Android UI - Menus

Sang Shin
Michèle Garoche
www.javapassion.com
“Learn with Passion!”

Disclaimer

- Portions of this presentation are modifications based on work created and shared by the Android Open Source Project
  - http://code.google.com/policies.html
- They are used according to terms described in the Creative Commons 2.5 Attribution License
  - http://creativecommons.org/licenses/by/2.5/

Topics

- Types of menus
- Options menu
- Context menu
- Popup menu
- Creating menu using Menu resource

Types of Menus

Types of Menus

- Options menu and action bar
  - Since Android 3.0, the menu button is dropped
  - Consists of Navigation, Icons, & Menu
- Context menu and contextual action mode
  - Revealed on long-click on an element
- Popup menu
  - A modal menu anchored to a View

Options Menu and Action Bar

Action Bar Location

1. App Icon
2. Two Action Items
3. Action Overflow

How Options Menu Work?

- The Options Menu is opened by pressing the device MENU key for below Android 3.0
- Action Bar is space above the app display for App Icon, App Title, and user defined menu.
- Icons fit unto the Action Bar are displayed. The remaining are overflowed to a dropdown menu.

Populating Action Bar Menu: #1

/* AndroidManifest.xml

```
<application
        android:allowBackup="true"
        android:icon="@drawable/duke5"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.jpassion.menu_action_items.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

/* strings.xml

```
        <resources>
            <string name="app_name">Hello World</string>
        </resources>"
```

Populating Action Bar Menu: #2

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

    <item
        android:id="@+id/action_menu1"
        android:showAsAction="ifRoom|withText"
        android:title="@string/action_menu1"/>
    <item
        android:id="@+id/action_menu2"
        android:showAsAction="ifRoom|withText"
        android:title="@string/action_menu2"/>
    <item
        android:id="@+id/action_menu3"
        android:showAsAction="ifRoom|withText"
        android:title="@string/action_menu3"/>
    <item
        android:id="@+id/action_menu4"
        android:showAsAction="ifRoom|withText"
        android:title="@string/action_menu4"/>
    <item
        android:id="@+id/action_menu5"
        android:showAsAction="ifRoom|withText"
        android:title="@string/action_menu5"/>

</menu>
```

How to handle Menu Selection?

- When a menu item is selected from the Options Menu, *onOptionsItemSelected()* callback method of your Activity gets called
  - This callback passes you the *MenuItem* that has been selected.
  - You can identify the item by requesting the itemId, with *getItemId()*, which returns the integer that was assigned with the add() method.
  - Once you identify the menu item, you can take an appropriate action.

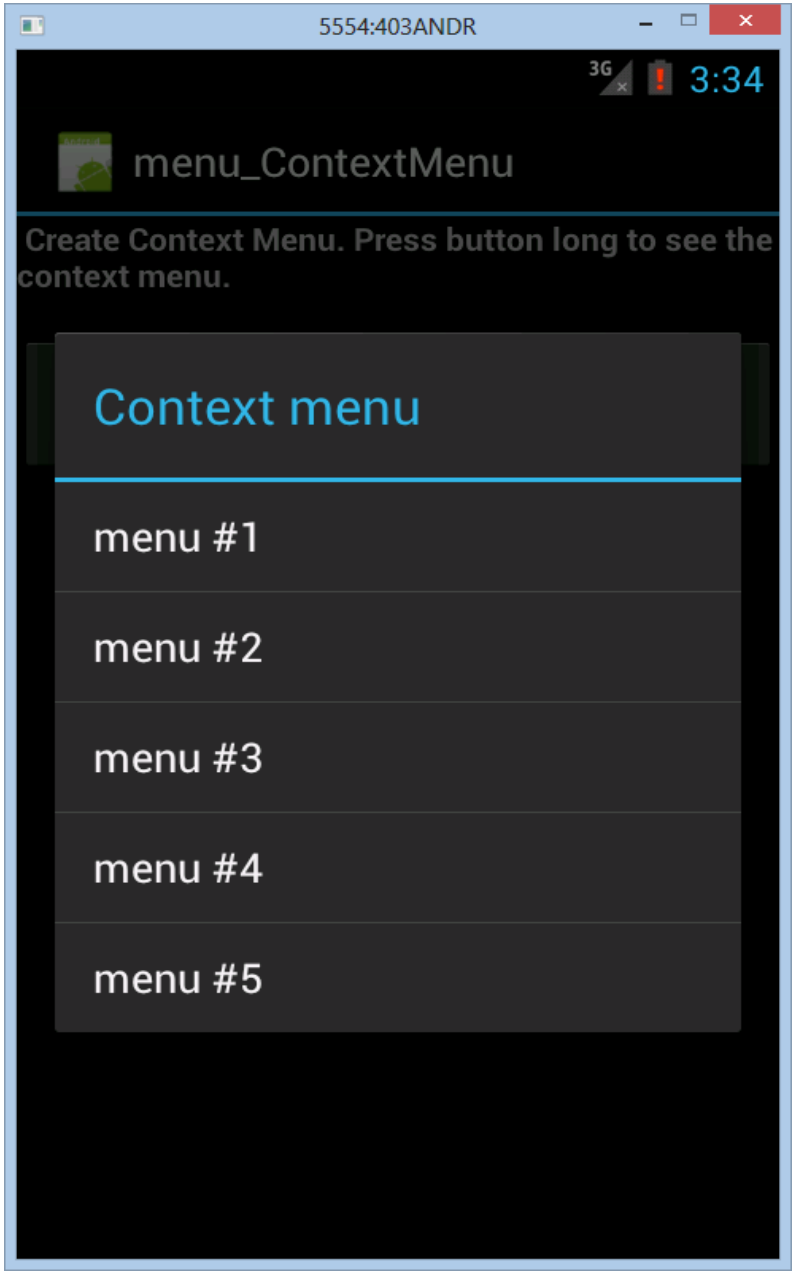
Example: Handling Menu Selection

```
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle presses on the action bar items
   switch (item.getItemId()) {
      case R.id.action_menu1:
               Toast.makeText(MainActivity.this,
                              "You selected menu item #1",
                              Toast.LENGTH_LONG).show();
         return true;
      case R.id.action_menu2:
               Toast.makeText(MainActivity.this,
                              "You selected menu item #2",
                              Toast.LENGTH_LONG).show();
         return true;
      case R.id.action_menu3:
               Toast.makeText(MainActivity.this,
                              "You selected menu item #3",
                              Toast.LENGTH_LONG).show();
         return true;
```

Context Menu

Context Menu

- Context menus do not support item shortcuts and item icons.



How to Create Context Menu?

- When Context menu is opened for the first time, the Android system will call the Activity's *onCreateContextMenu(Menu menu)* callback method.
  - You, as a context menu developer, override this method in your Activity class and populate the Menu object given to you with *MenuItem's*.
- You can populate the menu in two ways
  - Scheme #1: by calling add() for each item you'd like in the menu.
  - Scheme #2: by inflating a menu resource that was defined in XML (preferred)

Populating Menu with Menu Items: #1

```
// Override this method of Activity class in order to create menu items.
 @Override
 public void onCreateContextMenu(
        ContextMenu menu, // Context menu that is being built
         View view, // The view for which the context menu is being built
         ContextMenuInfo menuInfo) {

    super.onCreateContextMenu(menu, view, menuInfo);
    menu.setHeaderTitle("Context menu");
    menu.add(0, Menu.FIRST      , Menu.NONE, "menu #1");
    menu.add(0, Menu.FIRST + 1, Menu.NONE, "menu #2");
    menu.add(0, Menu.FIRST + 2, Menu.NONE, "menu #3");
    menu.add(0, Menu.FIRST + 3, Menu.NONE, "menu #4");
 }
```

How to handle User's Menu Selection?

- When a menu item is selected by a user from the Context Menu, *onContextItemSelected()* callback method of your Activity gets called
  - This callback passes you the *MenuItem* that has been selected.
  - You can identify the item by requesting the itemId, with *getItemId()*, which returns the integer that was assigned with the add(int groupId, int itemId, int order, CharSequence title) method.
  - Once you identify the menu item, you can take an appropriate action.

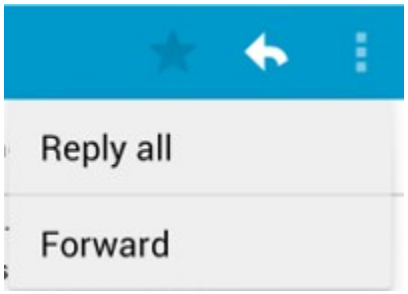
Example: Handling Menu Selection

```
/* Handles item selections */
public boolean onContextItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case MENU_NEW_GAME:
        newGame();
        return true;
    case MENU_QUIT:
        quit();
        return true;
    }
    return false;
}
```

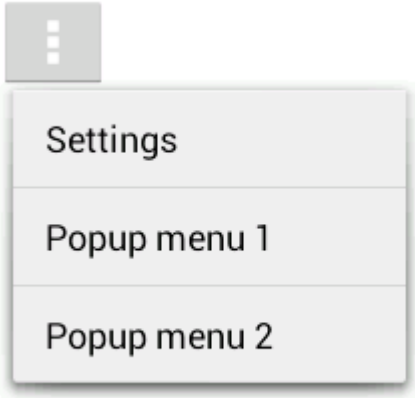
Popup

When to use Popup Menu?

Popup overflow-style
On Action Bar



Popup drop-down Spinner
On Button



Example: Creating Popup

```
<ImageButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/ic_overflow_holo_dark"
    android:contentDescription="@string/descr_overflow_button"
    android:onClick="showPopup" />

public void showPopup(View v) {
    PopupMenu popup = new PopupMenu(this, v);
    MenuInflater inflater = popup.getMenuInflater();
    inflater.inflate(R.menu.actions, popup.getMenu());
    popup.show();
}
```

Creating Menu using Menu Resource

What is & Why using Menu Resource?

- Define a menu and all its items in an XML menu resource, then inflate the menu resource (load it as a programmable object) in your application code.
- Defining your menus in XML is a better practice (than instantiating in code) because it separates your interface design from your application code (the same as when you define your Activity layout in XML).

How to Create/Use Menu Resource File?

- Create <menu_resource>.xml under *res/menu/* directory
- Inflate the Menu Resource file using *inflate(<menu-resource-id>)* method of the *MenuInflator* class
  - Menu objects are created from the Menu resource file

Example: Menu Resource File

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

    <item android:id="@+id/jump"
        android:title="Jump!"
        android:icon="@drawable/draw_jump" />

    <item android:id="@+id/dive"
        android:title="Dive!"
        android:icon="@drawable/draw_dive" />

</menu>
```

Example: Inflating Menu Resource

```
public boolean onCreateOptionsMenu(Menu menu) {

    // Inflate the menu XML resource.
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.title_only, menu);

    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {

    switch (item.getItemId()) {
    case R.id.jump:
        Toast.makeText(this, "Jump up in the air!", Toast.LENGTH_LONG)
                .show();
        return true;

    case R.id.dive:
        Toast.makeText(this, "Dive into the water!", Toast.LENGTH_LONG)
                .show();
        return true;


    }
```

Thank you!

Check JavaPassion.com Codecamps!
http://www.javapassion.com/codecamps
"Learn with Passion!"