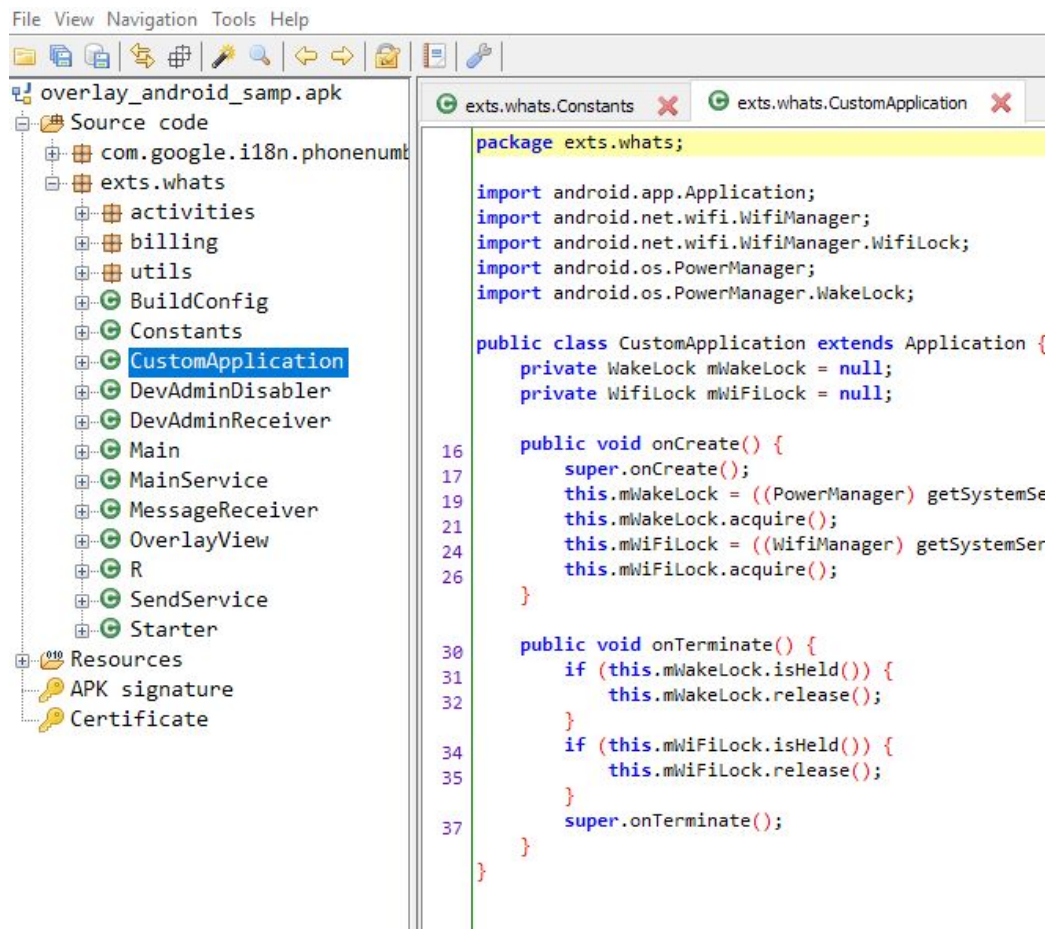


# Documentation Task 1

Imagine you are an expert in taint analysis and you want to create a new benchmark suite for evaluating your Android taint analysis tool. For this purpose, you have to document taint flows in an Android apk. Because we don't have so much time in a user study for you to discover taint flows by yourself, you will be given taint flows found by us.

You are given an Android apk **overlay\_android\_samp.apk** and a tool called **Jadx**. **Jadx** is a decompiler which can decompile Android apks and display decompiled source code in a window as shown below:



The screenshot shows the Jadx decompiler interface. On the left, a file tree for 'overlay\_android\_samp.apk' is visible, with 'Source code' expanded to show the package 'exts.whats'. The class 'CustomApplication' is selected and highlighted in blue. On the right, the decompiled source code for 'CustomApplication' is displayed in a window titled 'exts.whats.CustomApplication'. The code is as follows:

```
package exts.whats;

import android.app.Application;
import android.net.wifi.WifiManager;
import android.net.wifi.WifiManager.WifiLock;
import android.os.PowerManager;
import android.os.PowerManager.WakeLock;


public class CustomApplication extends Application {
    private WakeLock mWakeLock = null;
    private WifiLock mWifiLock = null;

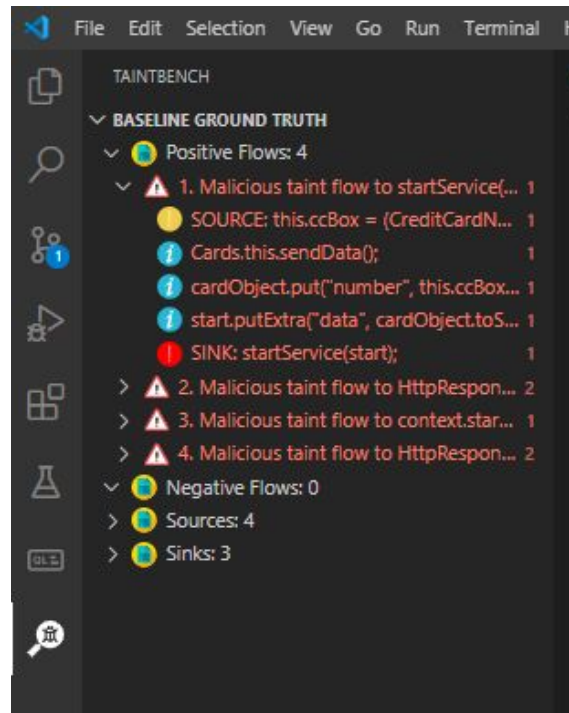
    16     public void onCreate() {
    17         super.onCreate();
    19         this.mWakeLock = ((PowerManager) getSystemService(
    21             Context.POWER_SERVICE)).newWakeLock(2, "exts.whats.CustomApplication");
    24         this.mWifiLock = ((WifiManager) getSystemService(
    26             Context.WIFI_SERVICE)).newWifiLock(2, "exts.whats.CustomApplication");
    }

    30     public void onTerminate() {
    31         if (this.mWakeLock.isHeld()) {
    32             this.mWakeLock.release();
    34         }
    35         if (this.mWifiLock.isHeld()) {
    36             this.mWifiLock.release();
    37         }
    38         super.onTerminate();
    }
}
```

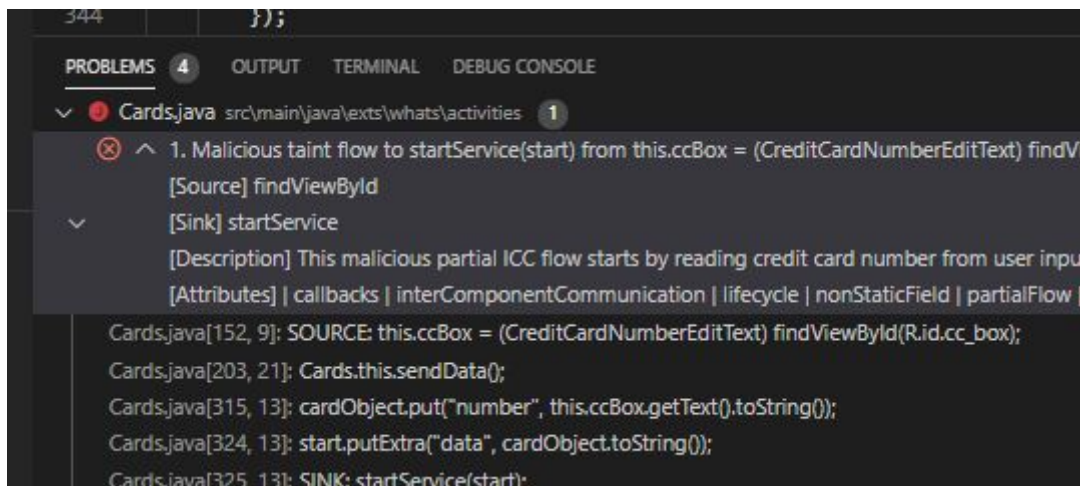
## Your Task:

1. Open **overlay\_android\_samp.apk** in **Jadx**.
2. Start Visual Studio Code, enable the extension TB-Viewer in Visual Studio Code if you have it disabled. Restart Visual Studio Code.
3. In Visual Studio Code, click **File >> Open Workspace >> Select myworkspace.code-workspace** in the folder **overlay\_android\_samp**

- Click  on the side panel, you will be shown 4 taint flows in the following screenshot:



- Open the json file **jadx.json** in Visual Studio Code or any editor you would like to use. All information you need about the taint flows can be found in the PROBLEMS view of Visual Studio Code as shown below:



- Find the positions of the taint flows with FlowID **3 and 4** in **Jadx** and document them in **jadx.json** using the format on the next page.
- Tell me when you start!**
- Tell me when you finish!**
- send me jadx.json.**

```

{
  "fileName": "example.apk",
  "findings": [
    {
      "source": {
        "statement": "String s = getIMEI();",
        "methodName": "public void onCreate()",
        "className": "com.example.MainActivity",
        "lineNo": 6,
        "targetName": "getIMEI",
        "targetNo": 1
      },
      "sink": {
        "statement": "sendTextMessage(Logger.imei);",
        "methodName": "public void send(String msg)",
        "className": "com.example.MainActivity",
        "lineNo": 10,
        "targetName": "sendTextMessage",
        "targetNo": 1
      },
      "intermediateFlows": [ {
        "statement": "s = \"IMEI: \" + s;",
        "methodName": "public void onCreate()",
        "className": "com.example.MainActivity",
        "lineNo": 7
      }, {
        "statement": "Logger.imei = s;",
        "methodName": "public void onCreate()",
        "className": "com.example.MainActivity",
        "lineNo": 8
      } ],
      "description": "This malicious flow sends the device' id to a premium number",
      "attributes": {
        "staticField": true,
        "appendToString": true
      }
    },
    {
      // another flow
    },
    ...
  ]
}

```

### Additional Explanation:

**lineNo:** Bytecode line number, displayed in Jadx

**targetName:** This is the name of the sensitive API

**targetNo:** If the sensitive API appears multiple times in a line, this specifies which one it is from left to right in the source code. It is 1 for the source, since getIMEI only appears once in the source statement.