

Basic Java and Android Programming for kBeacon

Casper LI @ Bid Dipper Studio



Introduction to Android

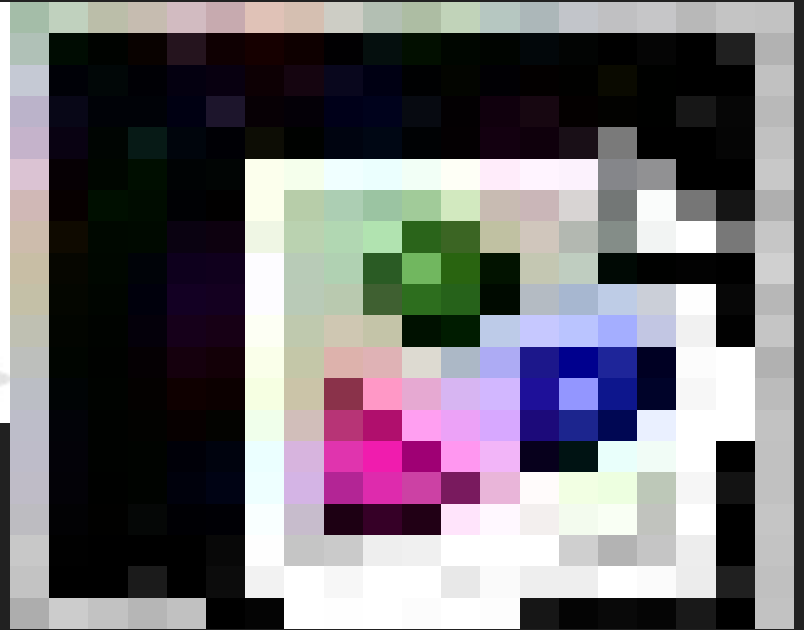


The “Software” running in your mobile computing device.

Computing Devices



Any machine that can store programs and execute programs and give out the results.



Linux and Android

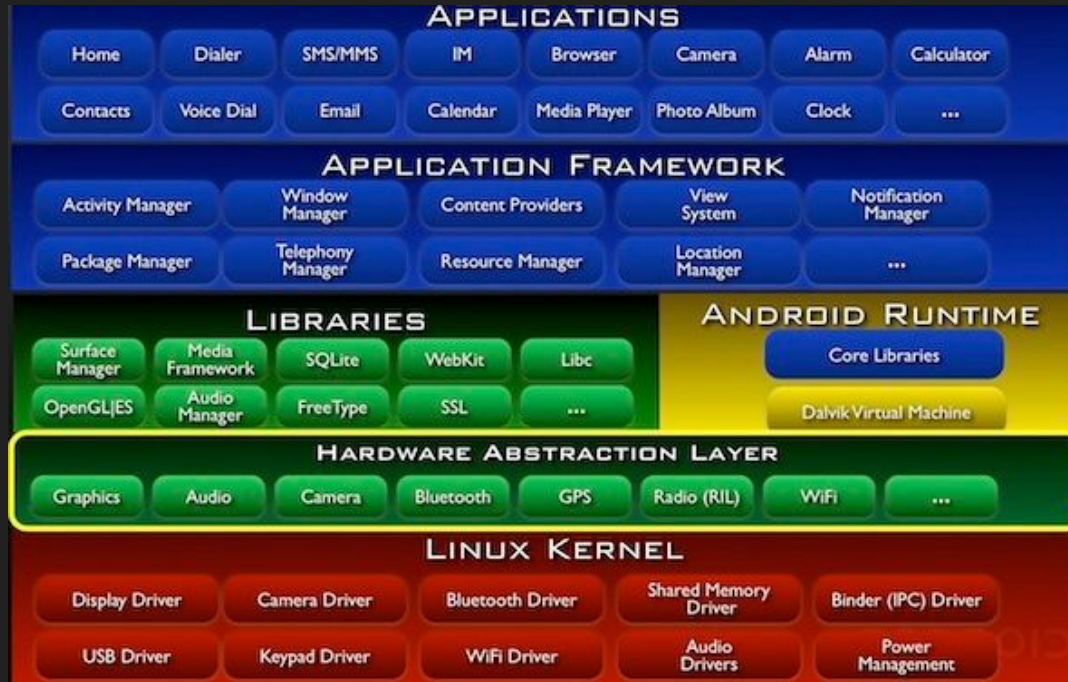


Android is a large software running on top of **Linux**.



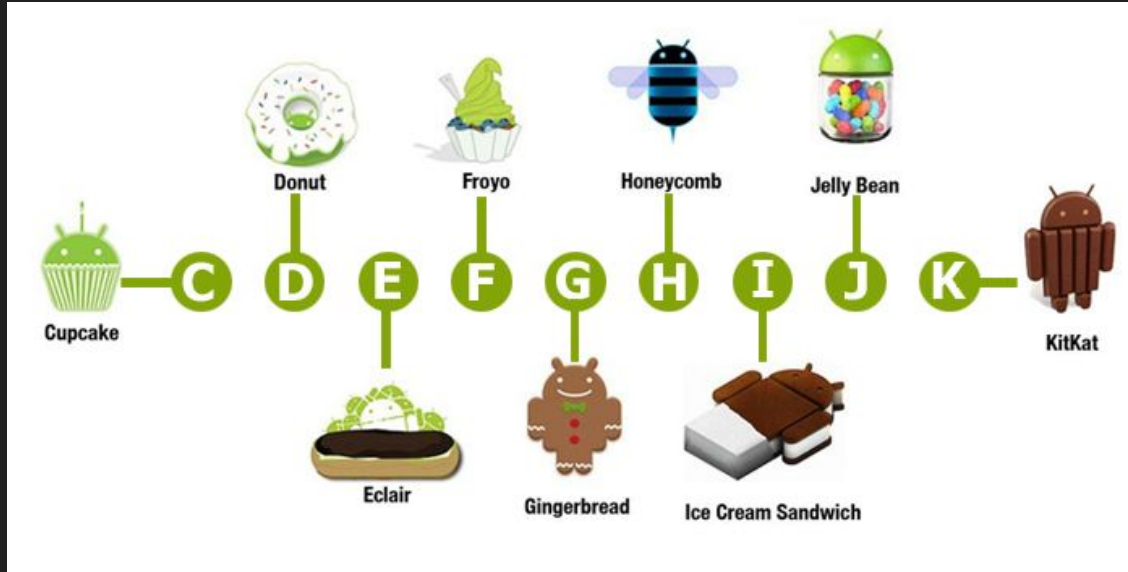
Just a Simplified Version

Android Architecture

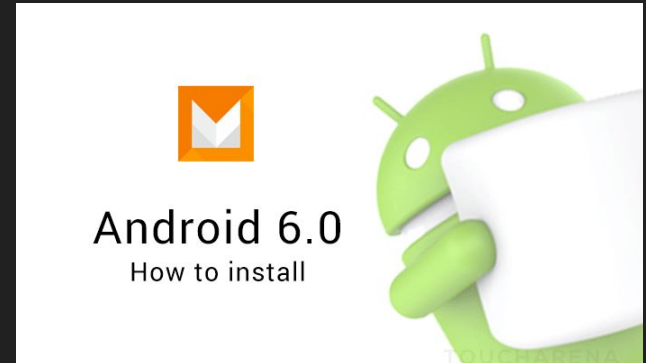
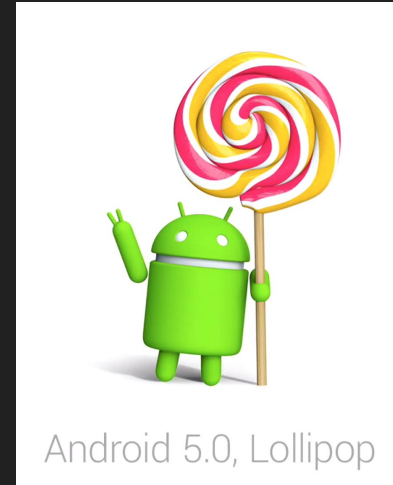


Also a simplified version. The actual implementation is much much more complicated. But it is **Not Required to Understand** in current state.

Android Version



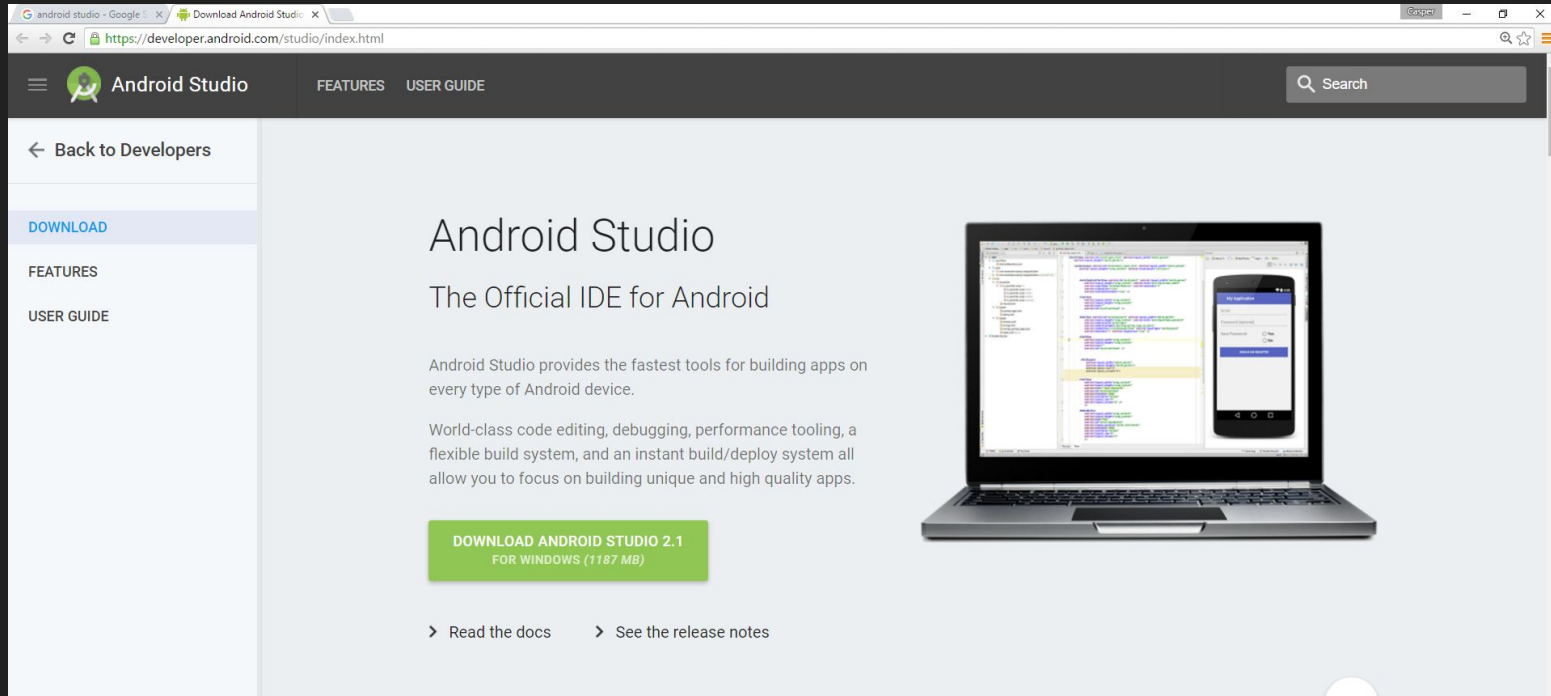
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, ...





Go into the Programming World!

Android Studio



The screenshot shows the official Android Studio website in a web browser. The browser's address bar displays the URL `https://developer.android.com/studio/index.html`. The website's header includes the Android Studio logo, navigation links for 'FEATURES' and 'USER GUIDE', and a search bar. A left sidebar contains a 'Back to Developers' link and a menu with 'DOWNLOAD', 'FEATURES', and 'USER GUIDE'. The main content area features the title 'Android Studio' and the subtitle 'The Official IDE for Android'. Below this, a paragraph states: 'Android Studio provides the fastest tools for building apps on every type of Android device.' Another paragraph describes the IDE's capabilities: 'World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.' A prominent green button reads 'DOWNLOAD ANDROID STUDIO 2.1 FOR WINDOWS (1187 MB)'. At the bottom of the main content, there are links to 'Read the docs' and 'See the release notes'. To the right of the text is a laptop displaying the Android Studio interface, which includes a code editor, a project explorer, and a virtual device preview.



android studio



Learn Android Programming (Java) Step by Step



Welcome to Android Studio

iBeaconLocator

E:\Users\User\Doc...0.3\iBeaconLocator

E:\Users\User\Documents\GitHub\iBeaconLoc

E:\Users\User\Doc...0.2\iBeaconLocator

BeaconScanner

E:\Users\User\Des...lder\BeaconScanner

beacon

E:\Users\User\Desktop\beacon

E:\Users\User\Desktop\BeaconScanner

E:\Users\User\Desktop\BeaconScanner




Android Studio


Version 2.1.2



 Start a new Android Studio project

 Open an existing Android Studio project


 Check out project from Version Control ▾

 Import project (Eclipse ADT, Gradle, etc.)

 Import an Android code sample

 Configure ▾  Get Help ▾

Create New Project

 **New Project**
Android Studio

Configure your new project

Application name:

Company Domain:

Package name: [Edit](#)

Project location: ...



Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available.

By targeting API 15 and later, your app will run on approximately **97.4%** of the devices

that are active on the Google Play Store.

[Help me choose](#)

☐ Wear

Minimum SDK API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK API 21: Android 5.0 (Lollipop)

☐ Android Auto

☐ Glass

Minimum SDK Glass Development Kit Preview (API 19)

Previous

Next

Cancel

Finish



Add an Activity to Mobile



Add No Activity



Basic Activity



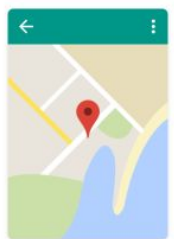
Empty Activity



Fullscreen Activity



Google AdMob Ads Activity



Google Maps Activity



Login Activity



Master/Detail Flow



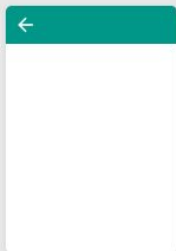
Navigation Drawer Activity



Scrolling Activity



Customize the Activity



Empty Activity

Creates a new empty activity

Activity Name: MainActivity

☒ Generate Layout File

Layout Name: activity_main

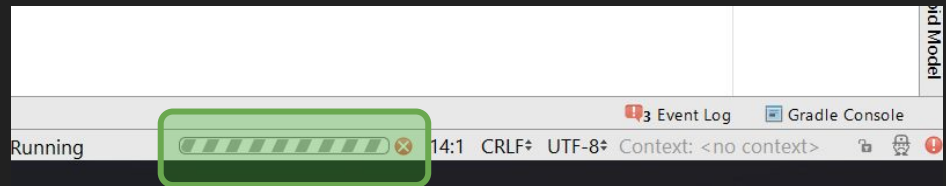
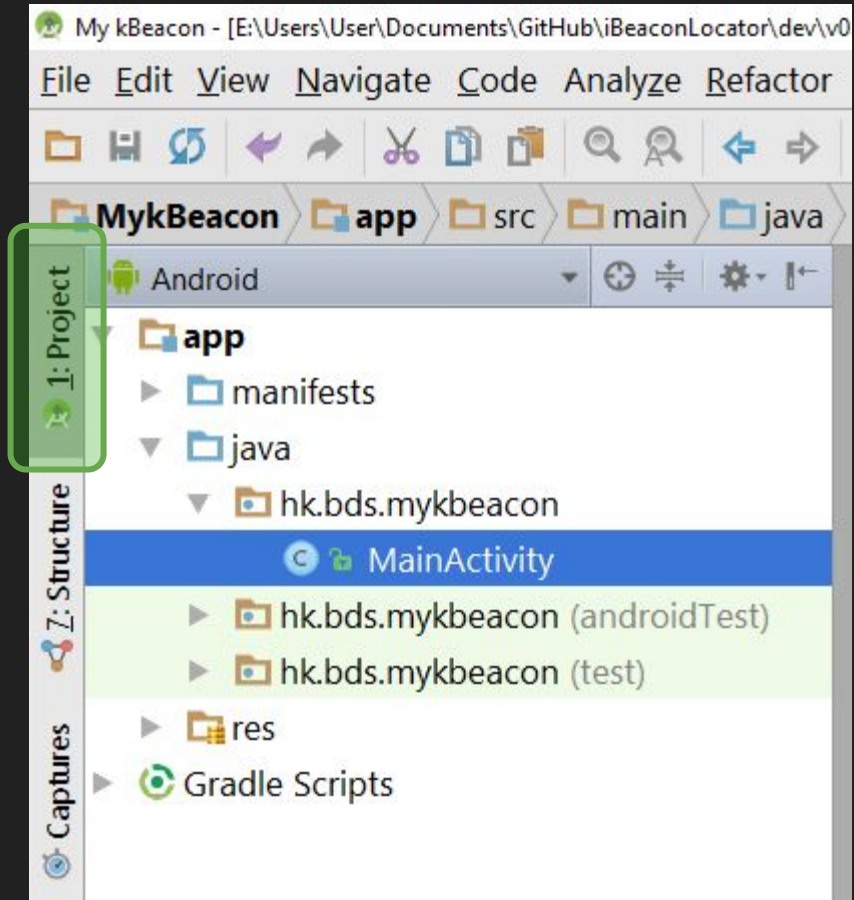
The name of the activity class to create

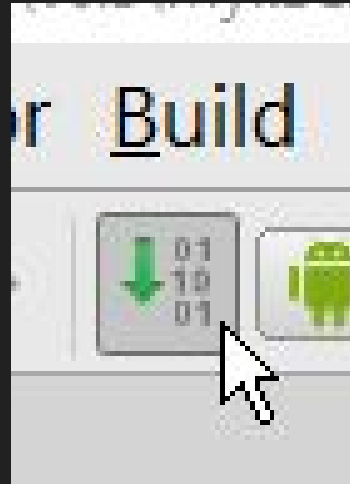
Previous

Next

Cancel

Finish





Try to Build!

Messages Gradle Build

- :app:generateDebugAndroidTestResources UP-TO-DATE
- :app:mergeDebugAndroidTestResources UP-TO-DATE
- :app:processDebugAndroidTestResources UP-TO-DATE
- :app:generateDebugAndroidTestSources UP-TO-DATE
- :app:mockableAndroidJar UP-TO-DATE
- :app:preDebugUnitTestBuild UP-TO-DATE
- :app:prepareDebugUnitTestDependencies
- :app:incrementalDebugJavaCompilationSafeguard UP-TO-DATE
- :app:compileDebugJavaWithJavac UP-TO-DATE
- :app:compileDebugNdk UP-TO-DATE
- :app:compileDebugSources UP-TO-DATE
- :app:incrementalDebugAndroidTestJavaCompilationSafeguard UP-TO-DATE
- :app:compileDebugAndroidTestJavaWithJavac UP-TO-DATE
- :app:compileDebugAndroidTestNdk UP-TO-DATE
- :app:compileDebugAndroidTestSources UP-TO-DATE
- :app:incrementalDebugUnitTestJavaCompilationSafeguard UP-TO-DATE
- :app:compileDebugUnitTestJavaWithJavac UP-TO-DATE
- :app:processDebugJavaRes UP-TO-DATE
- :app:processDebugUnitTestJavaRes UP-TO-DATE
- :app:compileDebugUnitTestSources UP-TO-DATE

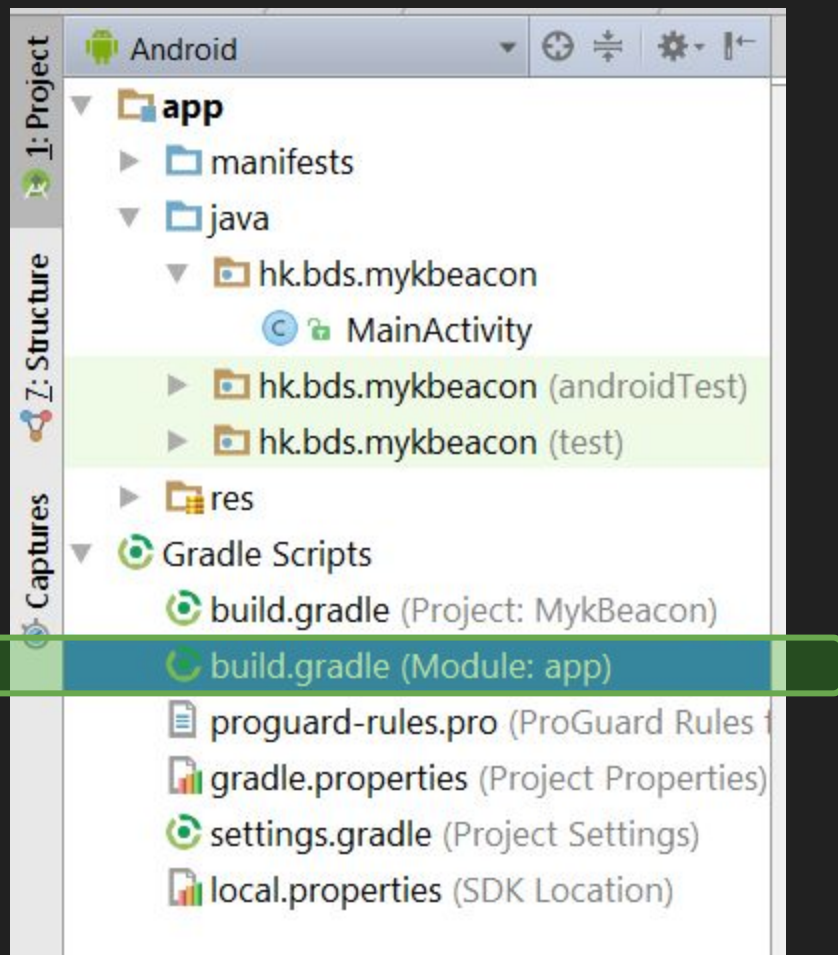
BUILD SUCCESSFUL

- Total time: 1.528 secs
- 0 errors
- 0 warnings
- [See complete output in console](#)

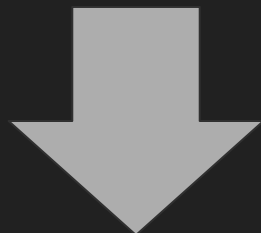
Terminal Android Monitor Messages TODO

Gradle build finished in 1s 560ms (moments ago)

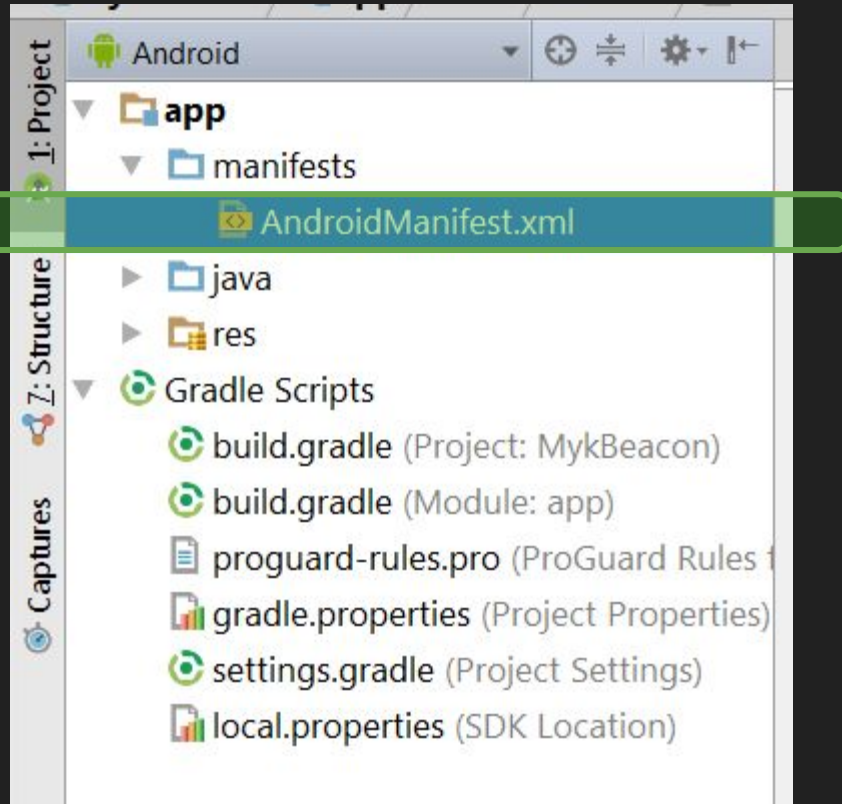
Everytime when you finished any little subsection of your program, verify whether it is bug free!



```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    testCompile 'junit:junit:4.12'  
    compile 'com.android.support:appcompat-v7:24.0.0'  
}
```



```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    testCompile 'junit:junit:4.12'  
    compile 'com.android.support:appcompat-v7:24.0.0'  
    compile 'org.altbeacon:android-beacon-library:2+' // This is the dependency of Android Beacon Library  
}
```



Get the permission to access the bluetooth and location services in your android phone.

manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="hk.bds.ibeaconlocator" >

    <uses-sdk
        android:minSdkVersion="18"
        android:targetSdkVersion="18" />

    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

<application

```
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="iBeaconLocator"
    android:supportRtl="true"
    android:theme="@style/AppTheme" >
```

<activity android:name=".MainActivity" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

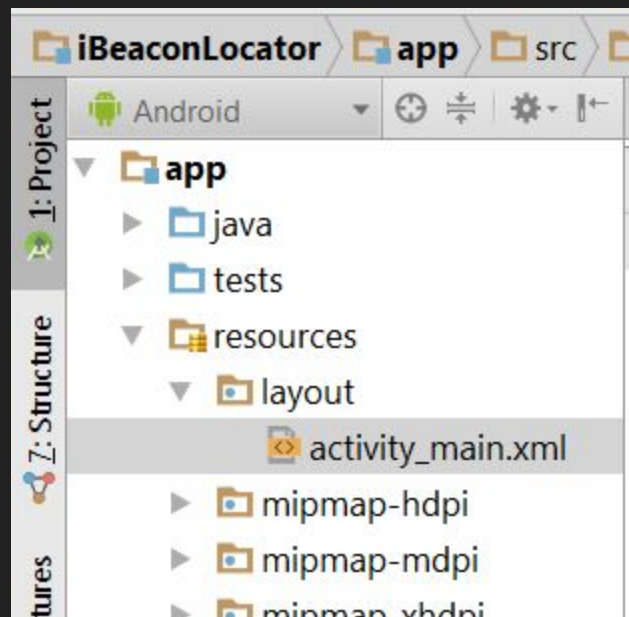
<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

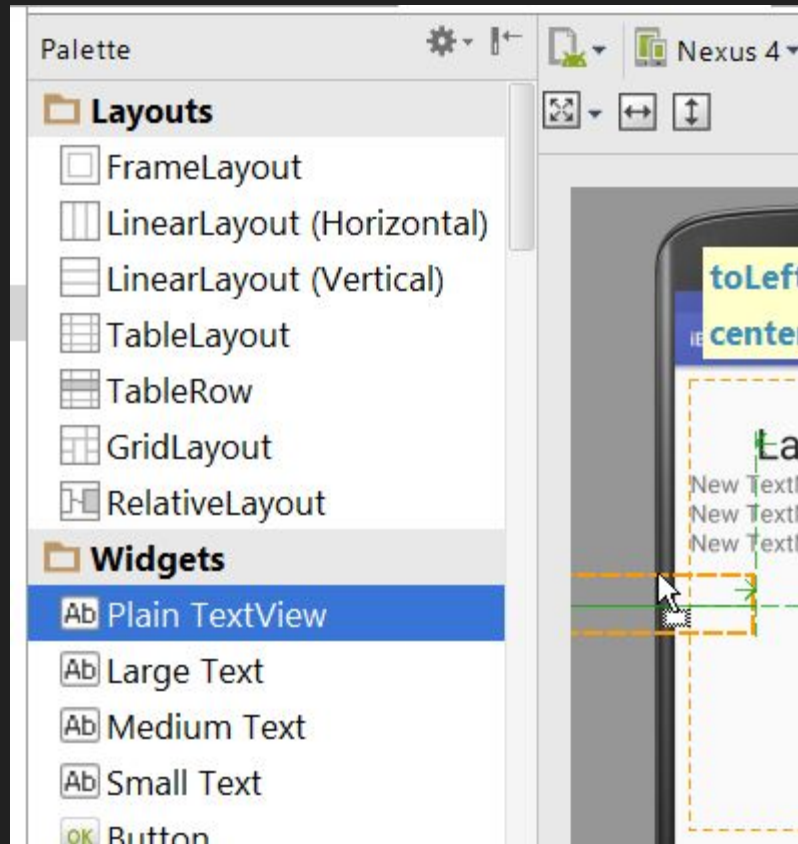
</activity>

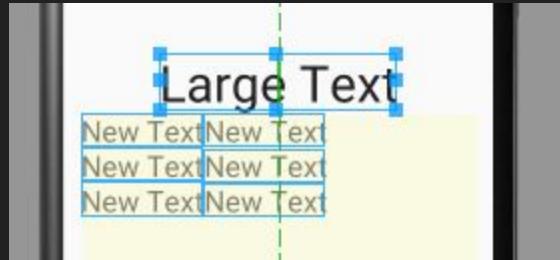
</application>

</manifest>

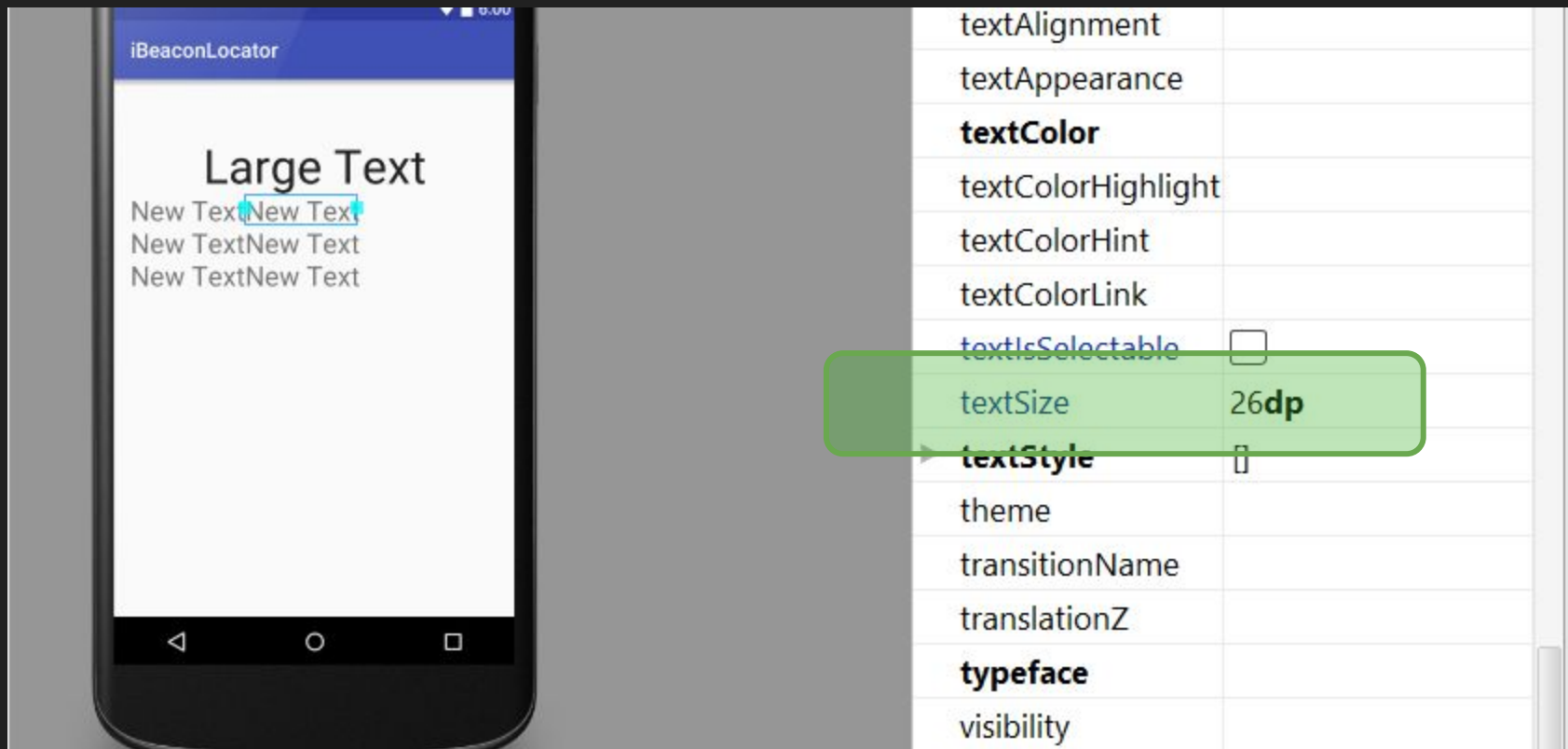


Drag and drop the
TextView to the screen
of the virtual android
phone.

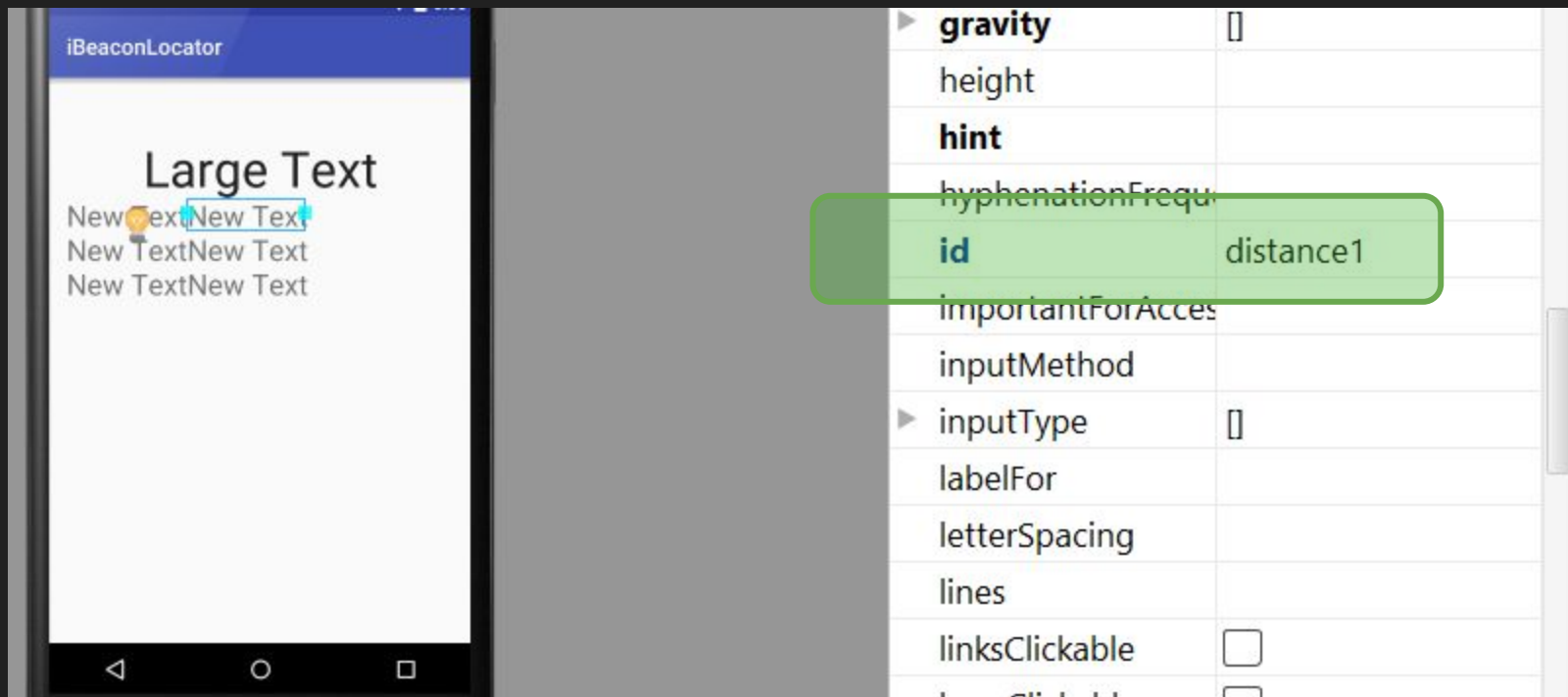




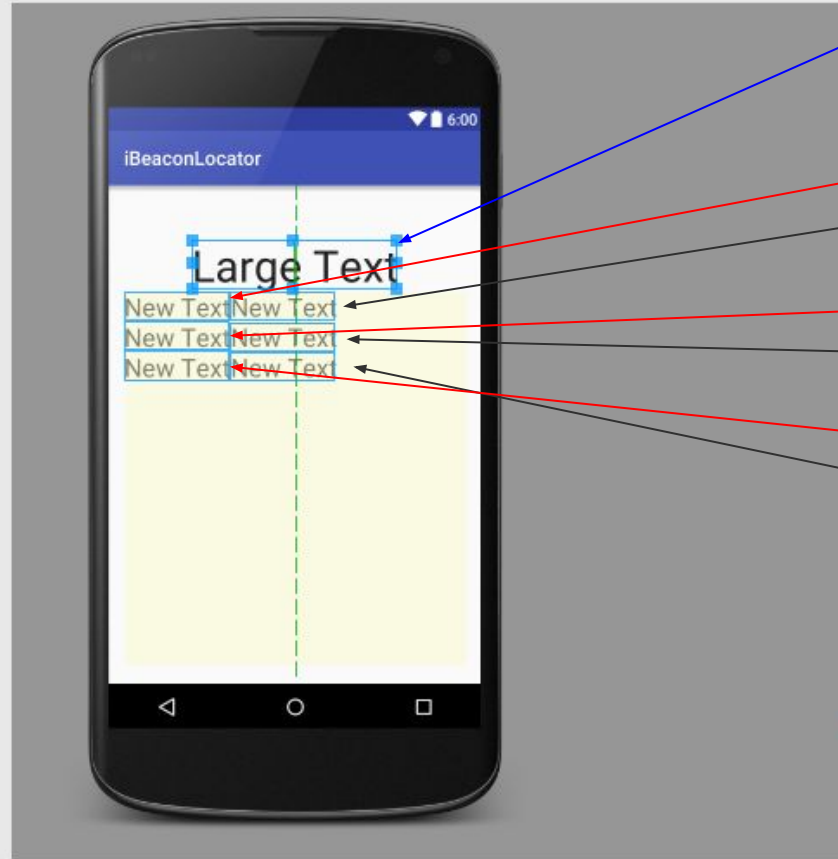
6 TextViews



You can try to do some changes to the appearance in the property



The id of each element (e.g. TextView) is important for program to access it.



Component Tree

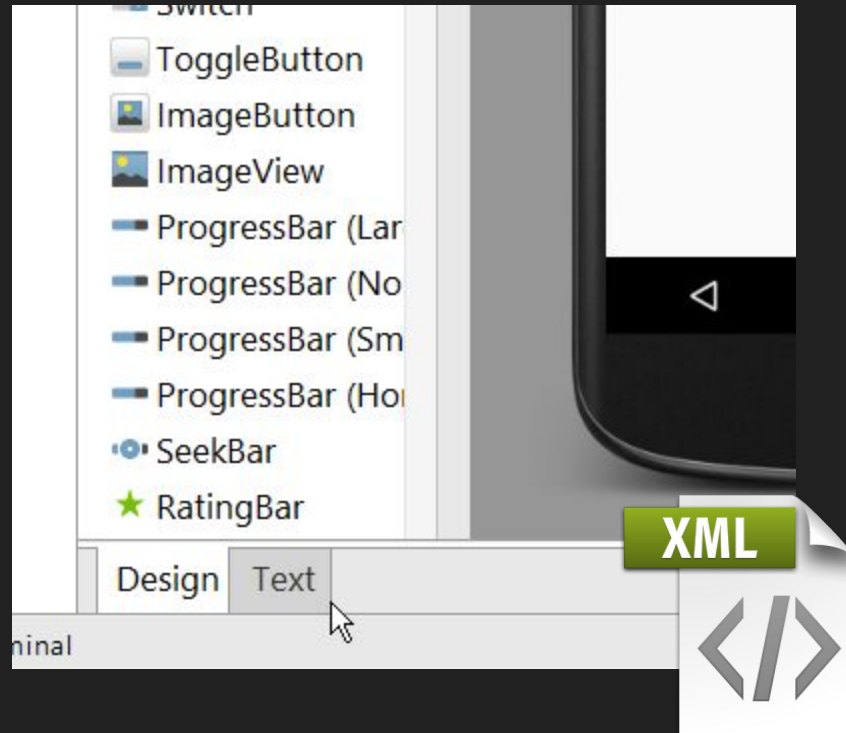
- Device Screen
 - RelativeLayout
 - location** (TextView) - "Large Text"
 - TableLayout
 - TableRow
 - beacon1** (TextView) - "New"
 - distance1** (TextView) - "Ne"
 - TableRow
 - beacon2** (TextView) - "New"
 - distance2** (TextView) - "Ne"
 - TableRow
 - beacon3** (TextView) - "New"
 - distance3** (TextView) - "Ne"

Property

stateListAnimator

text

Please rename the id like this.



You can view the XML source.

Actually, UI programmers prefer to use “Text” mode.

```

+import ...

public class MainActivity extends AppCompatActivity implements BeaconConsumer {

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

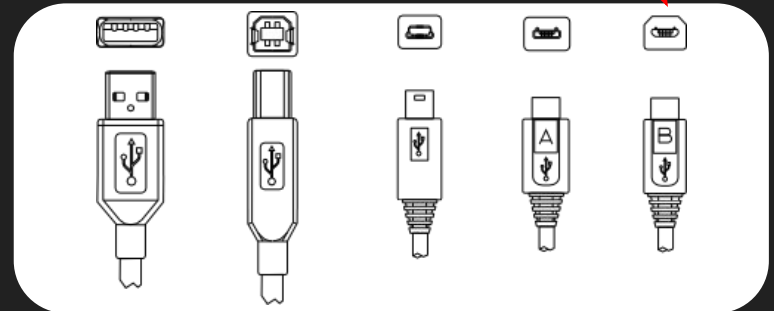
```

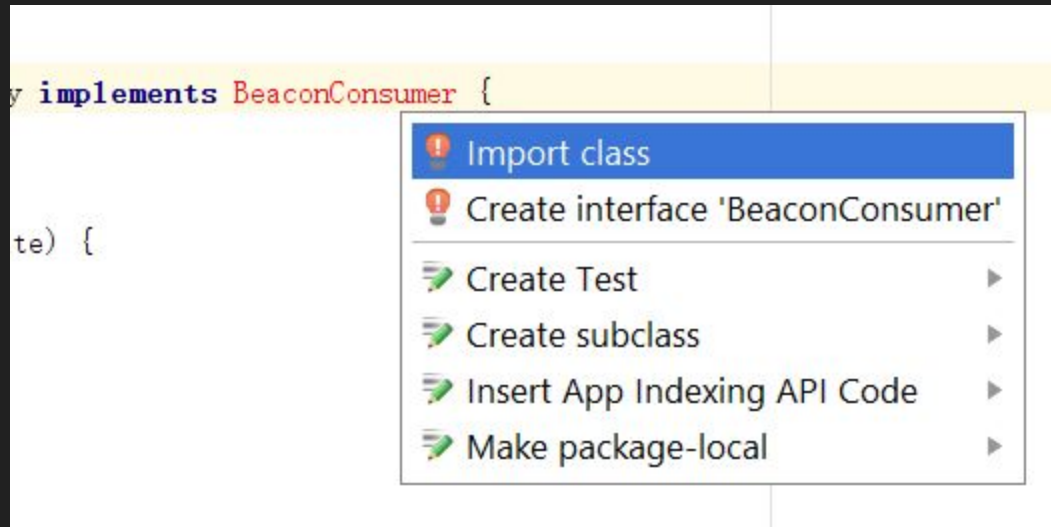


This is called an
“interface”

If you launch your app, the codes inside
will be executed.

But if you just come back to the app, the
codes will not be executed





Android Studio does not know what is “BeaconConsumer”.

[ALT] + [ENTER] to import class in order to let Android Studio know what is this.

```
package hk.bds.mykbeacon;

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

import org.altbeacon.beacon.BeaconConsumer;

public class MainActivity extends AppCompatActivity implements BeaconConsumer {

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

Now you imported the package of “BeaconConsumer”.

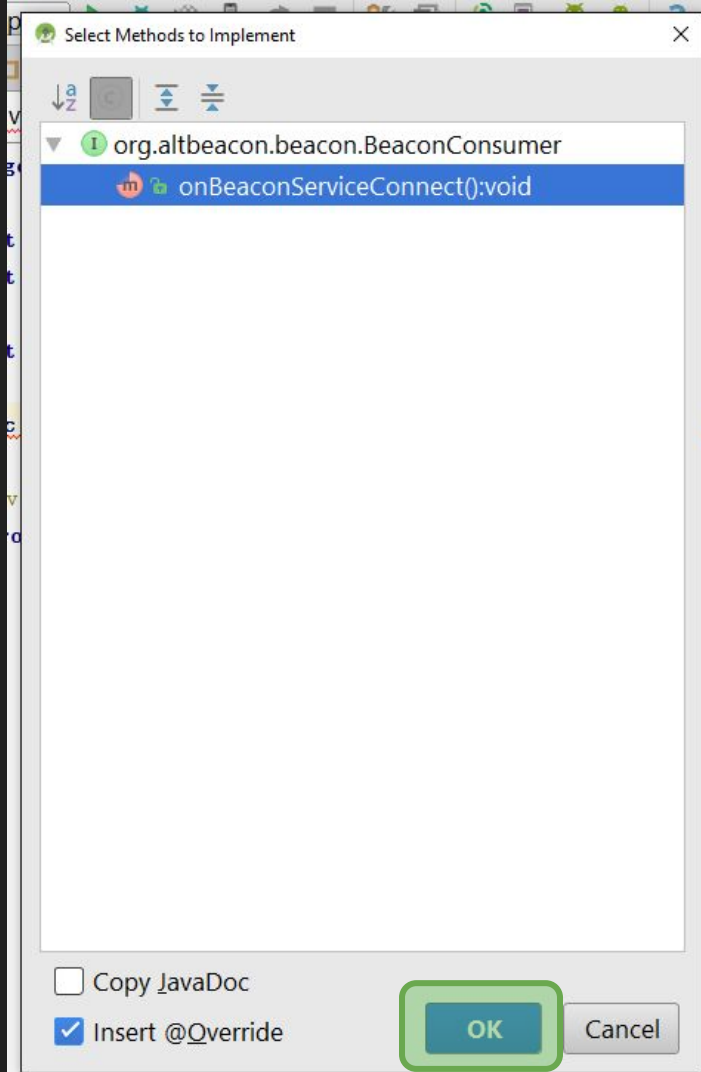
You can also type it by yourself. The effect is the same.


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

- Implement methods
- Make 'MainActivity' abstract
- Create Test
- Create subclass
- Insert App Indexing API Code
- Make package-local

Since you said you will make the interface “BeaconConsumer”, you need to implement it.

[ALT] + [ENTER] to implement method



For “BeaconConsumer”, you need to make this public method “onBeaconServiceConnect()”

public means that you can let other programs call this method in your program.

```
@Override  
💡 public void onBeaconServiceConnect() {  
  
}
```

This **method** is now created.

And we need to implement the content of this method.

Now we want to connect the UI to your program.

```
public class MainActivity extends AppCompatActivity implements BeaconConsumer {
```

```
    private TextView    displayName1,  
                        displayName2,  
                        displayName3,  
                        displayDistance1,  
                        displayDistance2,  
                        displayDistance3,  
                        displayLocation;
```

```
@Override
```

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

Now we need to declare the TextView objects.

```
private TextView displayName1
```

A diagram illustrating a variable declaration. A light gray rounded rectangle contains the code 'private TextView displayName1'. A white arrow points from the right side of this rectangle to a large red question mark on the right side of the slide.

Actually you have a pointer but it points to nothing.
You need to get the actual TextView object for it.

```
private TextView displayName1
```



A diagram illustrating the relationship between a variable and an object. On the left, a light gray rounded rectangle contains the code `private TextView displayName1`. A white arrow points from this rectangle to a light gray 3D box on the right. The box contains the text "I am a TextView object.".

I am a TextView
object.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Init displays
    setContentView(R.layout.activity_main);
    displayName1 = (TextView) findViewById(R.id.beacon1);
    displayName2 = (TextView) findViewById(R.id.beacon2);
    displayName3 = (TextView) findViewById(R.id.beacon3);
    displayDistance1 = (TextView) findViewById(R.id.distance1);
    displayDistance2 = (TextView) findViewById(R.id.distance2);
    displayDistance3 = (TextView) findViewById(R.id.distance3);
    displayLocation = (TextView) findViewById(R.id.location);
}
```

Now we need to get the TextView objects.


```
displayDistance1,  
displayDistance2,  
displayDistance3,  
displayLocation;
```

```
private BeaconManager beaconManager;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState)  
    super.onCreate(savedInstanceState);
```

```
displayName2 = (TextView) findViewById(R.id.beacon2);  
displayName3 = (TextView) findViewById(R.id.beacon3);  
displayDistance1 = (TextView) findViewById(R.id.distance1);  
displayDistance2 = (TextView) findViewById(R.id.distance2);  
displayDistance3 = (TextView) findViewById(R.id.distance3);  
displayLocation = (TextView) findViewById(R.id.location);
```

```
// Init Beacon
```

```
beaconManager = BeaconManager.getInstanceForApplication(this);  
beaconManager.getBeaconParsers().add(new BeaconParser().setBeaconLayout("m:2-3=0215,i:4-19,i:20-21,i:22-23,p:24-24"));  
RangedBeacon.setSampleExpirationMilliseconds(1100); // The refresh interval  
Beacon.setHardwareEqualityEnforced(true);  
beaconManager.setBackgroundBetweenScanPeriod(20);  
beaconManager.setForegroundBetweenScanPeriod(20);  
beaconManager.bind(this);
```

```
});  
  
}  
  
@Override  
protected void onDestroy() {  
    super.onDestroy();  
    beaconManager.unbind(this);  
}  
  
@Override  
public void onBeaconServiceConnect() {
```

When you terminate the app, the codes inside will be executed.

```

protected void onDestroy() {
    super.onDestroy();
    beaconManager.unbind(this);
}

@Override
public void onBeaconServiceConnect() {
    //This method will be called when the Beacon Manager is binded.
    beaconManager.setRangeNotifier( new RangeNotifier() {
        @Override
        public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {
            // This method will be executed many times according to the size of the refresh interval.
            // Try to update the distance of the devices
            if (beacons.size() > 0) {
                // Do something
            }
        }
    });

    try {
        // Tells the BeaconService to start looking for beacons that match the passed Region object,
        // and providing updates on the estimated mDistance every seconds while beacons in the Region are visible.
        beaconManager.startRangingBeaconsInRegion(new Region("myRangingUniqueId", null, null, null));
    } catch (RemoteException e) { /* Error is detected. */ }
}
}

```

```
// Try to update the distance of the devices
if (beacons.size() > 0) {
    // Do something
    for (Beacon b : beacons ) {
        // Access all detected beacons, one per a loop
        double distance = b.getDistance(); // Get the distance
        String macAddress = b.getBluetoothAddress(); // Get the beacon MAC address
    }
}
```

Now you can get the distances and MAC addresses of the bluetooth devices.

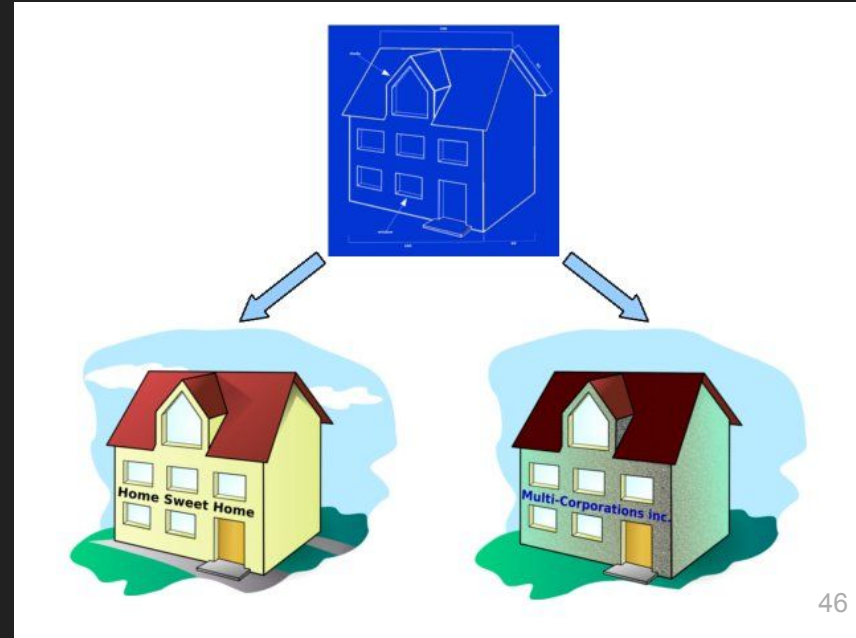
Now I want to determine the nearest kBeacon, and display it out.

But the program will get more and more complicated.
So you have to create a data model

Object

Before you create any object, you need to have a blueprint and you use the blueprint to generate the object(s). The blueprint is called “class”

```
class House {  
  
    // Data stored inside  
  
    int height, width, length;  
  
    int num_of_people;  
  
    // Some methods for operating  
        the data inside and outside  
  
    // ...  
  
}
```



Create a Class “MyBeacon”

```
class MyBeacon {  
    public String name;  
    public String macAddress;  
    public double distance = 0d; // initially the distance is 0.  
    //public double predistance = 0d;  
  
    // Display reference pointers.  
    public TextView displayName;  
    public TextView displayDistance;  
  
    public MyBeacon(String _name, String _macAddress, TextView _displayName, TextView _displayDistance) {  
        name = _name;  
        macAddress = _macAddress;  
        displayName = _displayName;  
        displayDistance = _displayDistance;  
        displayName.setText(name + " : ");  
    }  
  
    public boolean updateDistance(Beacon _beacon) {  
        if (_beacon.getBluetoothAddress().equals(this.macAddress)) {  
            distance = _beacon.getDistance(); // Calculate the distance based on RSSI.  
            return true;  
        } else  
            return false;  
    }  
}
```

```

public boolean updateDistance(Beacon _beacon) {
    if (_beacon.getBluetoothAddress().equals(this.macAddress)) {
        distance = _beacon.getDistance(); // Calculate the distance based on RSSI.
        return true;
    } else
        return false;
}

```

```

public void updateDistance(Collection<Beacon> beacons) {
    for (Beacon theBeacon : beacons) {
        if (updateDistance(theBeacon))
            return;
    }
    distance = 0d;
}

```

```

public void updateDisplayDistance() {
    String str;
    if (distance == 0d)
        str = "UNDETECTED";
    else
        str = String.format("%.4f", distance);
}

```



```
}
```

```
public void updateDisplayDistance() {  
    String str;  
    if (distance == 0d)  
        str = "UNDETECTED";  
    else  
        str = String.format("%.4f", distance);  
    displayDistance.setText(str);  
}
```

```
public String toString() {  
    String str = "";  
    str += "\n=====";  
    str += "\nBeacon Name: " + this.name;  
    str += "\nMac Address: " + this.macAddress;  
    str += "\nDistance : " + this.distance;  
    str += "\n=====";  
    return str;  
}
```

Create Objects of the Class

```
private BeaconManager beaconManager;
```

```
private MyBeacon b1, b2, b3, theNear;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);
```

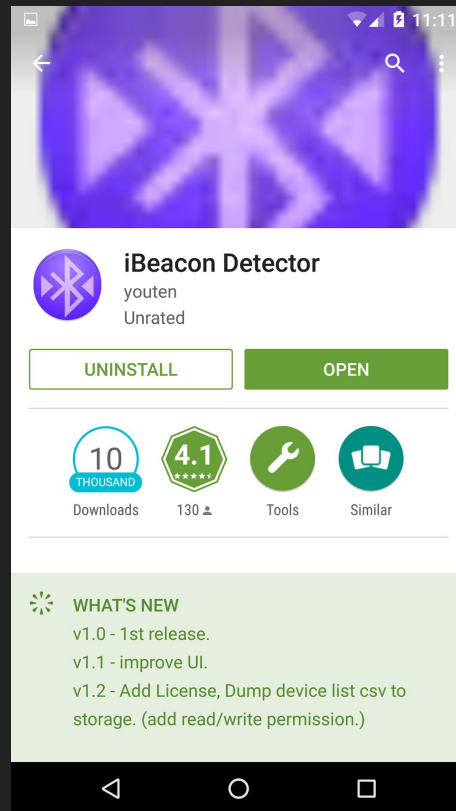
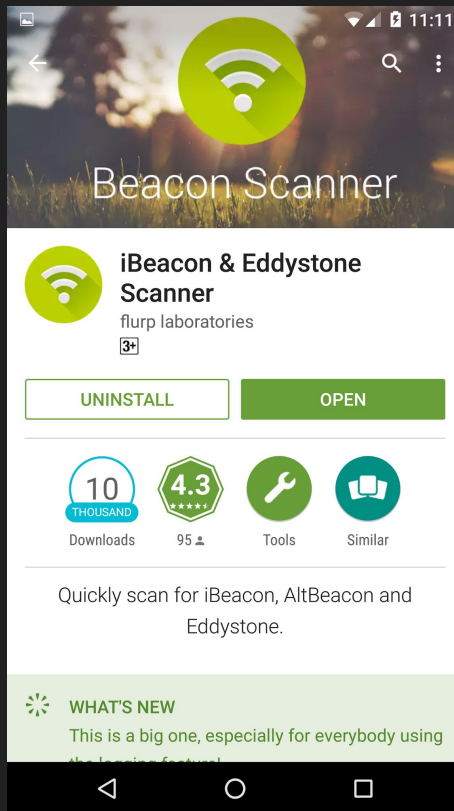
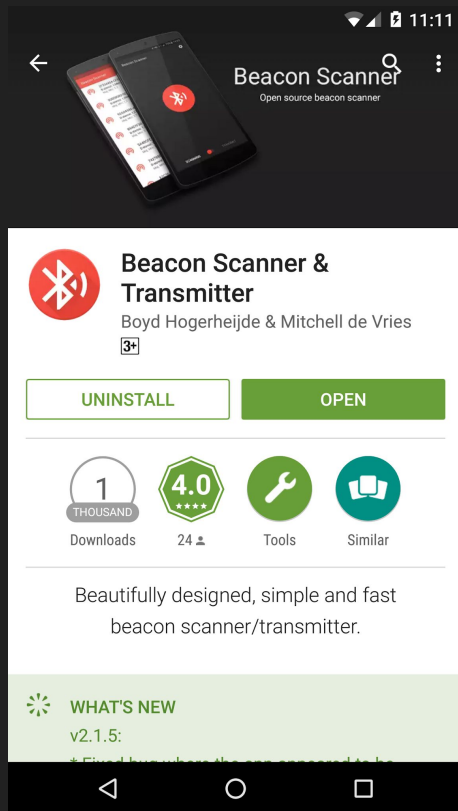
```
// Init Beacon
```

```
beaconManager = BeaconManager.getInstanceForApplication(this);  
beaconManager.getBeaconParsers().add(new BeaconParser().setBeaconLayout("m:2-3=0215,i:4-19,  
RangedBeacon.setSampleExpirationMilliseconds(1100); // The refresh interval  
beaconManager.setBackgroundBetweenScanPeriod(20);  
beaconManager.setForegroundBetweenScanPeriod(20);  
beaconManager.bind(this);
```

```
// Init beacon devices
```

```
b1 = new MyBeacon("Room S505", "BC:6A:29:25:0F:52", displayName1, displayDistance1);  
b2 = new MyBeacon("Room S506", "BC:6A:29:27:A4:2D", displayName2, displayDistance2);  
b3 = new MyBeacon("Room S507", "BC:6A:29:28:01:BD", displayName3, displayDistance3);
```

How to find out the MAC address of your beacon?



```

private MyBeacon getMinOne(MyBeacon a, MyBeacon b) {
    if (a != null && b != null) {
        if ((a.distance > 0d && b.distance > 0d) && !(a.distance == b.distance)) {
            if (b.distance < a.distance)
                return b;
            else
                return a;
        } else if ((a.distance <= 0d && b.distance <= 0d) || (a.distance == b.distance)) {
            return null;
        } else {
            if (a.distance == 0d)
                return b;
            else
                return a;
        }
    } else if ((a == null && b == null) || (a == null && b.distance <= 0d) || (b == null && a.distance <= 0d)) {
        return null;
    } else {
        if (a != null)
            return a;
        else
            return b;
    }
}

```

This method is used to find out the nearest MyBeacon from the 2 MyBeacon a and b.

```
// Try to update the distance of the devices
if (beacons.size() > 0) {
    // Do something
    for (Beacon b : beacons ) {
        // Access all detected beacons, one per a loop
        double distance = b.getDistance(); // Get the distance
        String macAddress = b.getBluetoothAddress(); // Get the beacon MAC address
    }
}
```

Delete those codes inside and we can do a little bit more.

```
public void didRangeBeaconsInRegion(Collection<Beacon> beacons, Region region) {  
    // This method will be executed many times according to the size of the refresh interval.  
    // Try to update the distance of the devices  
    b1.updateDistance(beacons);  
    b2.updateDistance(beacons);  
    b3.updateDistance(beacons);  
    theNear = getMinOne(getMinOne(b1, b2), b3);  
}
```

Update the distances of the MyBeacon objects

and

Find out the nearest one.

Actually, this is a question. We assume that the distance of $b1 > b2 > b3$
(b3 is the nearest one)

```
theNear = getMinOne(getMinOne(b1,b2),b3)
         = getMinOne(b2, b3)
         = b3
```

Now we know that theNear is b3.

```

        b3.updateDistance(beacons);
        theNear = getMinOne(getMinOne(b1, b2), b3);

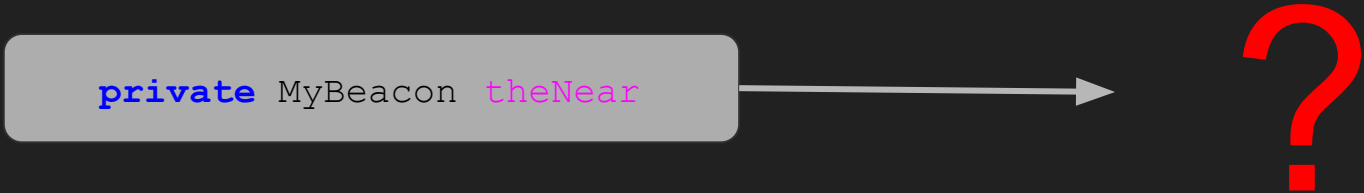
        // Update Displays
        MainActivity.this.runOnUiThread( new Runnable() {
            public void run() {
                if (theNear != null && theNear.distance < 0.35) {
                    // If the near beacon is inside the range, do this action
                    displayLocation.setText(theNear.name); // Update the largest text.
                } else {
                    // You are not close enough to the near one.
                    displayLocation.setText("UNCERTAIN");
                }
                // Update the distance display
                b1.updateDisplayDistance();
                b2.updateDisplayDistance();
                b3.updateDisplayDistance();
            }
        });
    }
}

```

Now we already get all required information.

We can display them to the UI.


```
private MyBeacon theNear
```



null

Since we cannot determine the nearest one,
the “getMinOne (getMinOne (b1, b2) , b3)” method give me “null”.

```
theNear = null
```

Now we make it speak what is the location of the nearest kBeacon detected.

Google TextToSpeech



```

        displayDistances,
        displayLocation;
        private BeaconManager beaconManager;
        private TextToSpeech ttobj;
        private MyBeacon previousLocation = null;

```

```

b2 = new MyBeacon("Room S506", "BC:6A:29:28:01:BD", displayName2,
b3 = new MyBeacon("Room S507", "BC:6A:29:28:01:BD", displayName3,

```

```

// Create Google TextToSpeech Object
ttobj = new TextToSpeech(getApplicationContext(), (status) -> {
    if(status != TextToSpeech.ERROR) {
        ttobj.setLanguage(Locale.UK); // Set to "UK" language
    }
});
}

```

Create an object of Google TextToSpeech service.

```
displayLocation,  
private BeaconManager beaconManager;  
private TextToSpeech ttobj;  
private MyBeacon previousLocation = null;  
  
@Override
```

We need an object to store the previous detected beacon.

This is used for determining the change of location.

```
MainActivity. this.runOnUiThread( new Runnable() {
```

```
    public void run() {
```

```
        if (theNear != null && theNear.distance < 0.35) {
```

```
            // If the near beacon is inside the range, do this action
```

```
            displayLocation.setText(theNear.name); // Update the largest text.
```

```
            if (theNear != previousLocation) {
```

```
                // If you just enter the area
```

```
                String toSpeak = "" + theNear.name;
```

```
                ttobj.speak(toSpeak, TextToSpeech.QUEUE_FLUSH, null); // Speak
```

```
            }
```

```
        } else {
```

```
            // You are not close enough to the near one.
```

```
            displayLocation.setText("UNCERTAIN");
```

```
            previousLocation = null;
```

```
        }
```

```
        previousLocation = theNear; // Store the previous location
```

```
        // Update the distance display
```

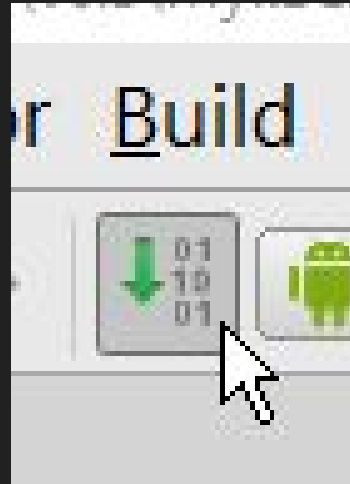
```
        b1.updateDisplayDistance();
```

```
        b2.updateDisplayDistance();
```

```
        b3.updateDisplayDistance();
```

```
    }
```














```
});
```








Try to Build!

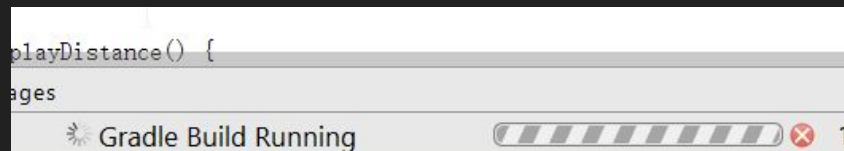
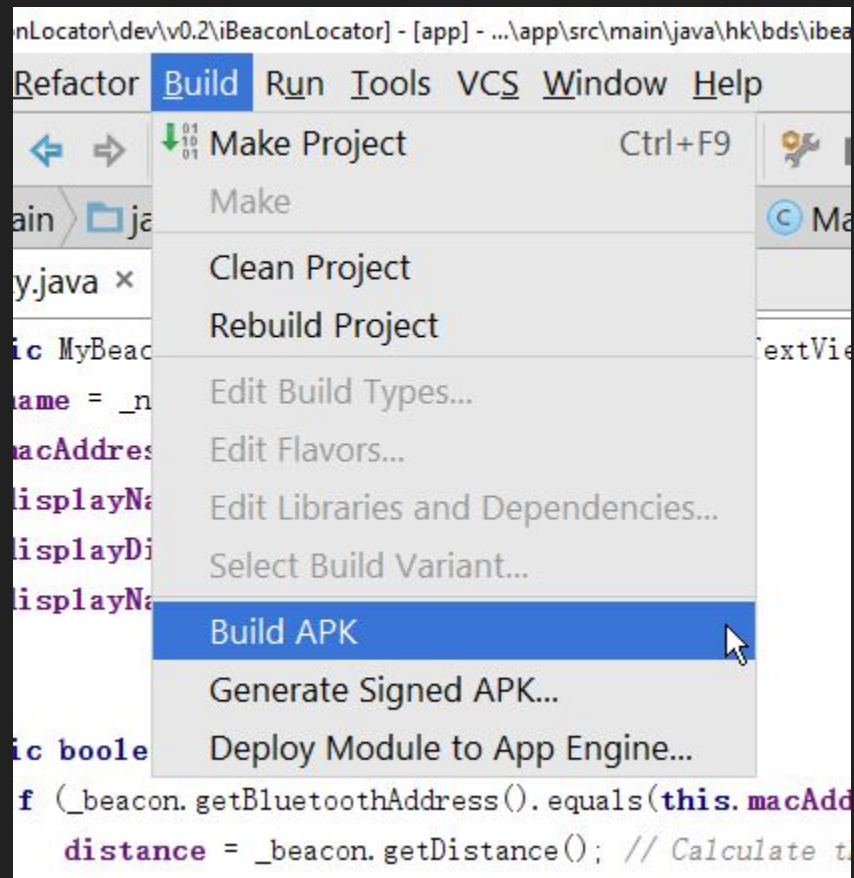
Messages Gradle Build

2: Favorites Build Variants

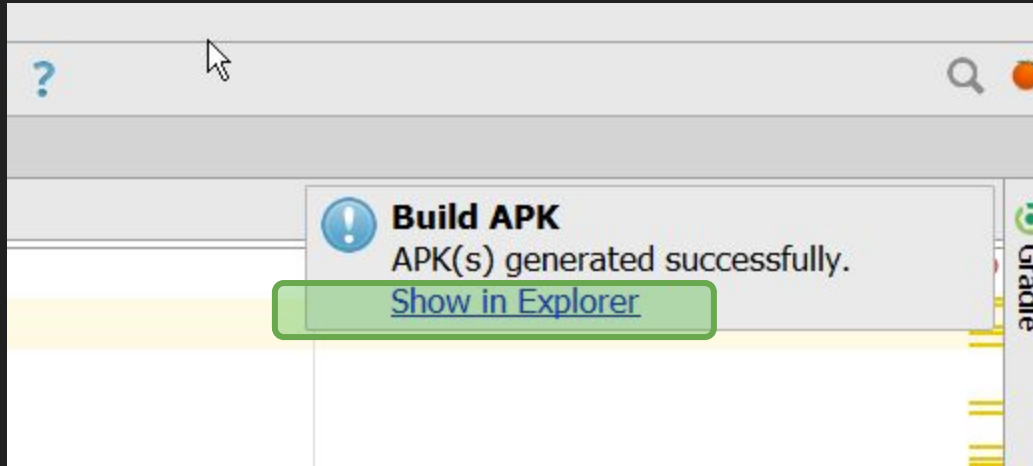
            

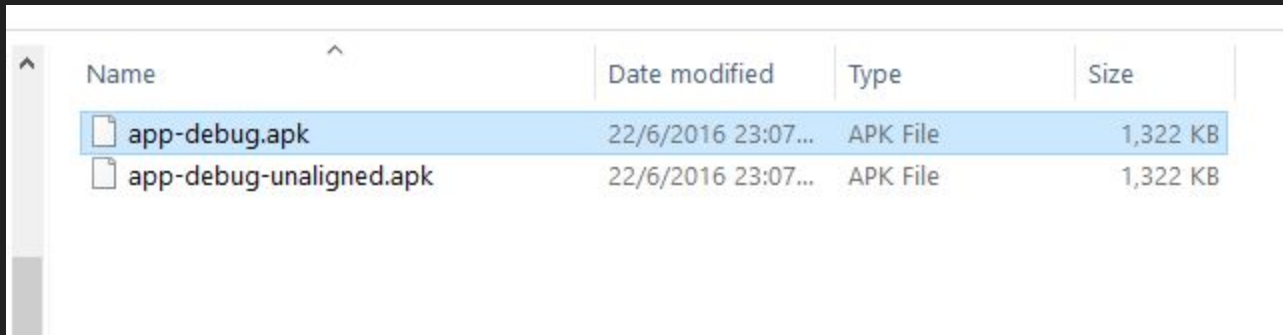
:app:processDebugUnitTestJavaRes UP-TO-DATE
:app:compileDebugUnitTestSources



-  BUILD SUCCESSFUL
-  Total time: 2.012 secs
-  0 errors
-  0 warnings
-  See complete output in console



Wait...





Name	Date modified	Type	Size
 app-debug.apk	22/6/2016 23:07...	APK File	1,322 KB
 app-debug-unaligned.apk	22/6/2016 23:07...	APK File	1,322 KB

Install it on your android device.



END