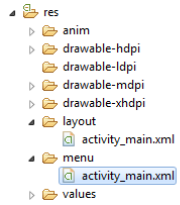
Using the onCreateOptionsMenu Method

Nowadays, Eclipse creates a res/menu folder when you create a new Android project. By default, the res/menu folder contains an xml file with the name activity_main.xml (the same as the default name for the automatically created layout file).



Here's a very simple activity_main.xml file. This file defines a menu containing two items. The "more important" item is the menu_start item, because the menu_start item has a lower android:orderInCategory value. (When Android displays menu items, it displays the more important items earlier in the list.) The android:showAsAction attribute tells Android never to put the item in the Action bar. (The Action bar is available from API Level 11 onward.)

activity_main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/menu_start"
        android:orderInCategory="1"
        android:showAsAction="never"
        android:title="@string/menu_start"/>

    <item
        android:id="@+id/menu_settings"
        android:orderInCategory="2"
        android:showAsAction="never"
        android:title="@string/menu_settings"/>

</menu>
```

When Android calls onCreateOptionsMenu, Android *inflates* the res/menu/activity_main.xml file. *Inflating* the file means that Android turns the XML code into runnable instructions.[1]

MainActivity.java

```
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        contentView = findViewById(R.id.linearLayout1);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menu_start:
            RunAnimations();
            return true;
        case R.id.menu_settings:
            showSettingsDialog();
            return true;
        default:
            return super.onOptionsItemSelected(item);
        }
    }
```

[1] Android never executes XML code directly. Instead, Android inflates the XML code and then runs the inflated code. Often, this happens automatically (that is, without having you write an explicit call to an inflate method). In this example with its options menu, you can to explicitly call an inflate method.

Returning `true` from the onCreateOptionsMenu method means that the onCreateOptionsMenu method has done everything that you want done about the creating of the options menu, and that no additional statements need to be executed in order to handle the creation of the options menu.

But what happens when the user clicks one of the items in the menu? Notice that, in addition to the onCreateMenuOptions method, I add a brand new onOptionsItemSelected method. The switch statement inside this method tells Android "if the item that's been selected is the menu_start item, then do one thing, and if the item that's been selected is the menu_settings item, then do another thing."

When the user clicks the menu button (the hardware button on an older device, or the virtual button on a newer device's screen) then the menu appears and displays the items described in the res/menu/activity_main.xml file.