Menus

- Menus are a common user interface component in many types of applications.
- For all menu types, Android provides a standard XML format to define menu items.
- Instead of building a menu in your activity's code, you should define a menu and all its items in an XML menu resource.
- You can then inflate the menu resource (load it as a Menu object) in your activity or fragment.

Why use Menu?

- It's easier to visualize the menu structure in XML.
- It separates the content for the menu from your application‘s behavioral code.
- It allows you to create alternative menu configurations for different platform versions, screen sizes, and other configurations by leveraging the app resources framework.

Menu structure

- To define the menu, create an XML file inside your project's res/menu/ directory and build the menu with the following elements:
- <menu>

  Defines a Menu, which is a container for menu items. A <menu> element must be the root node for the file and can hold one or more <item> and <group> elements.
- <item>

  Creates a MenuItem, which represents a single item in a menu. This element may contain a nested <menu> element in order to create a submenu.
- <group>

  An optional, invisible container for <item> elements. It allows you to categorize menu items so they share properties such as active state and visibility.

Menu

- The <item> element supports several attributes you can use to define an item's appearance and behavior.

- android:id : A resource ID that's unique to the item, which allows the application to recognize the item when the user selects it.

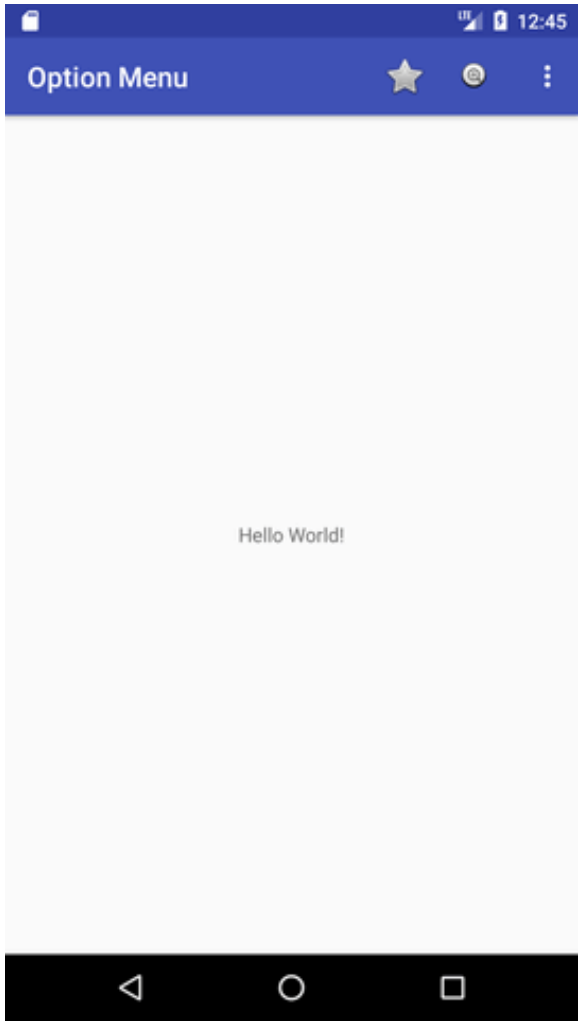
- android:icon : A reference to a drawable to use as the item's icon.

- android:title : A reference to a string to use as the item's title.

- app:showAsAction : Specifies when and how this item should appear as an action item in the app bar.

Options Menus

- Android Option Menus are the primary menus of android. They can be used for settings, search, delete item etc.
- We inflate the menu by calling the inflate() method of MenuInflater class. To perform event handling on menu items, you need to override onOptionsItemSelected() method of Activity class.
- Icons can be set with the menu items. You need to have icon images inside the res/drawable directory. The android:icon element is used to display the icon on the option menu.



Options Menu

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
<item android:id="@+id/newBookmark"
        android:title="@string/menu_add"/>
<item android:id="@+id/appInfo"
        android:title="@string/menu_info" />
</menu>
```

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

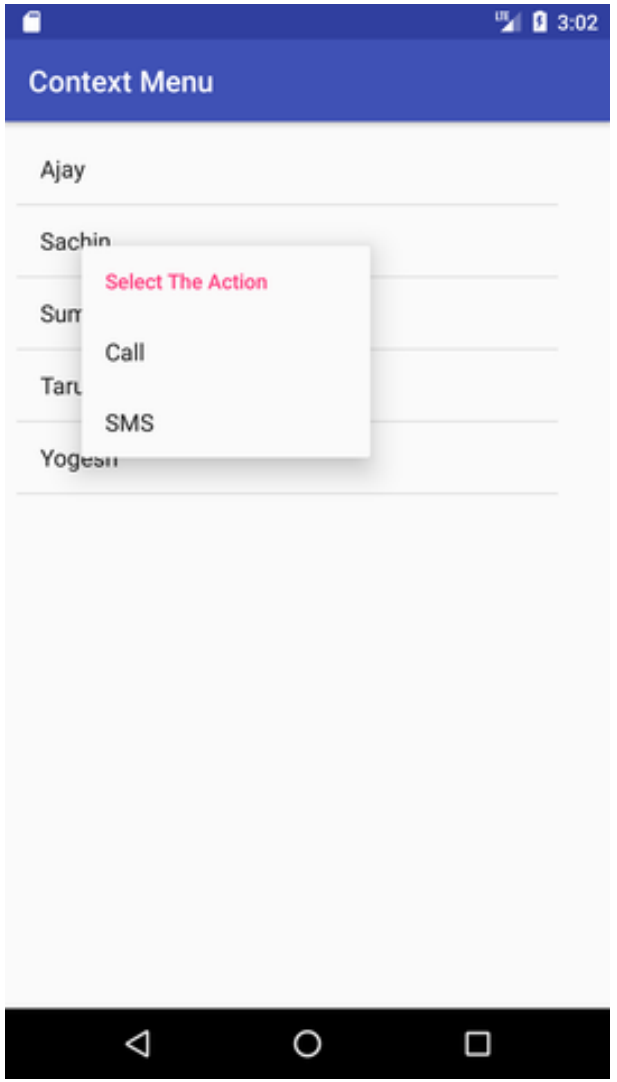
Options Menu

- You can identify the item by calling getItemId(), which returns the unique ID for the menu item (defined by the android:id attribute in the menu resource)
- You can match this ID against known menu items to perform the appropriate action.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.appInfo:
            openAppInfoActivity();
            return true;
        case R.id.newBookmark:
            openAddBookmarkActivity();
            return true;
        default:
    return super.onOptionsItemSelected(item);
    }
}
```

Contextual Menus

- A contextual menu offers actions that affect a specific item or context frame in the UI.
- You can provide a context menu for any view, but they are most often used for items in a ListView and GridView, or other view collections in which the user can perform direct actions on each item.



Creating a floating context menu

- Register the View to which the context menu should be associated by calling registerForContextMenu() and pass it the View. Implement the
- Implement the onCreateContextMenu() method in your Activity or Fragment.

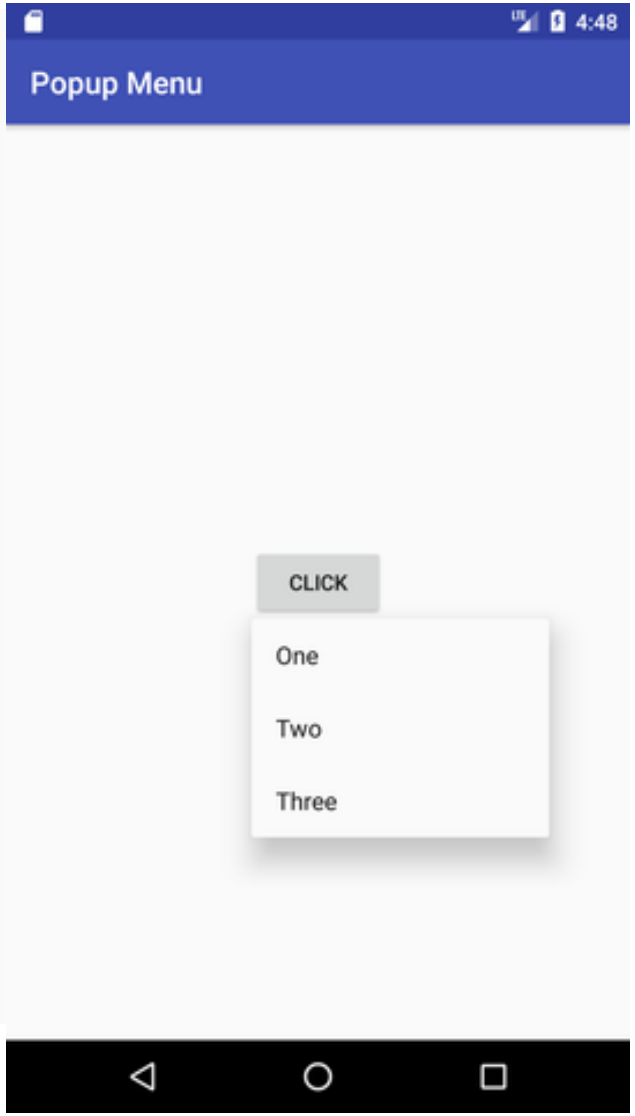
```
Button btn = (Button) findViewById(R.id.button1);
registerForContextMenu(btn);
```

```
@Override
   public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {
       super.onCreateContextMenu(menu, v, menuInfo);
       MenuInflater inflater = getMenuInflater();
       menu.setHeaderTitle("Select The Action");
       inflater.inflate(R.menu.menu_context, menu);
   }
```

Popup Menus

- Popup Menu displays a list of items in a modal popup window that is anchored to the View. The popup menu will appear below the view if there is a room or above the view in case if there is no space and it will be closed automatically when we touch outside of the popup.

- The android Popup Menu provides an overflow style menu for actions that are related to specific content.



Popup Menus

```
public void onClick(View v) {
    PopupMenu p = new PopupMenu(PopupMenuActivity.this, v);
    p.getMenuInflater().inflate(R.menu.menu_context, p.getMenu());
    p.show();

    //registering popup with OnMenuItemClickListener
   p.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener()
   {
         public boolean onMenuItemClick(MenuItem item) {
          Toast.makeText(MainActivity.this,"You Clicked : " + item.getTitle(),
                Toast.LENGTH_SHORT).show();
          return true;
         }
  });
```