CSE2MAD LAB 4 – BROADCAST RECEIVERS

AIMS

Today we will me making 2 apps, one to Broadcast a message and another to receive the broadcast.

STEP 1 – MAKING THE BROADCAST APPLICATION UI

We are to send a custom broadcast to another app and handle the broadcast via a Context-registered receiver.

a) Create a new project & choose the empty activity template. (See week 1 lab)
b) Create a new Android Project using API 25: Android 7.1.1 (Nougat)
c) Create the UI in the visual DESIGN view for the main activity as in Figure 1. If you need a hint see the component tree in Figure 2 and layout code from previous labs.

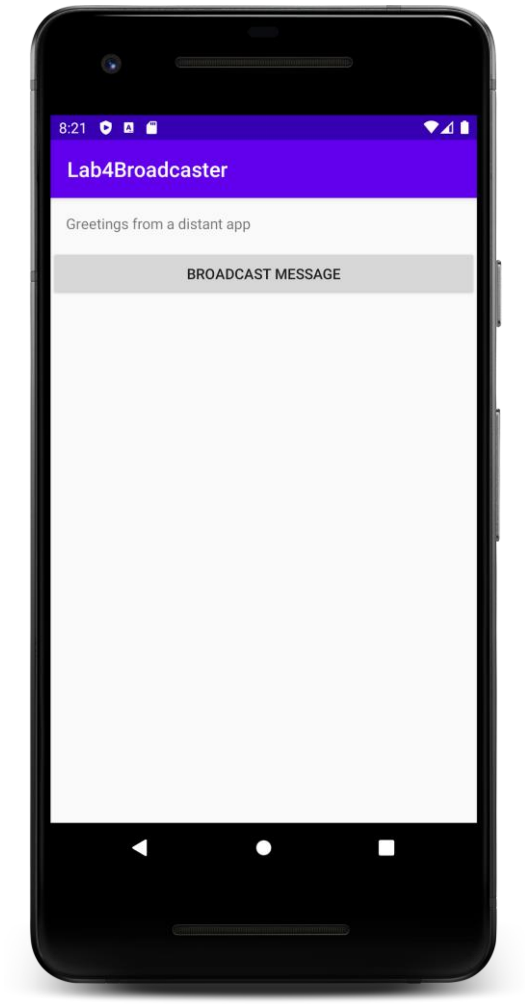
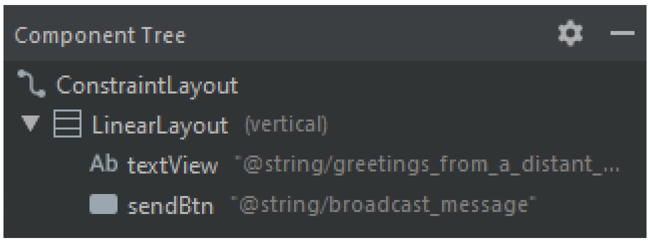


Figure 1

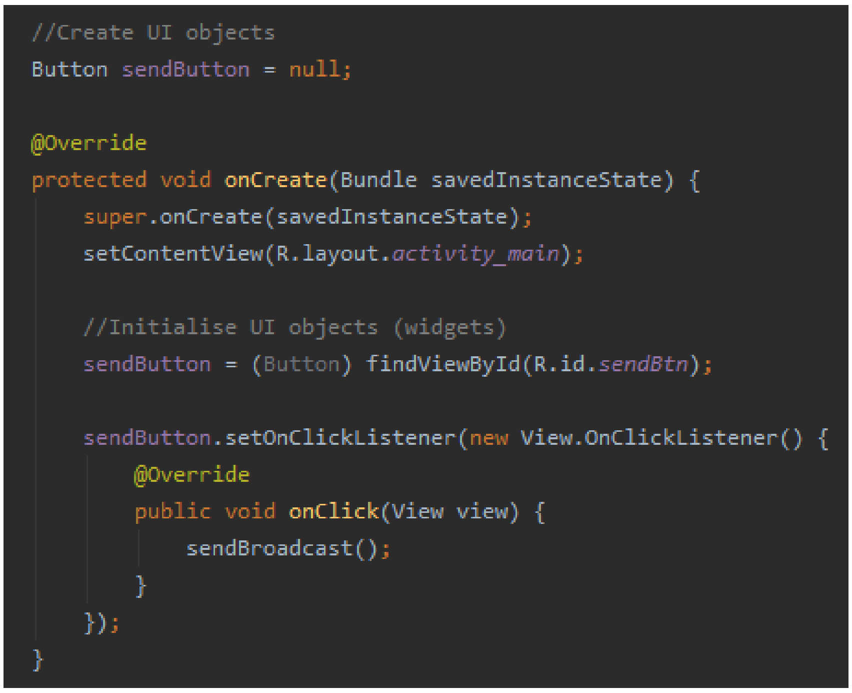


```
ConstraintLayout
▼ LinearLayout  (vertical)
    Ab textView  "@string/greetings_from_a_distant_...
    sendBtn  "@string/broadcast_message"
```

Figure 2

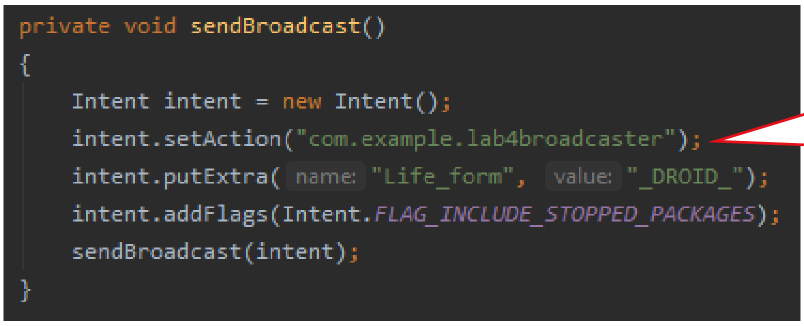
STEP 2 – CODE THE BROADCAST APPLICATION

a) In the MainActivity.java file do your regular setup and initialisation of the UI widgets, add a listener to onClick event from the button and call a method to send a broadcast.



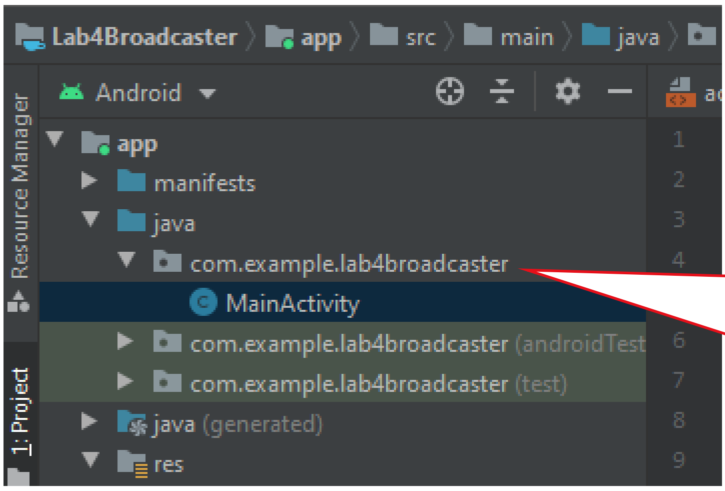
```java
//Create UI objects
Button sendButton = null;

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

    //Initialise UI objects (widgets)
    sendButton = (Button) findViewById(R.id.sendBtn);

    sendButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            sendBroadcast();
        }
    });
}
```

b) Add the method to send the broadcast



```java
private void sendBroadcast()
{
    Intent intent = new Intent();
    intent.setAction("com.example.lab4broadcaster");
    intent.putExtra( name: "Life_form",  value: "_DROID_");
    intent.addFlags(Intent.FLAG_INCLUDE_STOPPED_PACKAGES);
    sendBroadcast(intent);
}
```

Remember from the lecture, this is the package name and will vary depending on how you have set up your project



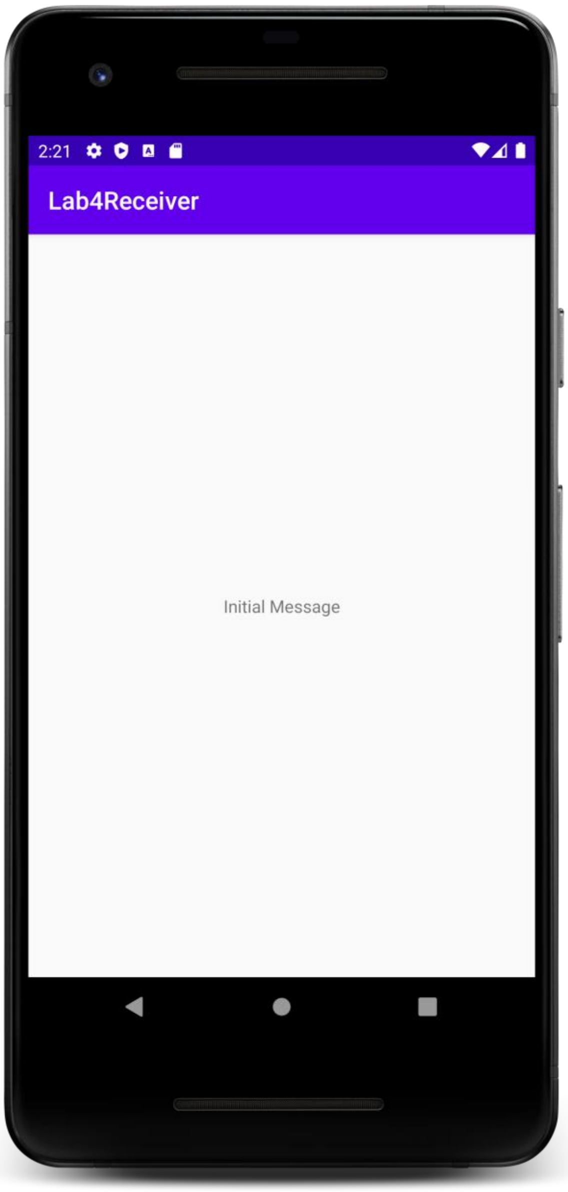
You can check by looking at the package name in the project files.

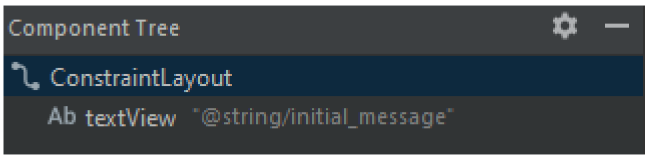
We are now broadcasting to all potential listeners. This is an asynchronous process, your code will continue while listeners process the broadcast.

STEP 3 – MAKING THE RECIEVER APPLICATION UI

Now we are making the separate app to receive the broadcast.

a. Create a new project & choose the empty activity template. (See week 1 lab)
b. Create a new Android Project using API 25: Android 7.1.1 (Nougat)
c. Create the UI in the visual DESIGN view for the main activity as in Figure 3. We do not need a linear layout this time as all we have is one view (the textView) which can be constrained directly. If you need a hint see the component tree in Figure 4.





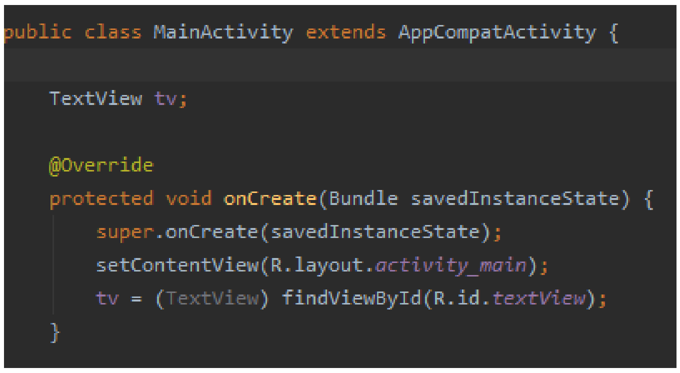
ConstraintLayout
Ab textView "@string/initial_message"

Figure 3

Figure 4

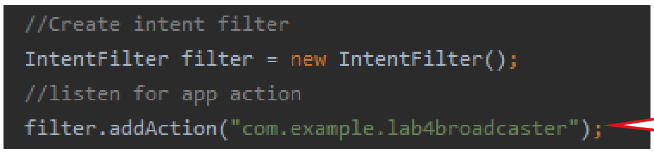
STEP 4 – CODE THE RECIEVER APPLICATION

a. In the MainActivity.java file do your regular setup and initialisation of the TextView.



```java
public class MainActivity extends AppCompatActivity {

    TextView tv;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tv = (TextView) findViewById(R.id.textView);
    }
```

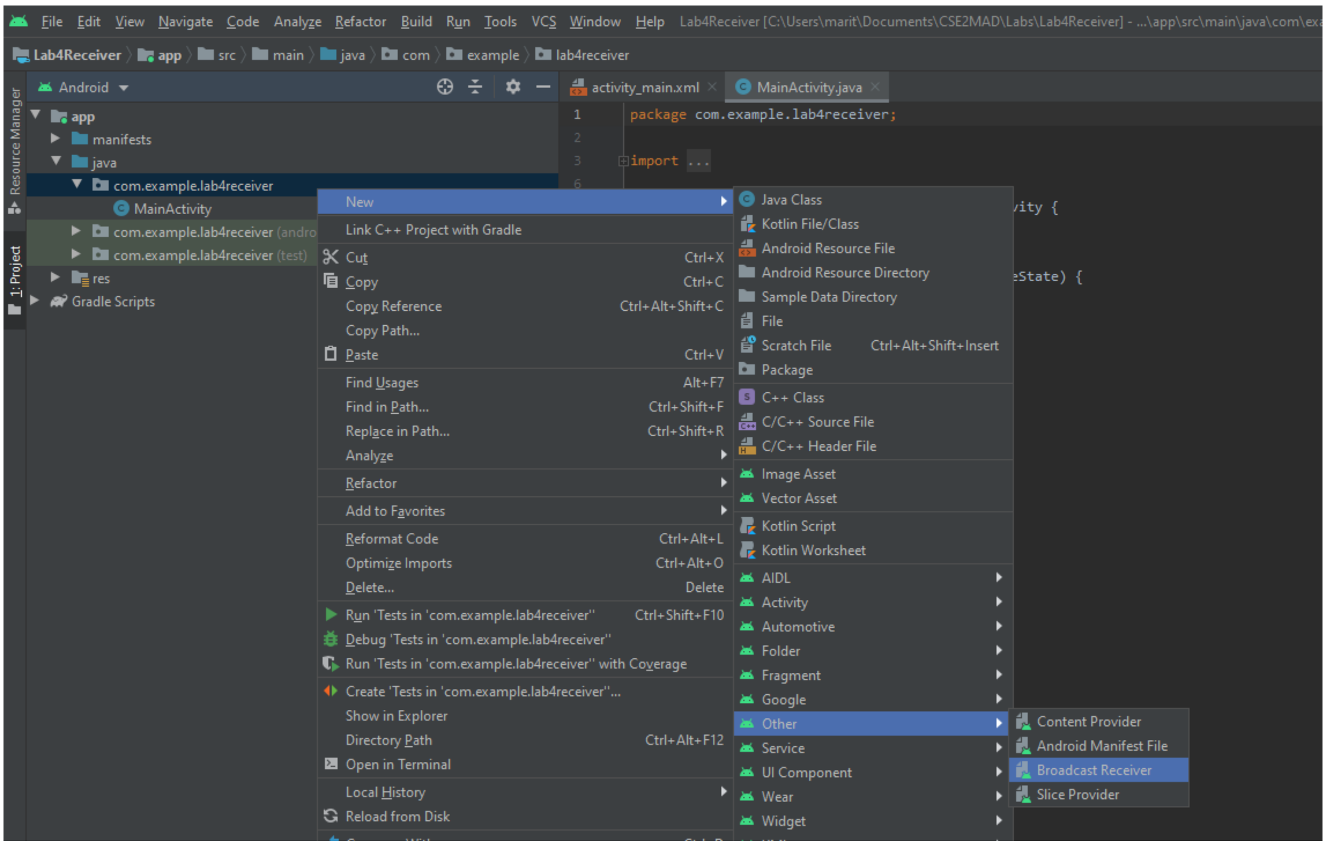
b. In the onCreate() method add an Intent Filter to specify which broadcasts your application will receive.



```java
//Create intent filter
IntentFilter filter = new IntentFilter();
//listen for app action
filter.addAction("com.example.lab4broadcaster");
```

This is the package name from the Broadcast application we made initially, as that's the broadcast we wish to receive in this filter.

c. Now we need to make the Broadcast receiver, right click on the package name and select New > Other > Broadcast Receiver and in the resulting pop up set the name to **MyStandAloneReceiver**. A new java file will be generated



File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Lab4Receiver [C:\Users\marit\Documents\CSE2MAD\Labs\Lab4Receiver] - ...\app\src\main\java\com\exa
Lab4Receiver app src main java com example lab4receiver
Android
app
manifests
java
com.example.lab4receiver
MainActivity
com.example.lab4receiver (andro
com.example.lab4receiver (test)
res
Gradle Scripts
activity_main.xml
MainActivity.java
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy Ctrl+C
Copy Reference Ctrl+Alt+Shift+C
Copy Path...
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Run 'Tests in 'com.example.lab4receiver'' Ctrl+Shift+F10
Debug 'Tests in 'com.example.lab4receiver''
Run 'Tests in 'com.example.lab4receiver'' with Coverage
Create 'Tests in 'com.example.lab4receiver''...
Show in Explorer
Directory Path Ctrl+Alt+F12
Open in Terminal
Local History
Reload from Disk
Java Class
Kotlin File/Class
Android Resource File
Android Resource Directory
Sample Data Directory
File
Scratch File Ctrl+Alt+Shift+Insert
Package
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset
Kotlin Script
Kotlin Worksheet
AIDL
Activity
Automotive
Folder
Fragment
Google
Other
Service
UI Component
Wear
Widget
Content Provider
Android Manifest File
Broadcast Receiver
Slice Provider

```java
package com.example.lab4receiver;

import ...
```

d. Navigate to the newly created **MyStandAloneReceiver.java** file**.** Now we need to add code to the receiver, to extract the message and generated a toast message for the user to view.

```
@Override
public void onReceive(Context context, Intent intent) {
    String msg = intent.getStringExtra( name: "Life_form");
    Toast.makeText(context,  text: "Broadcast intent detected:" + msg, Toast.LENGTH_LONG).show();
}
```

e. In the MainActivity.java file add a member variable for the broadcast receiver.

```
BroadcastReceiver standAloneRx;
```

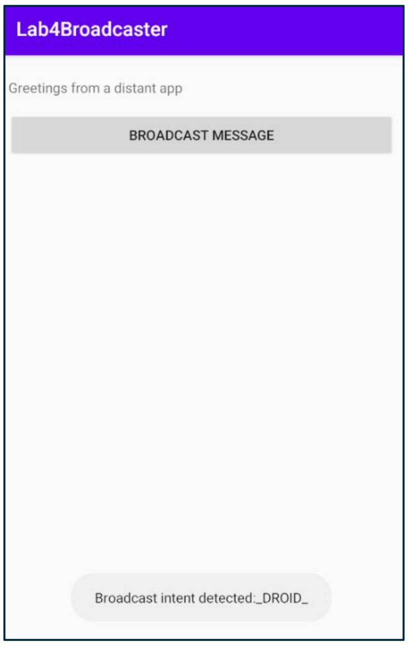
f. And instantiate the receiver object, using it and your previously created filter object to register the receiver. If you are wondering where this code belongs think about at what stage of the android lifecycle do you want to start receiving broadcasts?

```
//Instantiate receiver objects
standAloneRx = new MyStandAloneReceiver();

//Register receivers with intent filter & receiver objects (Context-registered receivers)
registerReceiver(standAloneRx, filter);
```

g. Add the onDestroy() method to unregister the reciever is the activity is destroyed. Why would you do this?

```
@Override
protected void onDestroy() {
    unregisterReceiver(standAloneRx);
    super.onDestroy();
}
```

Now run both apps on your emulator/hardware, navigate to the Broadcast app (with the receiver in the background) and click on the broadcast message, you should receive the toast message as below;


Lab4Broadcaster

Greetings from a distant app

BROADCAST MESSAGE

Broadcast intent detected:_DROID_

STEP 5 – ADD A SUBCLASSED BROADCAST RECIEVER (OPTIONAL BUT RECOMMENDED)

There are some instances where you would like the receiver to interact directly with an activity, in these instances you can create a subclass within the activity itself. Now we will add a receiver as a subclass but listen for the same broadcast as the MyStandAloneReceiver.

a. As per the stand alone receiver we will need to add a member variable for the broadcast receiver.

```
BroadcastReceiver subClassedRx;
```

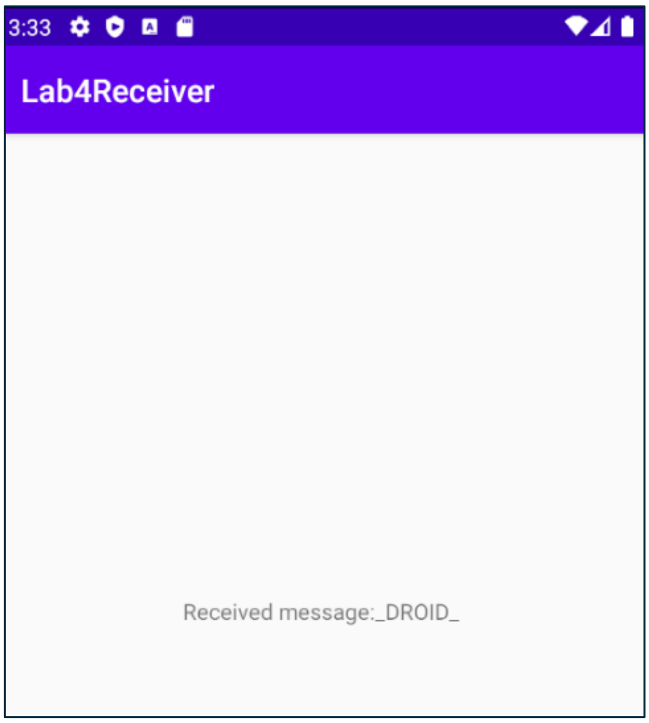
b. And instantiate the receiver object (subClassedRx), same as before, and using it and your previously created filter object to register the new receiver with the registerReciever(subClassedRx, filter) method.

c. Now we need to add the method receive the broadcast. This code will extract the message from the broadcast and display it in the TextView. As we have already registered the receiver in the onCreate() where do you think this code will go?

```
//By including the Broadcast Receiver as a subclass of your activity
//you can interact with the activity itself. (in this case the text view)
public class MySubclassReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        String msg = "Received message:" + intent.getStringExtra( name: "Life_form");
        tv.setText(msg);
    }
}
```

d. Finally unregister the subClassedRx object in the onDestroy() same as you did for the standAloneRx object.

e. Now run both apps on your emulator/hardware, navigate to the Broadcast app (with the receiver in the background) and click on the broadcast message, you should receive the toast message as previously but if you navigate back to the Receiver app the TextView content should have changed to reflect the message.



Received message:_DROID_