Intents

Background:

In Android, we can send data between apps quite easily. For example, when we want to share images on Facebook from a local app, we send the image to the Facebook app over the bundle. Then, the Facebook app reads the image data and posts it. Another example might be that when we want to call someone, we send the phone number to call over the bundle to the calling API.

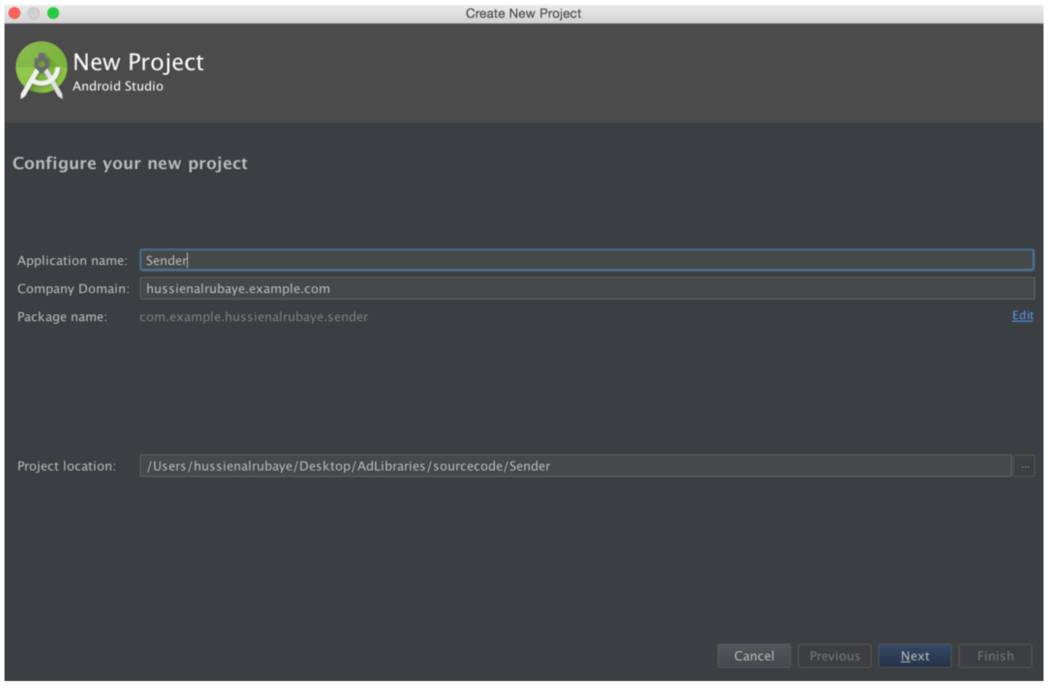
Sending data between apps is especially important for companies with multiple apps that need to interact in some way.

The problem is that sending data between apps is insecure in Android. For example, if we send data from app A to app B, any app C might have the same application ID of app B. App C could then receive the data that should be received by app B. We should not send important data between applications without taking defensive measures.

We will demonstrate examples to show how to send data between two applications. Next, we will show how a (possibly) malicious app could read the data, and finally, we will explain how we could prevent against this from happening.

Steps to build the sender app (app A):
This app will send comment data to app B.

1. Create a new project with name "Sender". Again, remember to save the package name. We will need that later.



Configure your new project

Application name: Sender
Company Domain: hussienalrubaye.example.com
Package name: com.example.hussienalrubaye.sender Edit
Project location: /Users/hussienalrubaye/Desktop/AdLibraries/sourcecode/Sender

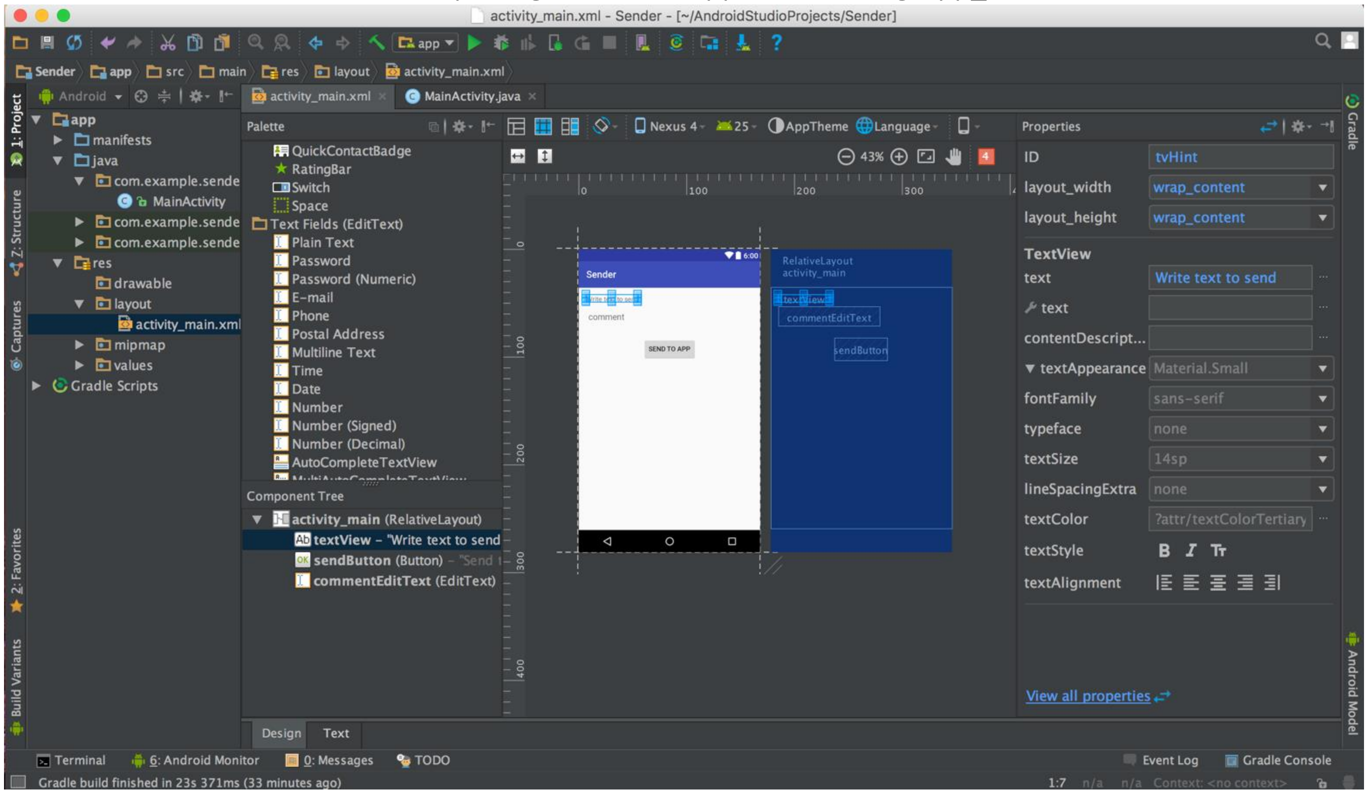
2. Add some objects to the **activity_main.xml**, found under "app/res/layout/". Add a TextView, EditText, and a Button.

   Once you have the objects in the device layout, click on the TextView and change its properties from right side panel to the following:
   - ID: tvHint

   The EditText should have these properties:
   - ID: commentEditText
   - Hint: comment
   - Delete all text in the box next to the "Text" label

   For the Button:
   - ID: sendButton
   - Text: Send to app

- You may need to play around with this to avoid Android Studio autocompleting the word "app" to "@string/app_name"



activity_main.xml - Sender - [~/AndroidStudioProjects/Sender]

Component Tree
- activity_main (RelativeLayout)
  - textView - "Write text to send
  - sendButton (Button) - "Send t
  - commentEditText (EditText)

Properties

ID	tvHint
layout_width	wrap_content
layout_height	wrap_content
TextView	
text	Write text to send
text	
contentDescript...	
textAppearance	Material.Small
fontFamily	sans-serif
typeface	none
textSize	14sp
lineSpacingExtra	none
textColor	?attr/textColorTertiary

Gradle build finished in 23s 371ms (33 minutes ago)

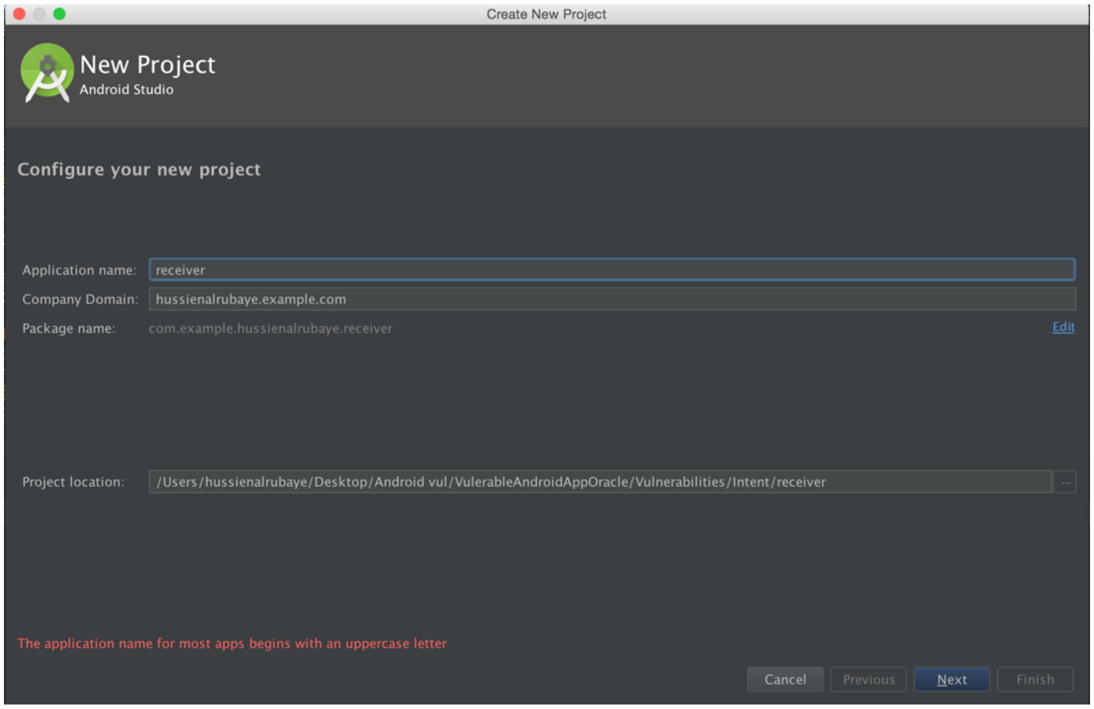
3. The code in **MainActivity.java**, found in "app/java/your_package_name_here/", will look like this:

```
package com.example.sender;

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

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Define send button
        Button sendButton = (Button)findViewById(R.id.sendButton);

        // init commentEditText
        final EditText commentEditText = (EditText)findViewById(R.id.commentEditText);

        // button listen to click event
        sendButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // set the package that we want to run
                Intent intent = getPackageManager().getLaunchIntentForPackage("com.example.receiver");


                // put the data that we want to send over intent
                intent.putExtra("Comment", commentEditText.getText().toString());

                // start another app
                startActivity(intent);
            }
        });
    }
}
```

Steps to build the Receiver app (app B):

This app will receive the comment data and read it.

1. Create a new project with name “Receiver”. Save the package name, as usual.

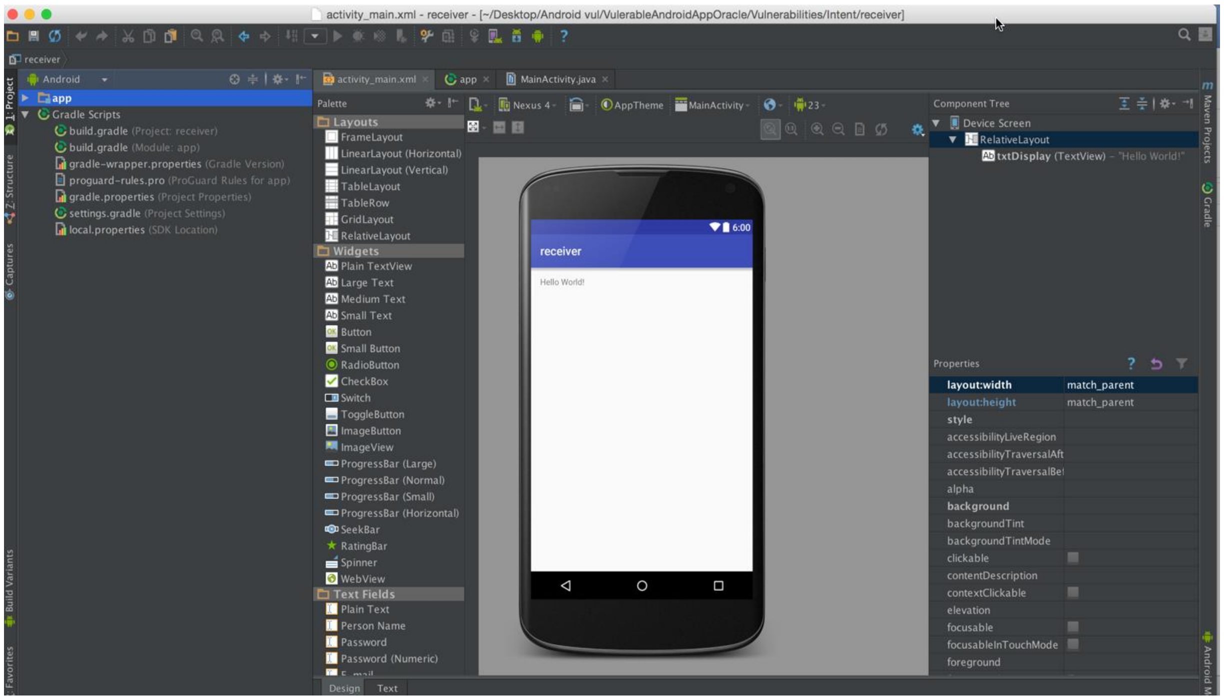


Configure your new project

Application name: receiver
Company Domain: hussienalrubaye.example.com
Package name: com.example.hussienalrubaye.receiver

Project location: /Users/hussienalrubaye/Desktop/Android vul/VulerableAndroidAppOracle/Vulnerabilities/Intent/receiver

The application name for most apps begins with an uppercase letter



2. Drag and drop a TextView from the “Palette” bar into the design view of **activity_main.xml**, found in “app/res/layout/”. Edit the properties of the TextView in the Properties panel that pops up when you click on it to be:
    - ID: txtDisplay

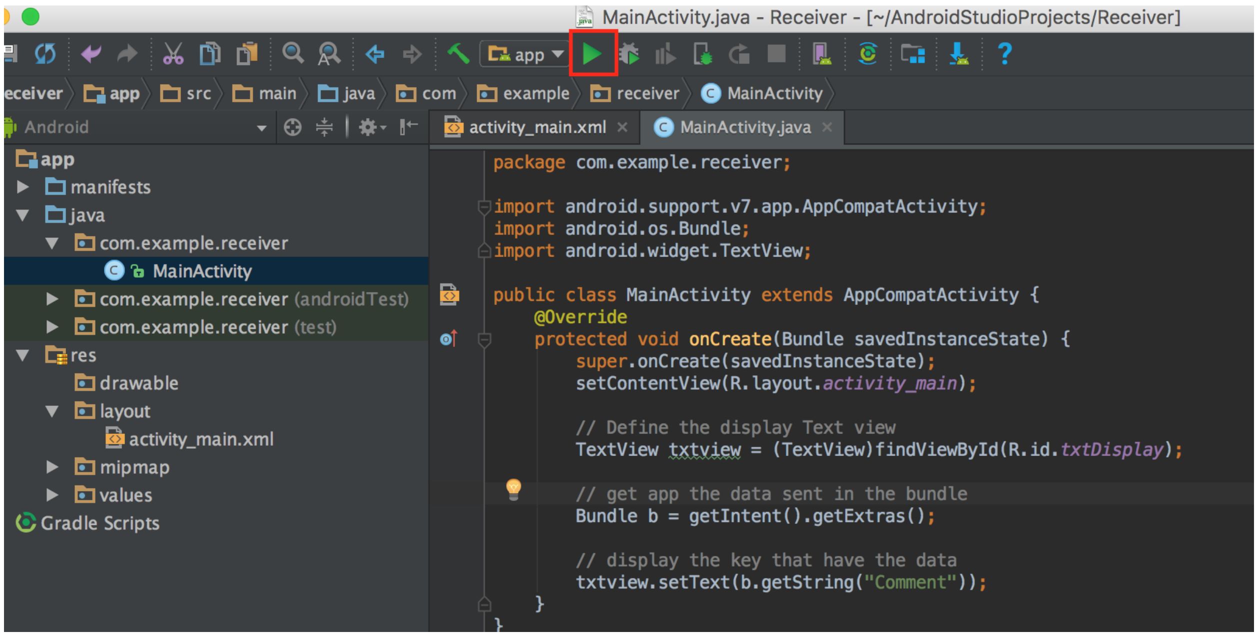
3. Copy and paste the following code into **MainActivity.java**.

```
package com.example.receiver;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      // Define the display Text view
      TextView txtview = (TextView)findViewById(R.id.txtDisplay);

      // get app the data sent on bundle
      Bundle b = getIntent().getExtras();
```

```
        // display the key that have the data
        txtview.setText(b.getString("Comment"));
    }
}
```

To run the apps:

1. Build and run the Receiver app by clicking the button circled in red below:



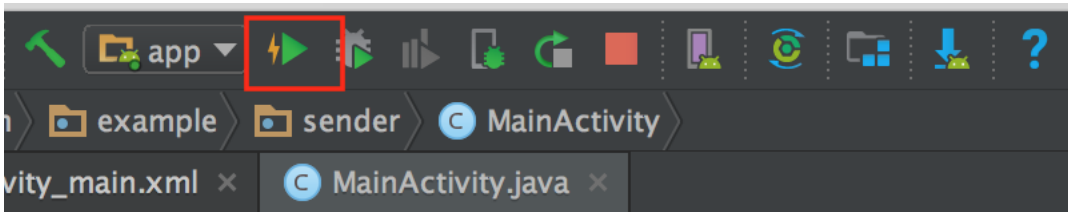
```
package com.example.receiver;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Define the display Text view
        TextView txtview = (TextView)findViewById(R.id.txtDisplay);

        // get app the data sent in the bundle
        Bundle b = getIntent().getExtras();

        // display the key that have the data
        txtview.setText(b.getString("Comment"));
    }
}
```

It will stop running and throw an error, but that is okay. We just need to run it that first time to load the app onto the device.

The Sender app will call the Receiver app, and as such, the Receiver app needs to be loaded onto the device before we run Sender. Note that it is important to load the apps onto the same virtual or physical device. We need all three apps to be running on the same device.

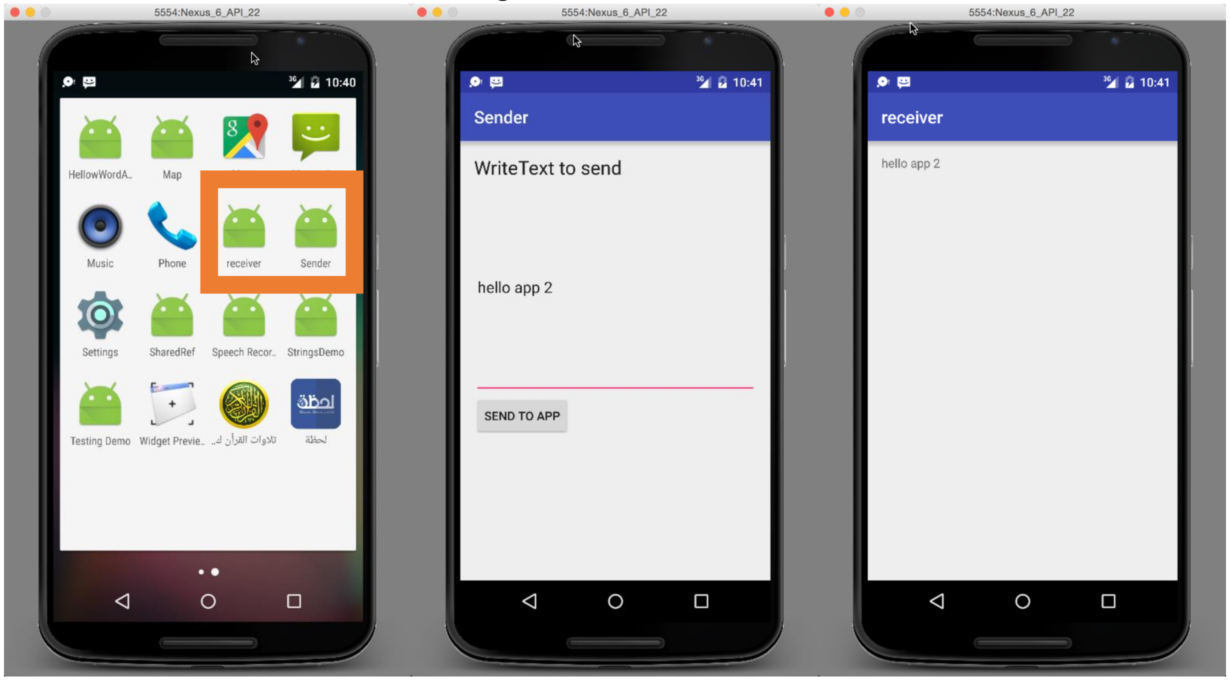
2. Build and run the Sender app. You can leave the emulator running in between these steps.

If the button to compile, load, and run has changed to the one depicted below, that simply means that Android Studio will be pushing any updates to the app directly, without rebuilding it from scratch. It is fine to continue.



3. Enter text into sender, click the Send to App button. When the Receiver app comes up, it will be displaying the comment you entered in Send to App.

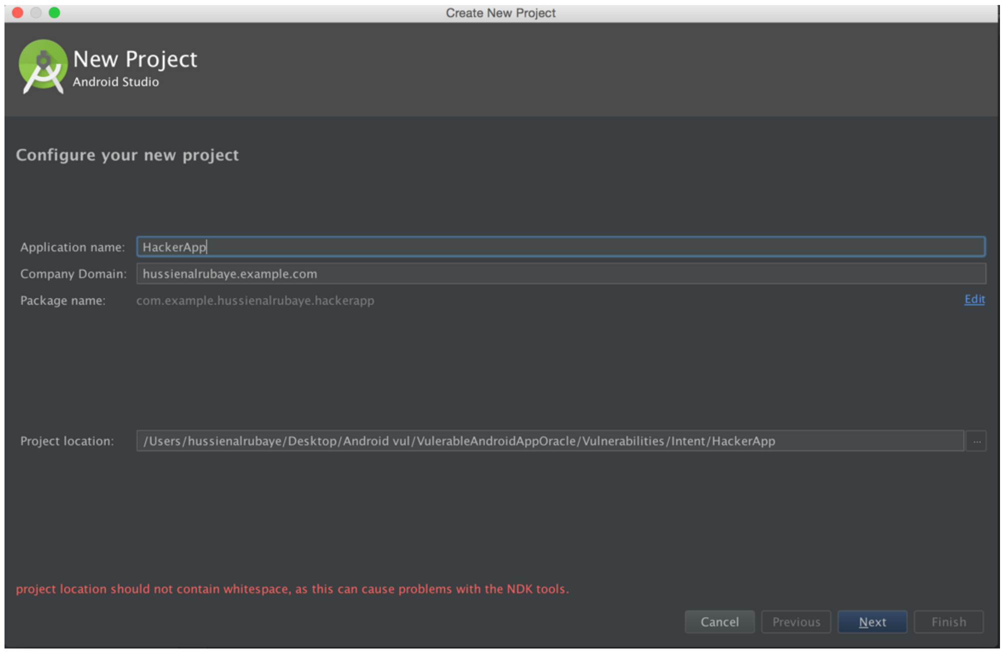
   See below for some images of what this could look like.



Steps to build the Hacker app (app C):

This app will try to read the data that should be read only by app B, or in our case, the Receiver app.

1. Create a new project with name “HackerApp”. Save the package.



Configure your new project

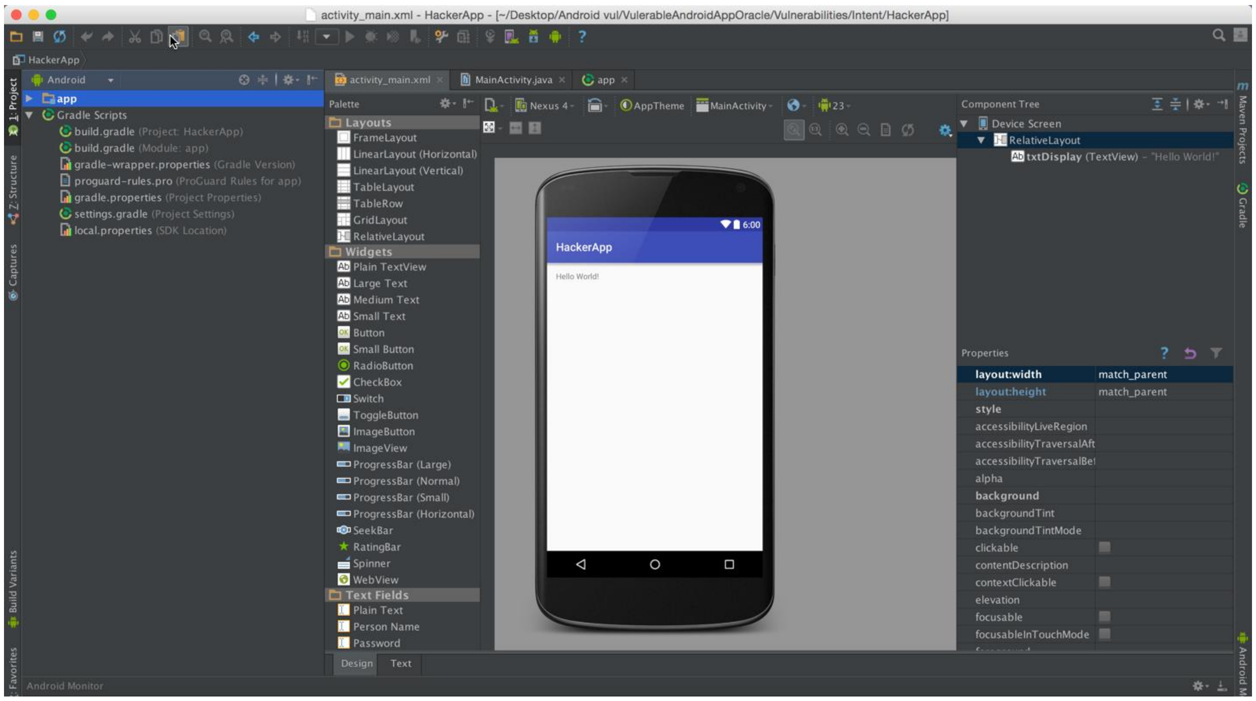
Application name: HackerApp
Company Domain: hussienalrubaye.example.com
Package name: com.example.hussienalrubaye.hackerapp

Project location: /Users/hussienalrubaye/Desktop/Android vul/VulerableAndroidAppOracle/Vulnerabilities/Intent/HackerApp

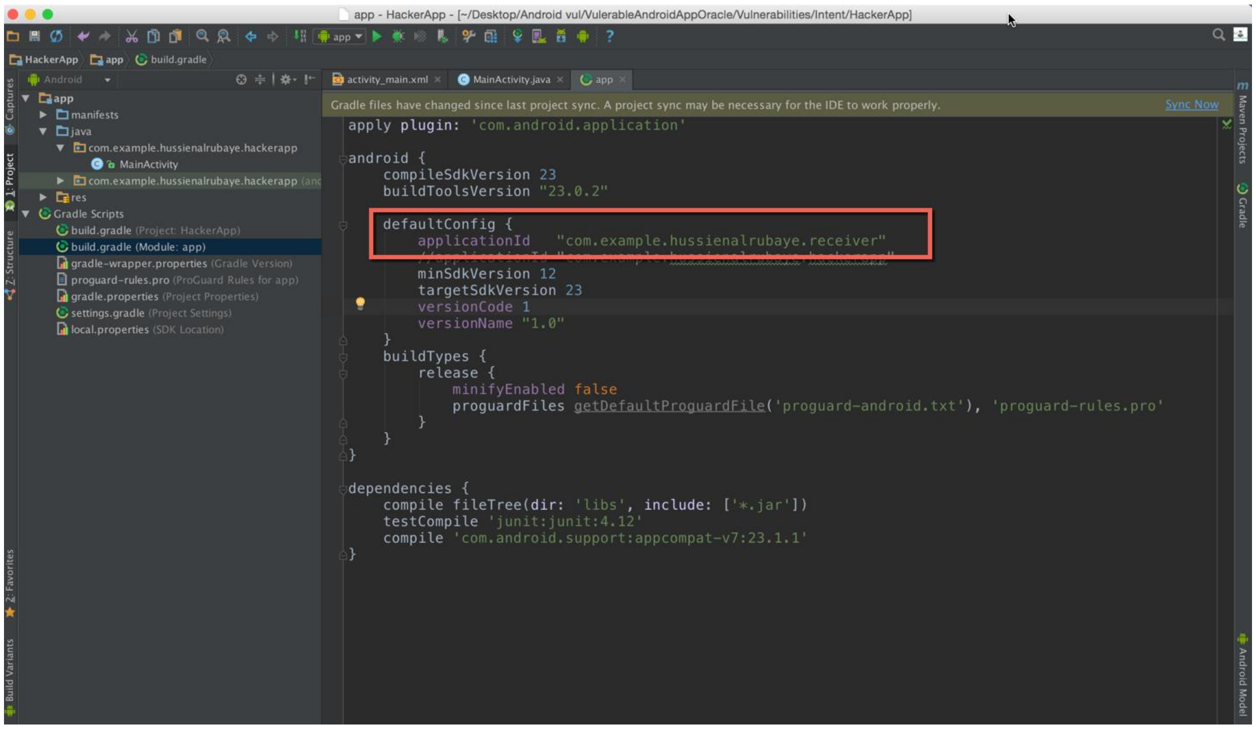
project location should not contain whitespace, as this can cause problems with the NDK tools.

2. Drag and drop a TextView object into the design view (found in “app/res/layout/activity_main.xml”), and access the properties panel by clicking on it in the Component Tree panel. Change the properties of this TextView to be:
    - ID: txtDisplay

   If the app template is Empty, it usually already has a “Hello, World” TextView present. We can use that.



3. Go to the Gradle build script for the specific module, found under “app/Gradle Scripts/”, in the build.gradle (Module: app) file. Change the applicationId to the applicationId the Receiver app has. Since they now have the same applicationId, it could receive the data that should be received by as the Receiver.



app - HackerApp - [~/Desktop/Android vul/VulerableAndroidAppOracle/Vulnerabilities/Intent/HackerApp]

Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

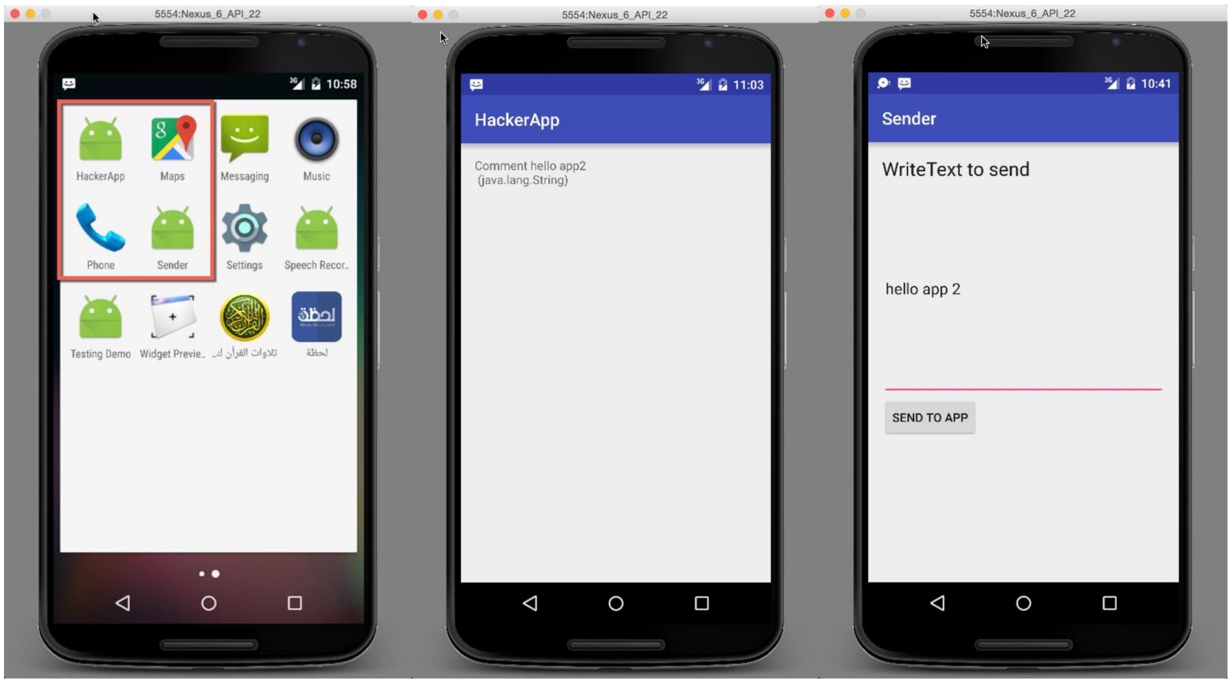
```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"

    defaultConfig {
        applicationId   "com.example.hussienalrubaye.receiver"
        //applicationId "com.example.hussienalrubaye.hackerapp"
        minSdkVersion 12
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.1.1'
}
```

4. **MainActivity.java** should have this code in it:

```java
package com.example.hackerapp;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

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

        // init the textView to display data
        TextView txtDisplay = (TextView)findViewById(R.id.txtDisplay);
        String dataBundle = "";

        // get the data sent in bundle from the app
        Bundle bundle = getIntent().getExtras();

        // loop through all keys in the bundle
        for (String key : bundle.keySet()) {
            // get object by key - we define it as object because we are not sure of type
            Object value = bundle.get(key);

            // get all keys
            dataBundle += String.format("%s %s (%s)", key, value.toString(), value.getClass().getName());
        }
        txtDisplay.setText(dataBundle);
    }
}
```

To run the apps:

1. Run and build the HackerApp as described above. Again, the app will crash. This is because it is reliant upon receiving a data bundle from the Intent. In this situation, however, we are running the app ourselves. The bundle ends up being empty (more correctly known as Null), and will crash when we try to iterate over it.
2. After we've loaded HackerApp onto our device, we go back to the home screen and run Sender. Write any comment and click to send.
3. The "Click to Send" button brings up HackerApp instead of Receiver. See that it has indeed intercepted the data meant for Receiver.



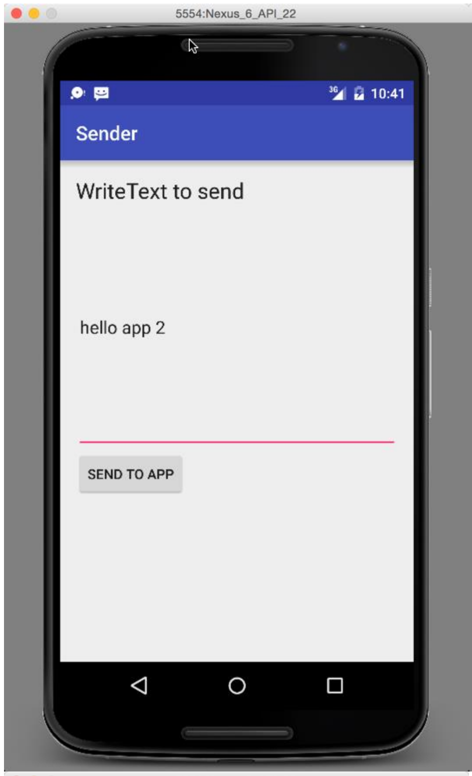
Comment hello app2
(java.lang.String)

hello app 2

To fix this problem:

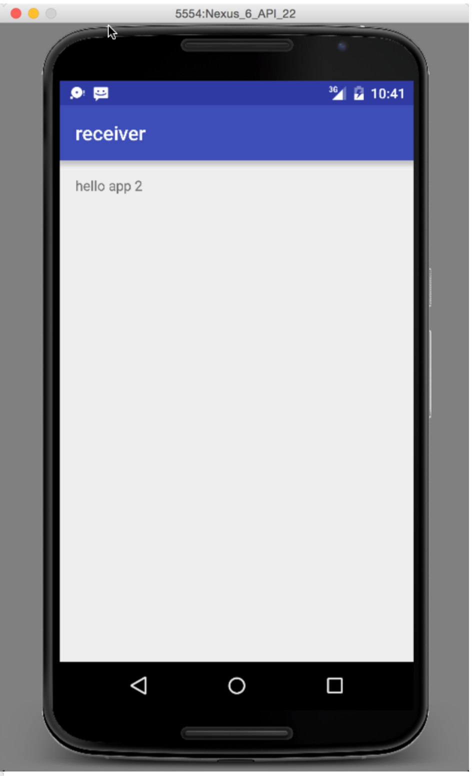
To fix this problem use encryption and decryption theory. That is mean that in sender app we encrypt the data before send it over bundle. Then in receiver app we decrypt that bundle data so the hacker app even could access to the data, it could not understand it.

See the example below. We can see that the Receiver app is unable to understand.

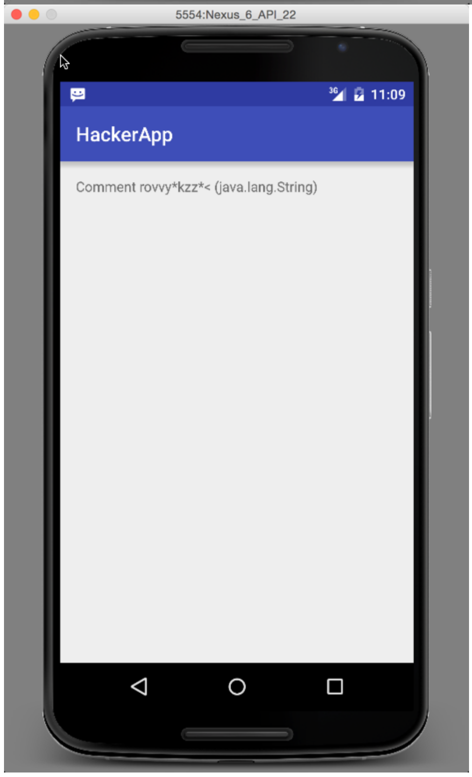


WriteText to send

hello app 2



hello app 2



Comment rovvy*kzz*< (java.lang.String)