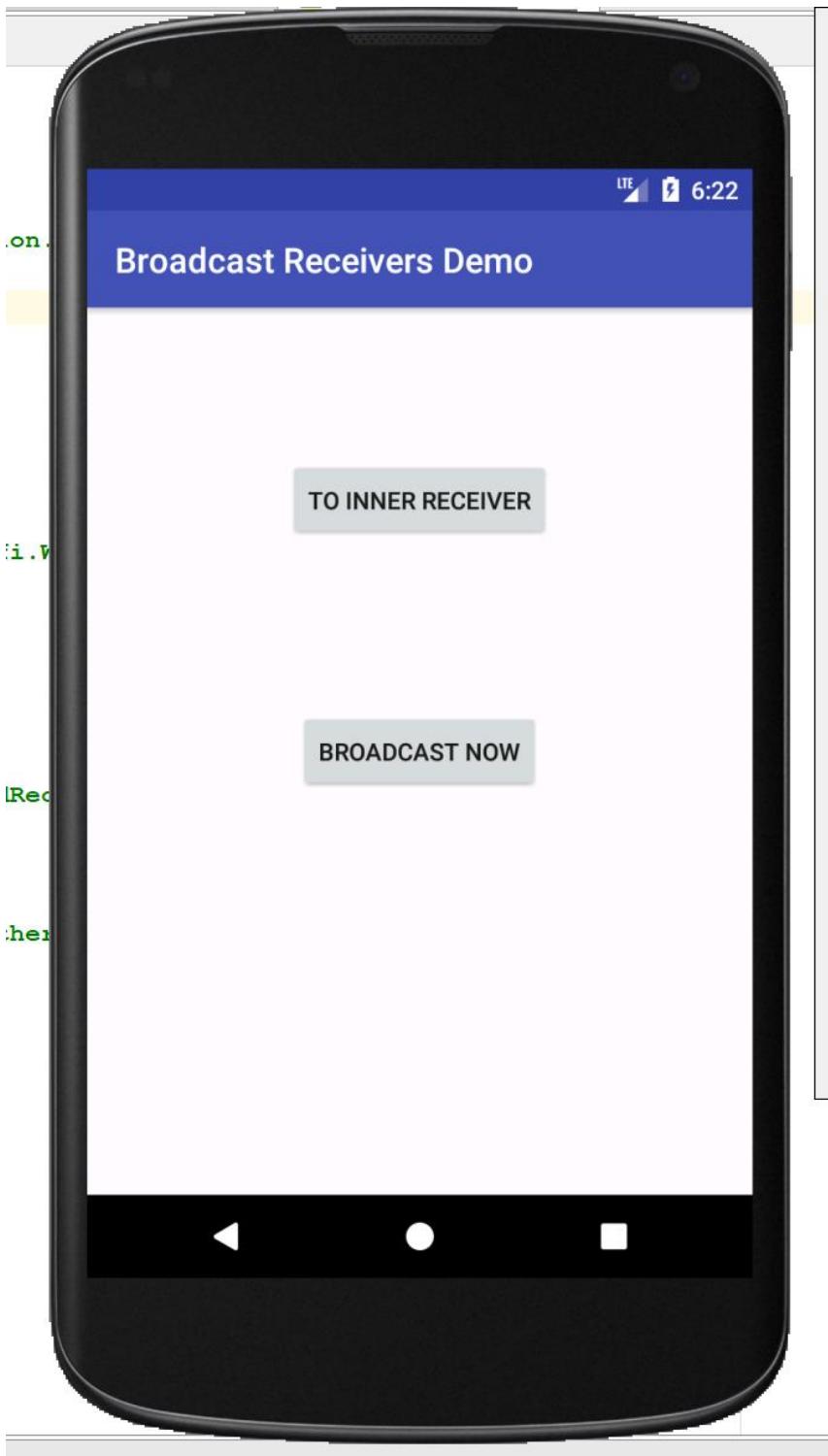




Turn it off again and another response from the receiver:



Now in your app:



Close the app by clicking on the left arrow and try to turn on and off the Wi-Fi and you'll see our first receiver still responds to the change in the state of Wi-Fi, another word to the action it is registered for even if the app is not running.



