# Laporan Praktikum 5
# Pengembangan Aplikasi Mobile

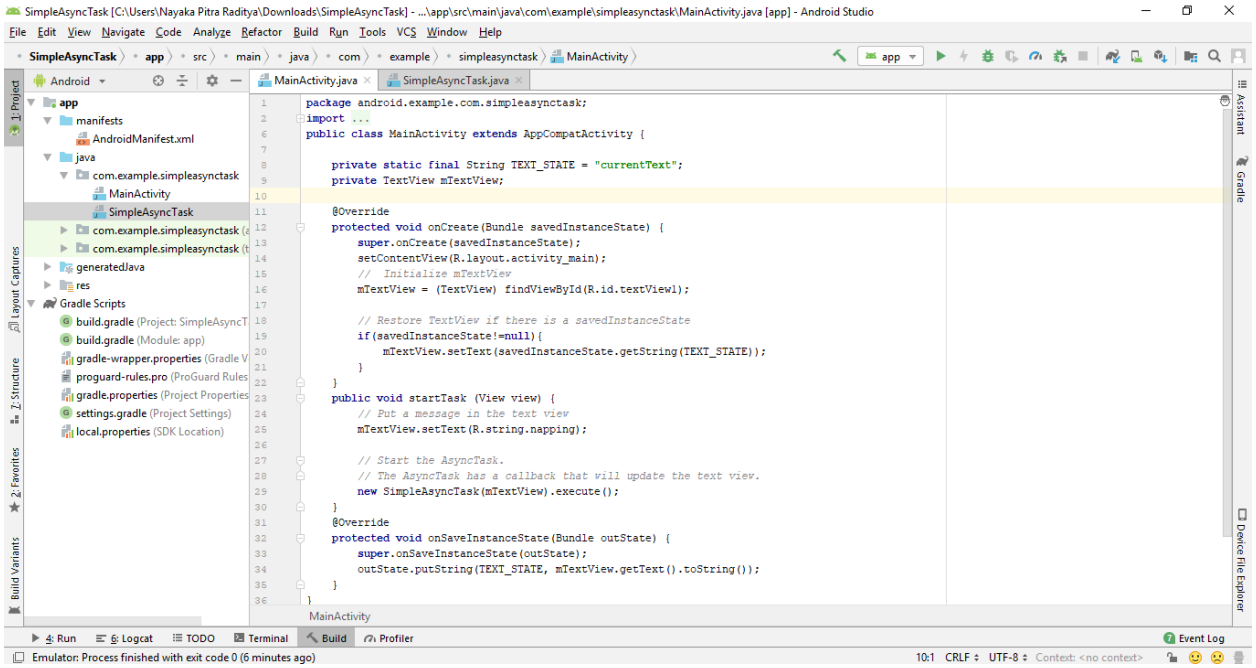
Nama : Nayaka Pitra Raditya
NIM : 14116030

TP 2018/2019

1. Membuat AsyncTask

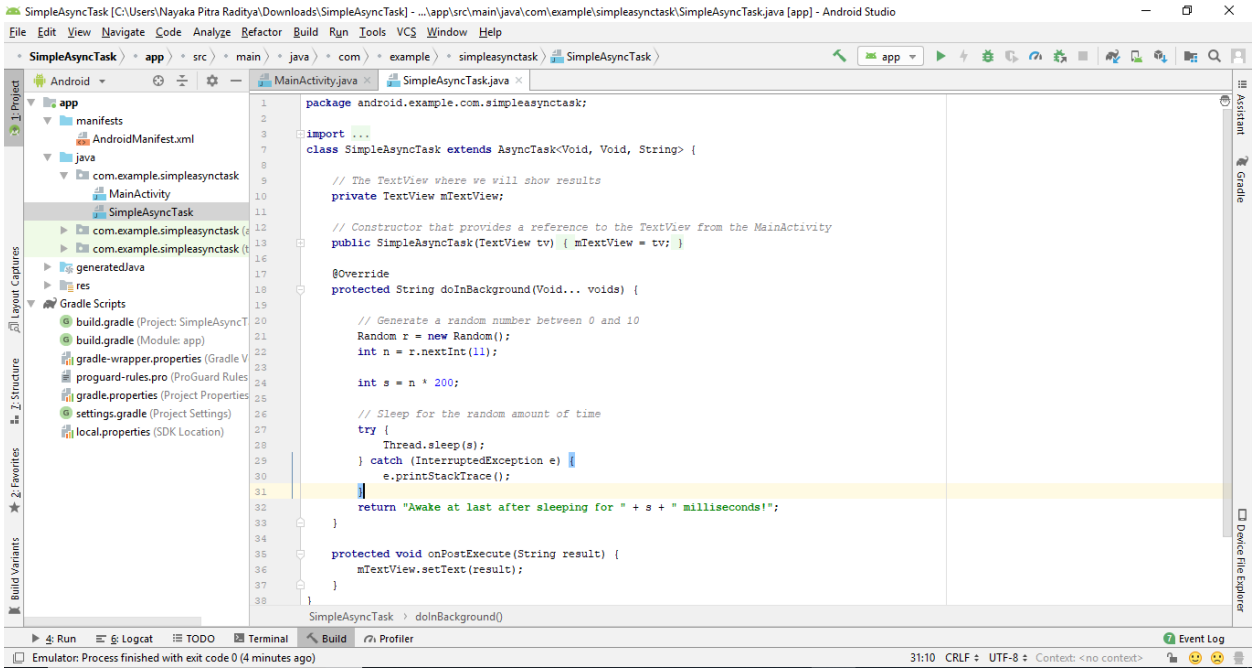
   Screenshot Source Code

   a. MainActivity



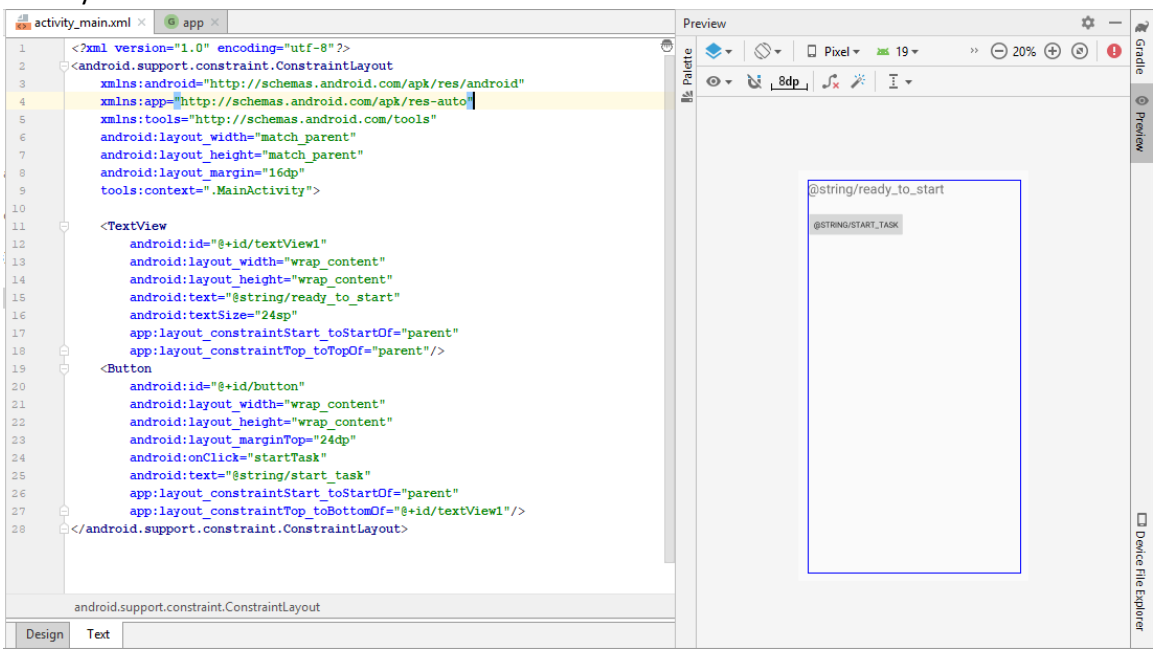
```
package android.example.com.simpleasynctask;
import ...
public class MainActivity extends AppCompatActivity {

    private static final String TEXT_STATE = "currentText";
    private TextView mTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //  Initialize mTextView
        mTextView = (TextView) findViewById(R.id.textView1);

        // Restore TextView if there is a savedInstanceState
        if(savedInstanceState!=null){
            mTextView.setText(savedInstanceState.getString(TEXT_STATE));
        }
    }
    public void startTask (View view) {
        // Put a message in the text view
        mTextView.setText(R.string.napping);

        // Start the AsyncTask.
        // The AsyncTask has a callback that will update the text view.
        new SimpleAsyncTask(mTextView).execute();
    }
    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        outState.putString(TEXT_STATE, mTextView.getText().toString());
    }
}
```

   b. SimpleAsyncTask

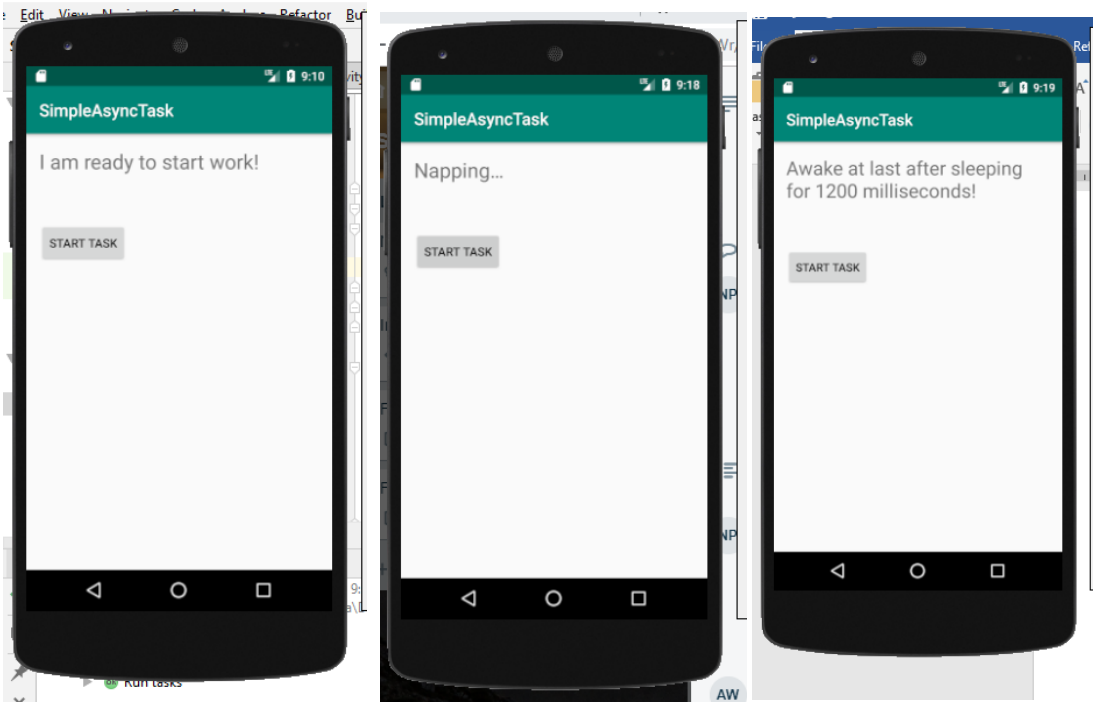


```
package android.example.com.simpleasynctask;

import ...
class SimpleAsyncTask extends AsyncTask<Void, Void, String> {

    // The TextView where we will show results
    private TextView mTextView;

    // Constructor that provides a reference to the TextView from the MainActivity
    public SimpleAsyncTask(TextView tv) { mTextView = tv; }

    @Override
    protected String doInBackground(Void... voids) {

        // Generate a random number between 0 and 10
        Random r = new Random();
        int n = r.nextInt(11);

        int s = n * 200;

        // Sleep for the random amount of time
        try {
            Thread.sleep(s);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        return "Awake at last after sleeping for " + s + " milliseconds!";
    }

    protected void onPostExecute(String result) {
        mTextView.setText(result);
    }
}
```

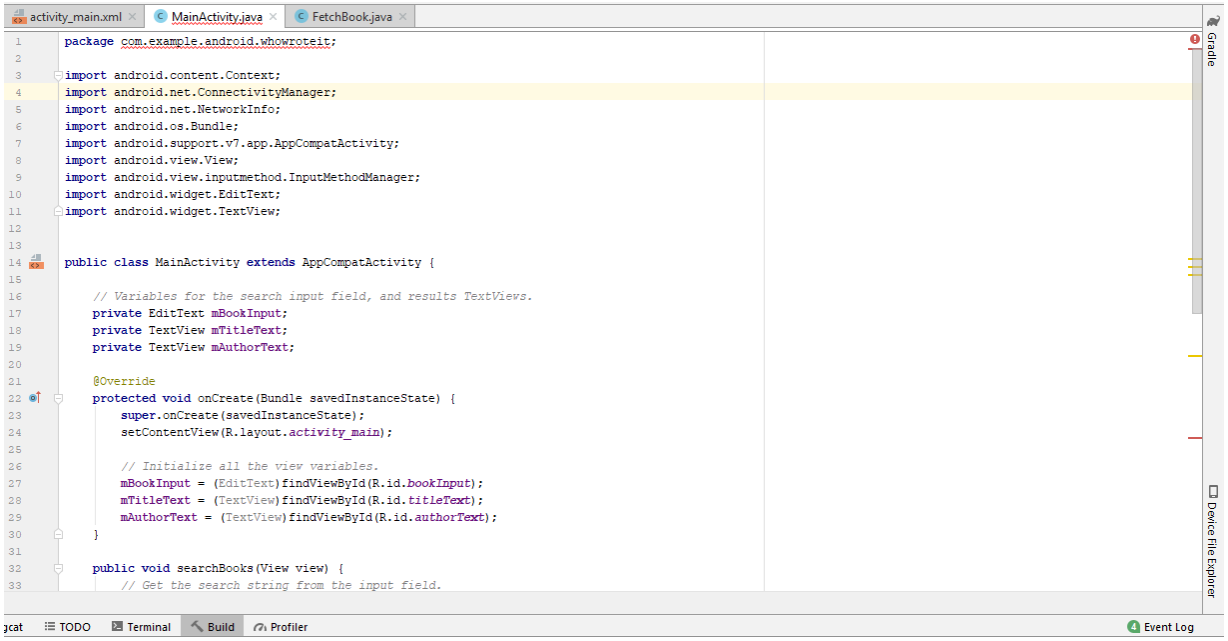
c. ActivityMain



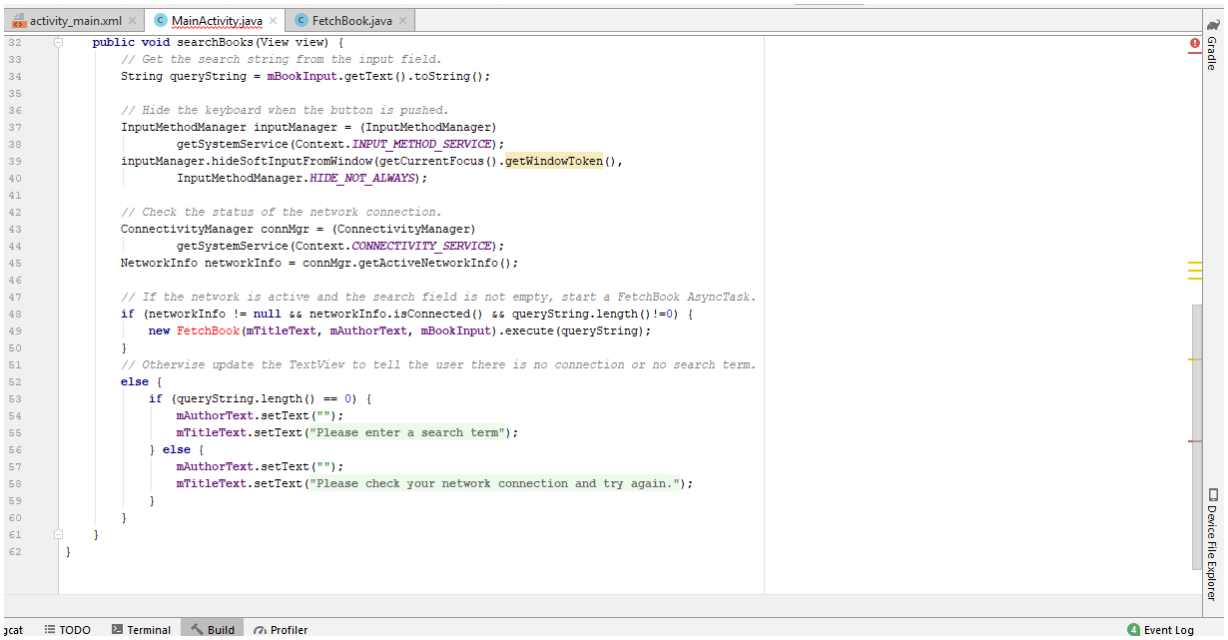

Screenshot Program



2. Menghubungkan ke Internet dengan AsyncTask dan AsyncTaskLoader
   Screenshot Source
   a. MainActivity

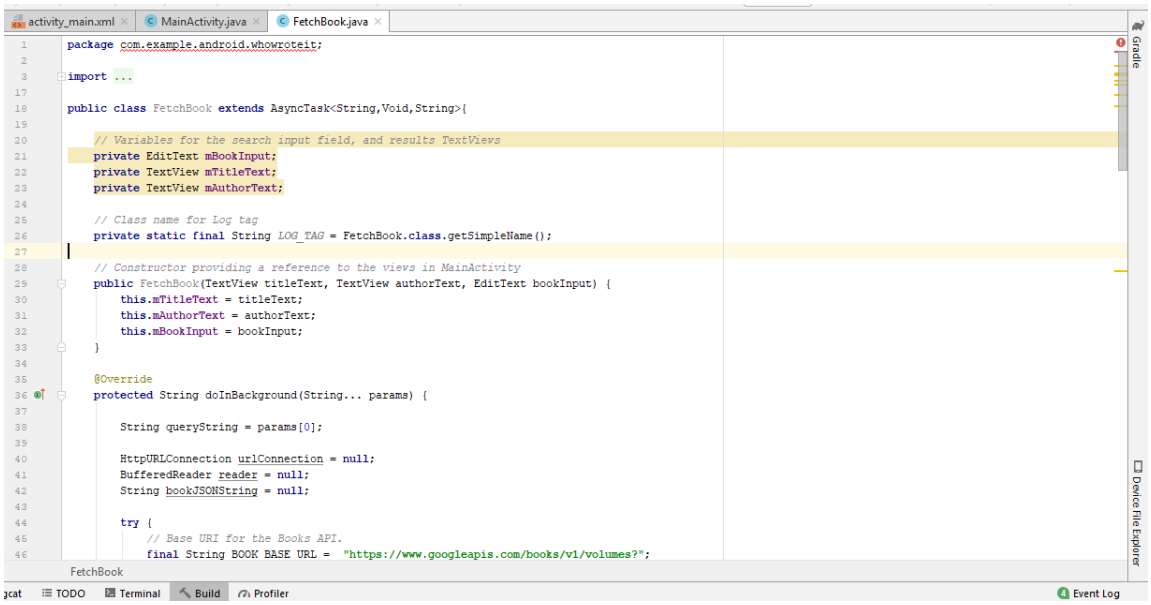


```
package com.example.android.whowroteit;

import android.content.Context;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.TextView;


public class MainActivity extends AppCompatActivity {

    // Variables for the search input field, and results TextViews.
    private EditText mBookInput;
    private TextView mTitleText;
    private TextView mAuthorText;

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

        // Initialize all the view variables.
        mBookInput = (EditText)findViewById(R.id.bookInput);
        mTitleText = (TextView)findViewById(R.id.titleText);
        mAuthorText = (TextView)findViewById(R.id.authorText);
    }

    public void searchBooks(View view) {
        // Get the search string from the input field.
```

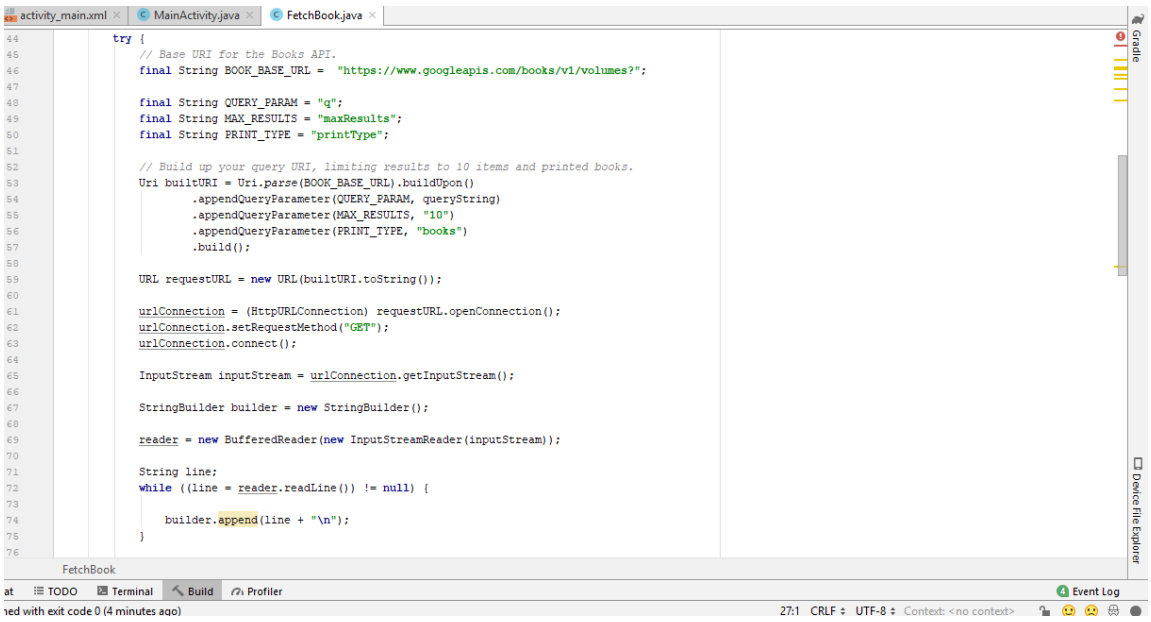


```
    public void searchBooks(View view) {
        // Get the search string from the input field.
        String queryString = mBookInput.getText().toString();

        // Hide the keyboard when the button is pushed.
        InputMethodManager inputManager = (InputMethodManager)
                getSystemService(Context.INPUT_METHOD_SERVICE);
        inputManager.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(),
                InputMethodManager.HIDE_NOT_ALWAYS);

        // Check the status of the network connection.
        ConnectivityManager connMgr = (ConnectivityManager)
                getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();

        // If the network is active and the search field is not empty, start a FetchBook AsyncTask.
        if (networkInfo != null && networkInfo.isConnected() && queryString.length()!=0) {
            new FetchBook(mTitleText, mAuthorText, mBookInput).execute(queryString);
        }
        // Otherwise update the TextView to tell the user there is no connection or no search term.
        else {
            if (queryString.length() == 0) {
                mAuthorText.setText("");
                mTitleText.setText("Please enter a search term");
            } else {
                mAuthorText.setText("");
                mTitleText.setText("Please check your network connection and try again.");
            }
        }
    }
}
```

b. FetchBook



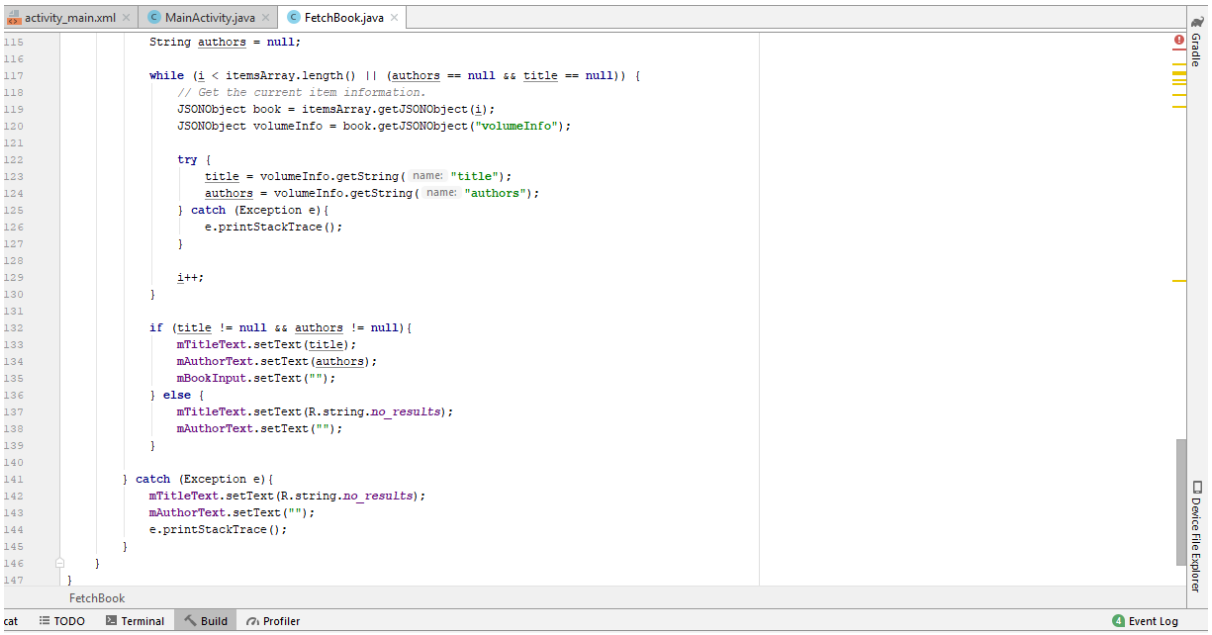
```
package com.example.android.whowroteit;

import ...

public class FetchBook extends AsyncTask<String,Void,String>{

    // Variables for the search input field, and results TextViews
    private EditText mBookInput;
    private TextView mTitleText;
    private TextView mAuthorText;

    // Class name for Log tag
    private static final String LOG_TAG = FetchBook.class.getSimpleName();

    // Constructor providing a reference to the views in MainActivity
    public FetchBook(TextView titleText, TextView authorText, EditText bookInput) {
        this.mTitleText = titleText;
        this.mAuthorText = authorText;
        this.mBookInput = bookInput;
    }

    @Override
    protected String doInBackground(String... params) {

        String queryString = params[0];

        HttpURLConnection urlConnection = null;
        BufferedReader reader = null;
        String bookJSONString = null;

        try {
            // Base URI for the Books API.
            final String BOOK_BASE_URL =  "https://www.googleapis.com/books/v1/volumes?";
```



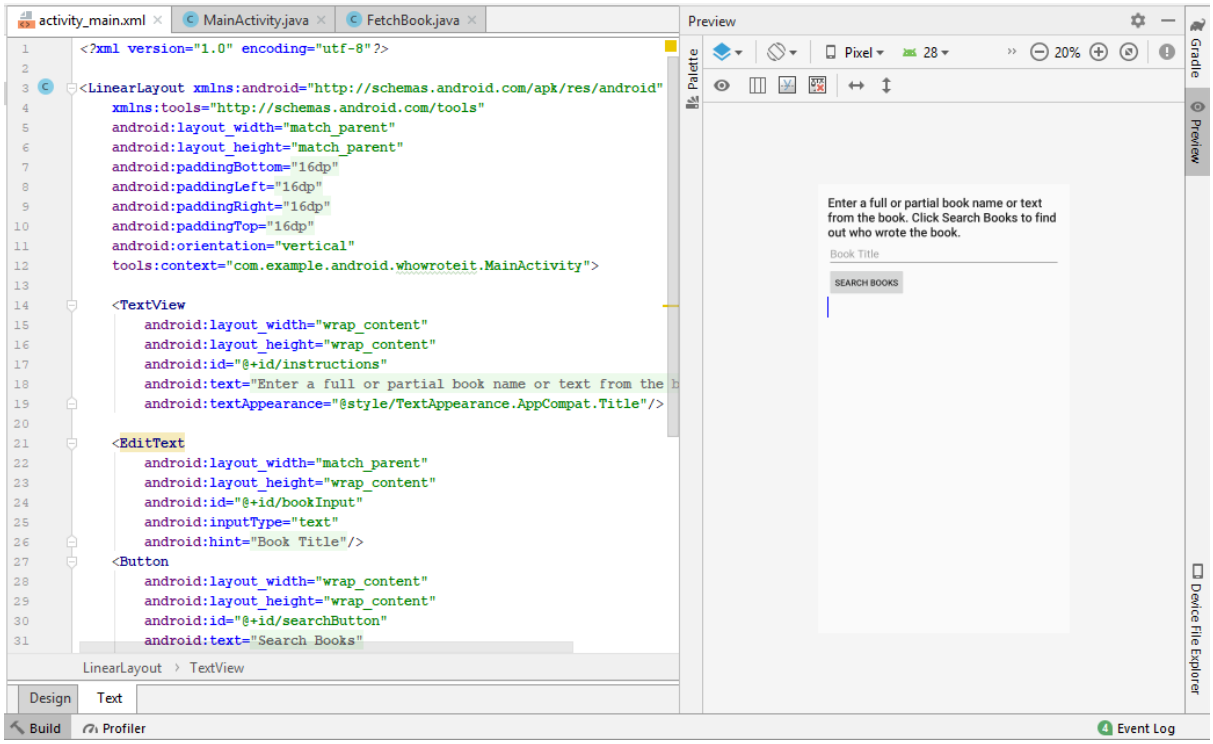
```
        try {
            // Base URI for the Books API.
            final String BOOK_BASE_URL =  "https://www.googleapis.com/books/v1/volumes?";

            final String QUERY_PARAM = "q";
            final String MAX_RESULTS = "maxResults";
            final String PRINT_TYPE = "printType";

            // Build up your query URI, limiting results to 10 items and printed books.
            Uri builtURI = Uri.parse(BOOK_BASE_URL).buildUpon()
                    .appendQueryParameter(QUERY_PARAM, queryString)
                    .appendQueryParameter(MAX_RESULTS, "10")
                    .appendQueryParameter(PRINT_TYPE, "books")
                    .build();

            URL requestURL = new URL(builtURI.toString());

            urlConnection = (HttpURLConnection) requestURL.openConnection();
            urlConnection.setRequestMethod("GET");
            urlConnection.connect();

            InputStream inputStream = urlConnection.getInputStream();

            StringBuilder builder = new StringBuilder();

            reader = new BufferedReader(new InputStreamReader(inputStream));

            String line;
            while ((line = reader.readLine()) != null) {

                builder.append(line + "\n");
            }
```

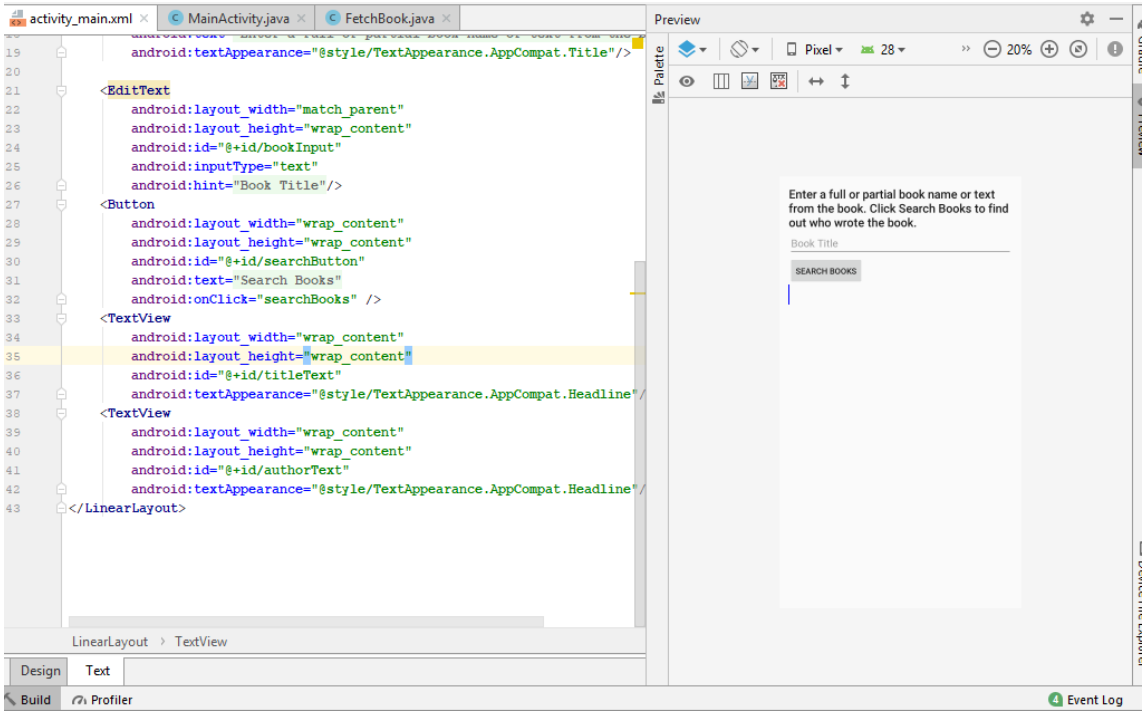
activity_main.xml
MainActivity.java
FetchBook.java
if (builder.length() == 0) {
return null;
}
bookJSONString = builder.toString();
// Catch errors.
} catch (IOException e) {
e.printStackTrace();
} finally {
if (urlConnection != null) {
urlConnection.disconnect();
}
if (reader != null) {
try {
reader.close();
} catch (IOException e) {
e.printStackTrace();
}
}
}
return bookJSONString;
}
@Override
protected void onPostExecute(String s) {
super.onPostExecute(s);
try {
// Convert the response into a JSON object.
JSONObject jsonObject = new JSONObject(s);
// Get the JSONArray of book items.
FetchBook
TODO
Terminal
Build
Profiler
Event Log
Gradle
Device File Explorer

activity_main.xml
MainActivity.java
FetchBook.java
// Get the JSONArray of book items.
JSONArray itemsArray = jsonObject.getJSONArray( name: "items");
// Initialize iterator and results fields.
int i = 0;
String title = null;
String authors = null;
while (i < itemsArray.length() || (authors == null && title == null)) {
// Get the current item information.
JSONObject book = itemsArray.getJSONObject(i);
JSONObject volumeInfo = book.getJSONObject("volumeInfo");
try {
title = volumeInfo.getString( name: "title");
authors = volumeInfo.getString( name: "authors");
} catch (Exception e){
e.printStackTrace();
}
i++;
}
if (title != null && authors != null){
mTitleText.setText(title);
mAuthorText.setText(authors);
mBookInput.setText("");
} else {
mTitleText.setText(R.string.no_results);
mAuthorText.setText("");
}
} catch (Exception e){
FetchBook
TODO
Terminal
Build
Profiler
Event Log
Gradle
Device File Explorer

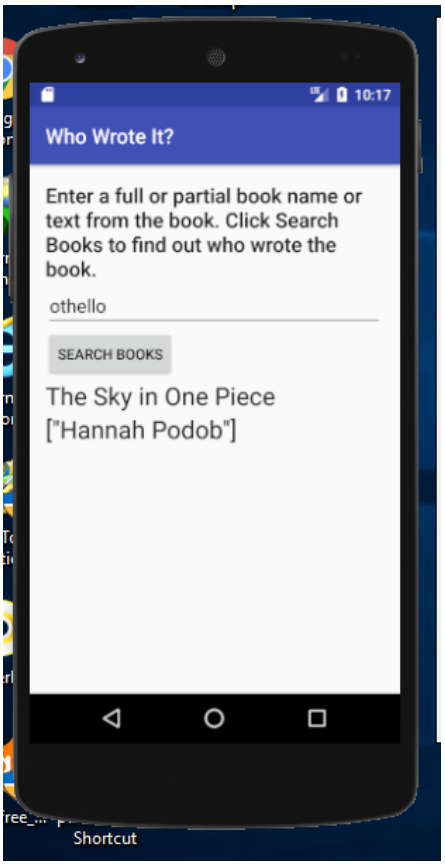



c. ActivityMain





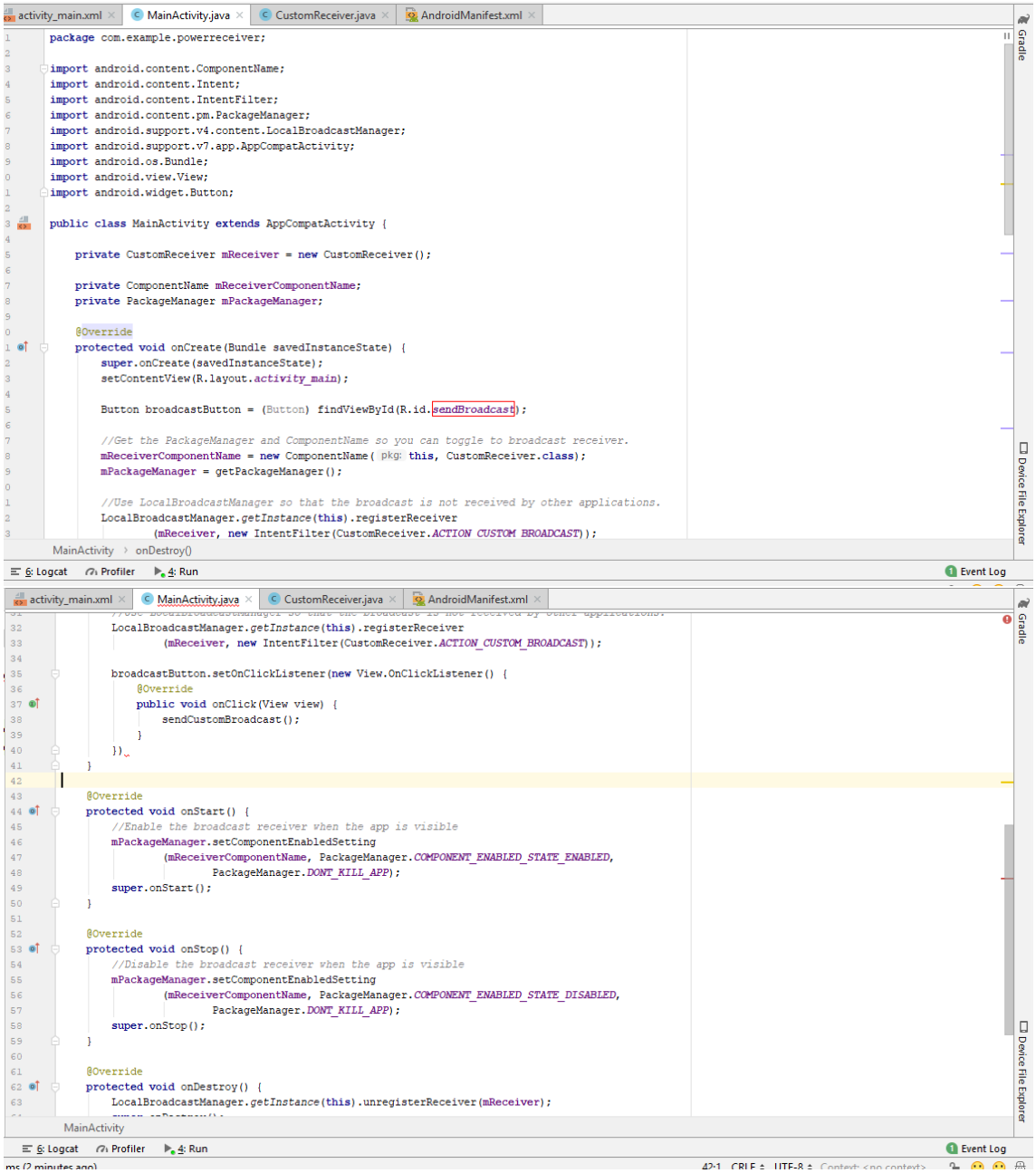

Screenshot Program



3. Penerima siaran

   Screenshot Source Code

   a. MainActivity



```
activity_main.xml | MainActivity.java | CustomReceiver.java | AndroidManifest.xml

package com.example.powerreceiver;

import android.content.ComponentName;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.support.v4.content.LocalBroadcastManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    private CustomReceiver mReceiver = new CustomReceiver();

    private ComponentName mReceiverComponentName;
    private PackageManager mPackageManager;

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

        Button broadcastButton = (Button) findViewById(R.id.sendBroadcast);

        //Get the PackageManager and ComponentName so you can toggle to broadcast receiver.
        mReceiverComponentName = new ComponentName( pkg: this, CustomReceiver.class);
        mPackageManager = getPackageManager();

        //Use LocalBroadcastManager so that the broadcast is not received by other applications.
        LocalBroadcastManager.getInstance(this).registerReceiver
                (mReceiver, new IntentFilter(CustomReceiver.ACTION_CUSTOM_BROADCAST));

MainActivity › onDestroy()
6: Logcat | Profiler | 4: Run | Event Log
```

```
activity_main.xml | MainActivity.java | CustomReceiver.java | AndroidManifest.xml

        LocalBroadcastManager.getInstance(this).registerReceiver
                (mReceiver, new IntentFilter(CustomReceiver.ACTION_CUSTOM_BROADCAST));

        broadcastButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                sendCustomBroadcast();
            }
        });
    }

    @Override
    protected void onStart() {
        //Enable the broadcast receiver when the app is visible
        mPackageManager.setComponentEnabledSetting
                (mReceiverComponentName, PackageManager.COMPONENT_ENABLED_STATE_ENABLED,
                        PackageManager.DONT_KILL_APP);
        super.onStart();
    }

    @Override
    protected void onStop() {
        //Disable the broadcast receiver when the app is visible
        mPackageManager.setComponentEnabledSetting
                (mReceiverComponentName, PackageManager.COMPONENT_ENABLED_STATE_DISABLED,
                        PackageManager.DONT_KILL_APP);
        super.onStop();
    }

    @Override
    protected void onDestroy() {
        LocalBroadcastManager.getInstance(this).unregisterReceiver(mReceiver);

MainActivity
6: Logcat | Profiler | 4: Run | Event Log
```

```
activity_main.xml | MainActivity.java | CustomReceiver.java | AndroidManifest.xml
```

```
            }
        });
    }

    @Override
    protected void onStart() {
        //Enable the broadcast receiver when the app is visible
        mPackageManager.setComponentEnabledSetting
                (mReceiverComponentName, PackageManager.COMPONENT_ENABLED_STATE_ENABLED,
                        PackageManager.DONT_KILL_APP);
        super.onStart();
    }

    @Override
    protected void onStop() {
        //Disable the broadcast receiver when the app is visible
        mPackageManager.setComponentEnabledSetting
                (mReceiverComponentName, PackageManager.COMPONENT_ENABLED_STATE_DISABLED,
                        PackageManager.DONT_KILL_APP);
        super.onStop();
    }

    @Override
    protected void onDestroy() {
        LocalBroadcastManager.getInstance(this).unregisterReceiver(mReceiver);
        super.onDestroy();
    }

    private void sendCustomBroadcast() {
        Intent customBroadcastIntent = new Intent(CustomReceiver.ACTION_CUSTOM_BROADCAST);
        LocalBroadcastManager.getInstance(this).sendBroadcast(customBroadcastIntent);
    }
}
```

```
MainActivity
6: Logcat | Profiler | 4: Run | Event Log
```

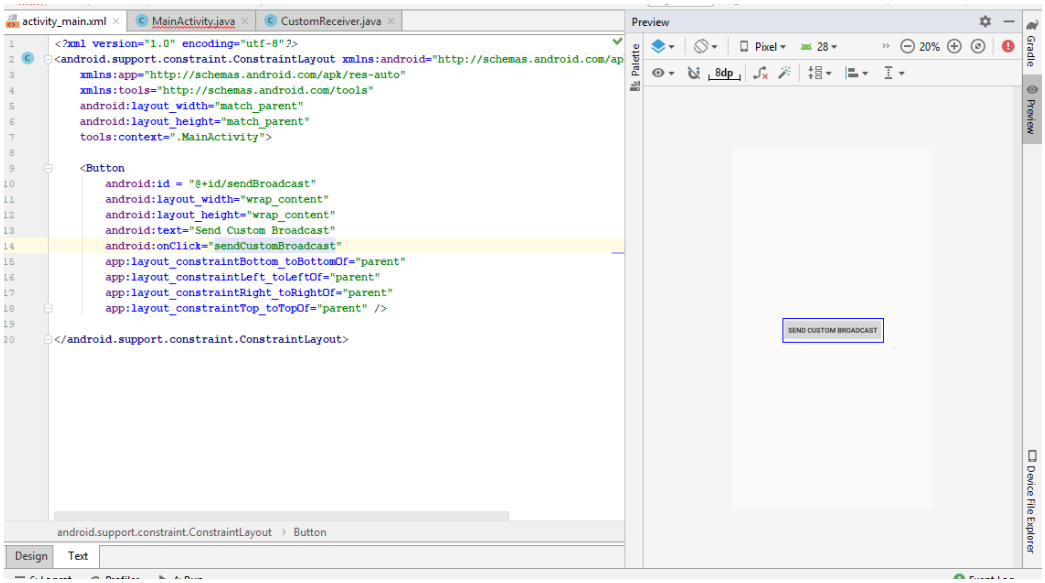
b. CustomReceiver

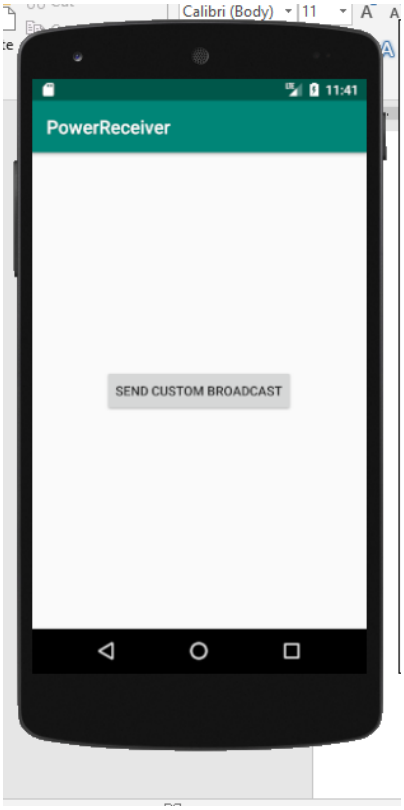
```
activity_main.xml | CustomReceiver.java | AndroidManifest.xml
```

```
package com.example.powerreceiver;

import ...

public class CustomReceiver extends BroadcastReceiver {

    //String constant that defines the custom Broadcast Action
    static final String ACTION_CUSTOM_BROADCAST =
            "com.example.android.powerreceiver.ACTION_CUSTOM_BROADCAST";

    public CustomReceiver() {
    }

    @Override
    public void onReceive(Context context, Intent intent) {
        String intentAction = intent.getAction();
        String toastMessage = null;
        switch (intentAction){
            case Intent.ACTION_POWER_CONNECTED:
                toastMessage = context.getString(R.string.power_connected);
                break;
            case Intent.ACTION_POWER_DISCONNECTED:
                toastMessage = context.getString(R.string.power_disconnected);
                break;
            case ACTION_CUSTOM_BROADCAST:
                toastMessage = context.getString(R.string.custom_broadcast_toast);
                break;
        }

        Toast.makeText(context, toastMessage, Toast.LENGTH_SHORT).show();
    }

}
```

```
CustomReceiver
6: Logcat | Profiler | 4: Run | Event Log
```

c. ActivityMain




Screenshot Program