

# MODULE 7

# ACTION BAR

# TABLE OF CONTENT

- Action Bar
- Adding Action Item
- Action Item Event
- Up Button
- Webography

# Action Bar

## ▶ Action bar functions

- To identify the user location

- Name of the app/activity

If no name is given to the activity in AndroidManifest.xml (through android:label attribute)

↳ the app name is displayed

- To provide user actions and navigation modes through

- Action button

- Appears directly in the action bar with an icon and/or text

- Action overflow (hidden actions)

- Contains actions that can't fit in the action bar or aren't important enough

# Action Bar

- ▶ Automatically adapted for different screen configurations
- ▶ Style
  - According to the application style defined in AndroidManifest.xml
    - For **<application>** or **<activity>**
    - Through **android:theme** attribute
    - *N.B: Define styles in **res/values/styles.xml** !*
    - E.g,
      - ***Theme.AppCompat.Light.DarkActionBar***

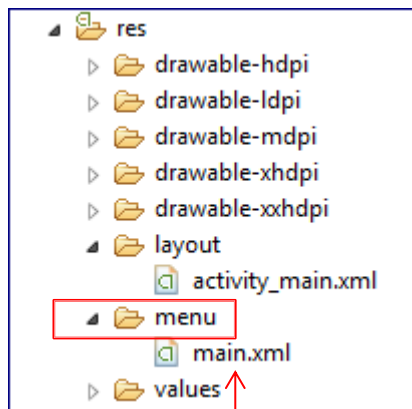
# Action Bar

► E.g,



# Action Bar

- ▶ Android Studio
  - Action bar included in Blank Activity
- ▶ Menu in xml file
  - In **res/menu** directory

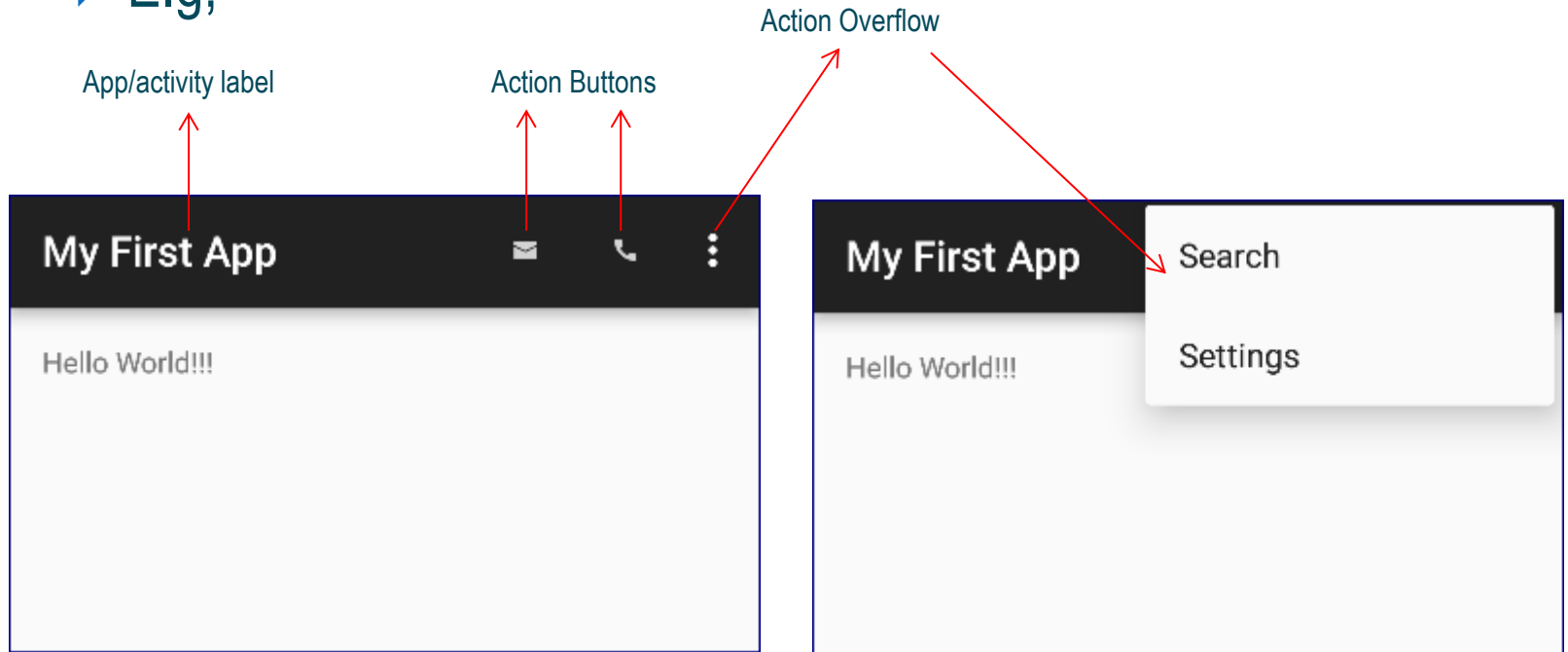


```
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    @Override  
    public boolean onOptionsItemSelected(Menu menu) {  
        getMenuInflater().inflate(R.menu.main, menu);  
        return true;  
    }  
}
```

Displays action bar

# Action Bar

► E.g,



# Adding Action Item

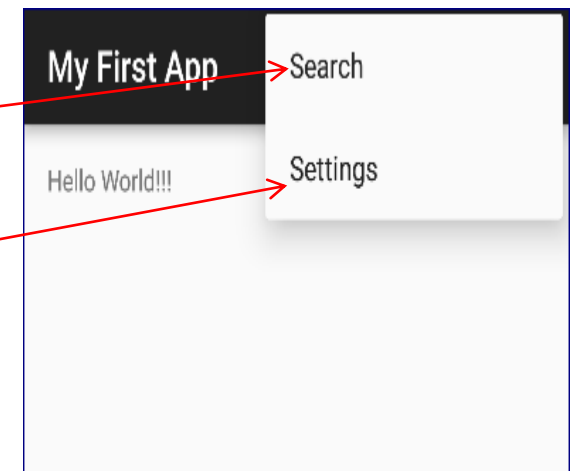
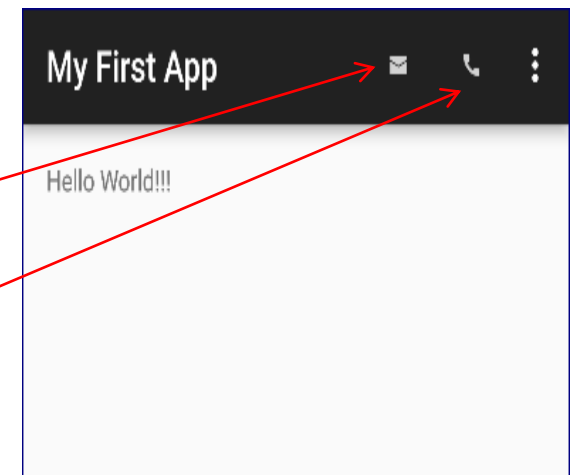
- ▶ Define action items in XML menu resource
  - **<item>** element for each item to include in the action bar
    - **title** attribute
      - Label of the action item
    - **icon** attribute
      - Place icons files in appropriate res/drawable-?? directory
      - Use icons provided in the Action Bar Icon Pack  
*<http://developer.android.com/design/downloads/index.html>*
    - **app:showAsAction** attribute
      - never (by default): action item hidden in the action overflow
      - always: item shown as action button in the action bar
      - ifRoom: item appears as an action button when room is available in the action bar



# Adding Action Item

► E.g,

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      tools:context="com.henallux.myfirstapp.ChildActivity">
    <item
        android:id="@+id/itemEmailID"
        android:icon="@drawable/ic_action_email"
        android:title="@string/emailTitleID"
        app:showAsAction="always" />
    <item
        android:id="@+id/itemCallID"
        android:icon="@drawable/ic_action_call"
        android:title="@string/CallTitleID"
        app:showAsAction="always" />
    <item
        android:id="@+id/itemSearchID"
        android:icon="@drawable/ic_action_search"
        android:title="@string/searchTitleID"
        app:showAsAction="ifRoom" />
    <item android:id="@+id/action_settings"
        android:title="@string/action_settings"
        android:orderInCategory="100"
        app:showAsAction="never" />
</menu>
```



# Action Item Event

- ▶ If press on action item
  - **onOptionsItemSelected()** callback method is called
    - Use **getItemId()** on the given MenuItem to determine which item is pressed

# Action Item Event

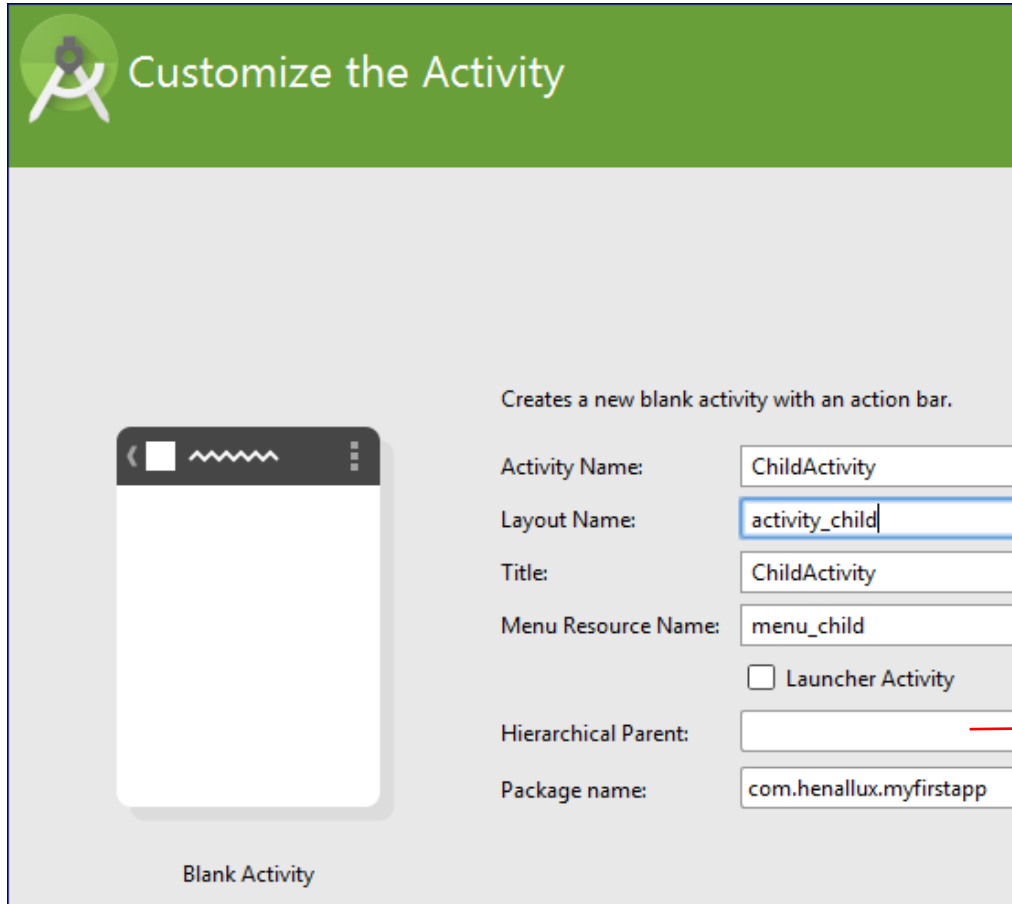
► E.g,

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId())
    {
        case R.id.itemEmailID:
            ... —————> To do when press on Email action item
            return true;
        case R.id.itemCallID:
            ... —————> To do when press on Phone action item
            return true;
        case R.id.itemSearchID:
            ... —————> To do when press on Search action item
            return true;
        case R.id.itemSettingsID:
            ... —————> To do when press on Settings action item
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

# Up Button

- ▶ To navigate to the logical parent screen in the app's hierarchy
- ▶ With Android 4.1 (API level 16) and above
  - Declare the parent activity in the AndroidManifest.xml
    - Through ***android:parentActivityName***

# Up Button



Customize the Activity

Creates a new blank activity with an action bar.

Blank Activity

Activity Name: ChildActivity

Layout Name: activity\_child

Title: ChildActivity

Menu Resource Name: menu\_child

☐ Launcher Activity

Hierarchical Parent:

Package name: com.henallux.myfirstapp

Activity to go back when Up Button is pressed

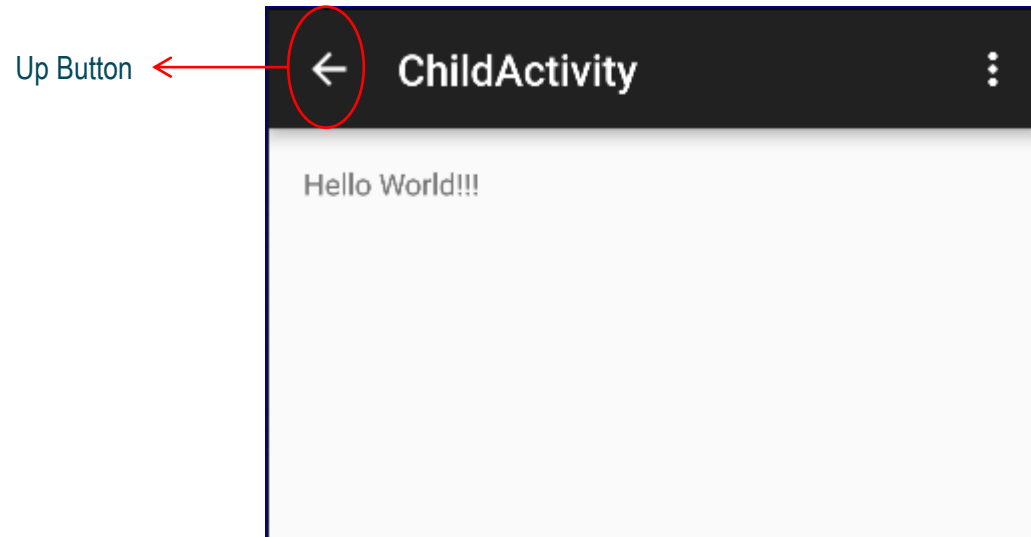
# Up Button

- ▶ E.g,
  - In AndroidManifest.xml

Parent of the child activity ←

```
<activity
    android:name="com.henallux.applicationbarapp.ChildActivity"
    android:label="@string/title_activity_child"
    android:parentActivityName="com.henallux.applicationbarapp.MainActivity" >
</activity>
```

# Up Button



# Webography

- ▶ <http://developer.android.com/training/basics/actionbar/index.html>
- ▶ <http://developer.android.com/guide/topics/ui/actionbar.html>