

A seagull is shown in flight, wings spread wide, against a dramatic sky with orange and blue clouds. The bird is white with dark wingtips and a yellow beak. The sky transitions from a deep blue at the top to a bright orange near the horizon, with scattered white clouds.

**globsyn**  
*Taking People To The Next Level*

# globalsyn



**globsyn**   
**finishing school**

# Menu

# Menu



- Menus are the basic UI components used in almost all the applications. Menus help the user interact with the application.
- Android has a built-in API called as the Menu API from which you can derive many methods.
- The new versions of Android have replaced the default menu with a new option called as the Action Bar.

# Types of Menus

There are three types of Menus in Android

- ♣ Options Menu – This is displayed when you press the hardware menu button or by touching the Menu Button.
- ♣ Context Menu – These are the menu items that appear if you long press an element. They provide changes to that particular element alone. They are also called as Floating Menus.
- ♣ Popup Menu – It displays the options in a vertical list. It does not affect the content but extends the actions to that content.

# How to Define a Menu in Android?

- Usually, a menu is defined with the help of three containers.
- `<menu>` - which is the root element that contains the following elements. It is compulsory.
- `<item>` - This defines a single item which is going to be present inside a menu. So, if your menu is going to contain six options, you need six `<item>` containers.
- `<group>` - This is optional container. It can be used to group the items into a collection so that you could implement and modify them together.

## The <item> container

- 1 The <item> container used in Menus is one of the most important containers in Android. It defines each Menu item individually.
- 2 It contains the following attributes
- 3 android:id – This gives a unique ID to the <item>
- 4 android:icon – Refers to the drawable icon of the item.
- 5 android:title – Refers to a string, the title of the <item>
- 6 android:showasAction – Specifies when this particular item should appear as an action.

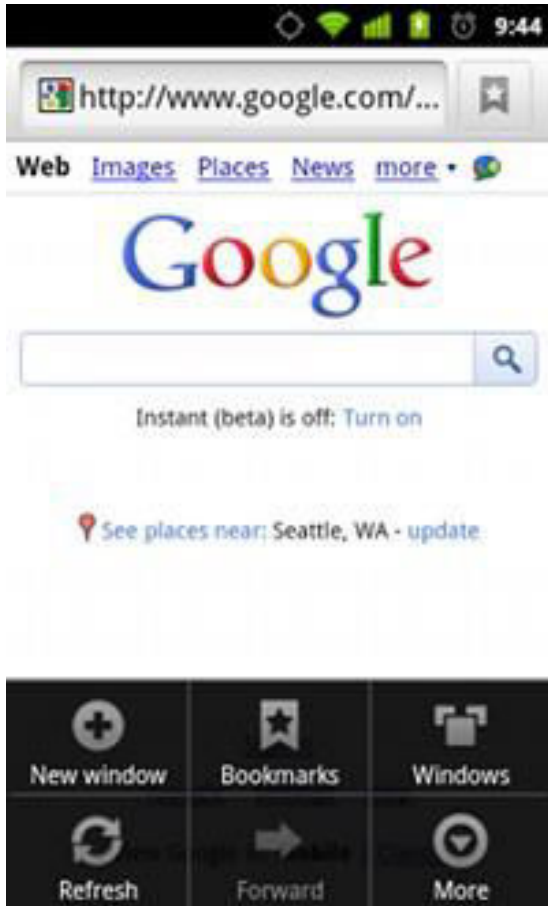


# Creating Option Menu

- 1 The Option Menu is the most common Menu that is used in Android.
- 2 It appears during actions such as clicking the search bar, clicking on compose in an e-mail window or during the selection of settings.
- 3 To create, Options Menu, use following syntax.

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

# Option Menu Example



An example for Options Menu would look like the below.

As soon as the Google Search Bar is clicked, the menu will appear.

By default, it can contain 6 items. If any more items are present, it would be placed into an Overflow Menu called as More

# What are Contextual Menus?

┌ A menu that provides options for a certain element or a context are known as Contextual Menus.

┌ Contextual Menus can be applied using two ways

- ♣ The first way is using a floating menu in which a floating menu will appear with corresponding options when the user long presses an element.
- ♣ The second method is more efficient. It is given as an option on a bar on top with needed actions. The user selects the element he wants and clicks that option on the bar and necessary options will be displayed.

# Creating Contextual Menus

A Normal Floating Contextual Menu can be created just like the Options Menu.

@Override

```
public void onCreateContextMenu(ContextMenu menu, View v,  
    ContextMenuInfo menuInfo) {  
    super.onCreateContextMenu(menu, v, menuInfo);  
    MenuInflater inflater = getMenuInflater();  
    inflater.inflate(R.menu.context_menu, menu);  
}
```

Creating Contextual Menus on action bars is a bit difficult. You need `callback()` as well as `startActionMode()` interfaces to do that.

# Contextual Menus Example

Are you wondering how the two types of Contextual Menus look like ? Take a look.



The first image is that of the floating menu and the next represents a selection in the action bar.

# Popup Menus

- 1 } Popup Menus are those that appear below a View when the View is clicked.
- 2 } If there is no space below the View, the popup menu appears above it. Anyhow, there would be contact between the View and the popup menu at all times.
- 3 } They can be defined as Overflow style menu items that appear when an option is clicked.

## ■ How are Popup Menus Different from Contextual Menus?

---

- When you look at it, both Popup menus and Contextual Menus on action bars look like the same.
- The major difference is the fact that you apply the Contextual Menu to a particular selection or text. But, the popup menus provide menus for actions that relate to a specific context.
- For instance, if you click reply to an email, a popup of Forward or Reply all will appear.

## Creating Popup Menus

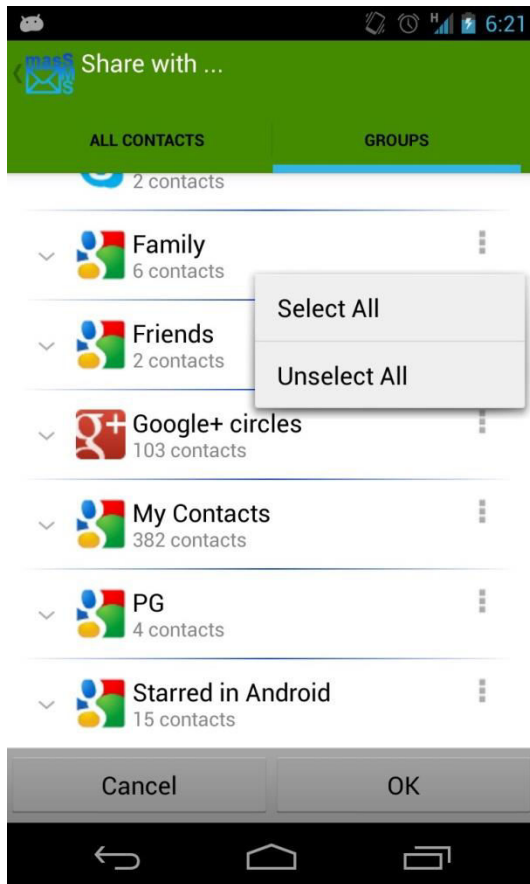
Creation of Popup Menus is an easy task. You should initialize the PopupMenu, then Inflate it and finally display it using `popup.show()`

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

Thus, you have created a Popup Menu.



# Popup Menu Example



Here is a very good example for Popup Menu

Once many options are visible, you could get a popup menu, which gives you “select all” or “unselect all” options. This does not refer to any particular element but relates to the context.

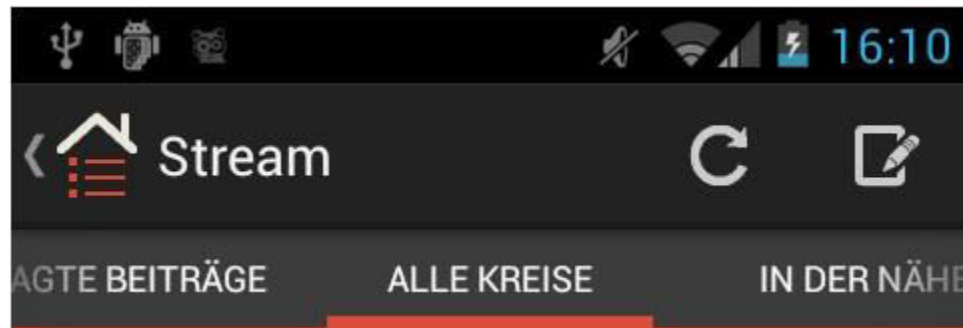
# Action Bars

- So, do you want to know what an Action Bar is?
- You have probably seen it a hundred times but have never really noticed it.
- It is something that is present at the top of almost all Activity Screens. You can define various operations in these Action Bars to navigate through your application.
- Older Android devices used an option called as Menus but after Android 3.0 upgrade, Menu has been replaced with Action Bars

# Why should You Use an Action Bar?

Why not keep up with the existing Menu bars? What is the actual need for these Action Bars? Take a look.

- ♣ Action Bars provide a dedicated space where the options can be placed.
- ♣ It provides a consistent pattern of navigation across all windows, panes and interfaces.
- ♣ You can give prominence and a special icon to key options such as search.



# How to Add Action Bars?

Adding action bars is easy. Before you begin, make sure you are running on Android Versions higher than Android 3.0 because the Older Versions do not support Action Bars.

```
<manifest ... >  
    <uses-sdk android:minSdkVersion="4"  
        android:targetSdkVersion="11" />  
    ...  
</manifest>
```

By adding the above code, you could create an Action Bar.

## How do You Delete/Hide an Action Bar

Deleting or Hiding an action bar too isn't a tough job. All you need to do is to mention the activity as NoActionBar along with the activity theme.

```
<activity android:theme="@android:style/Theme.Holo.NoActionBar">
```

If you want to hide the action bar, you have to call the hide() function.

```
ActionBar actionBar=getActionBar();  
actionBar.hide();
```

## Adding New Items into the Action Bar

- Usually, not all the options are displayed on the Action Bar itself. Only the important actions that the user feels are displayed on the Action Bar and the rest of them are stacked inside an overflow stack. Items are added just like they are in Menus.
- Once the program begins, the `onCreateOptionsMenu()` is called.

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

## Adding New Items into the Action Bar Contd.

- Now, how do you declare which items appear on the Action Bar and which appear inside a overflow stack? That is done using the `ifRoom` declaration of the `android:showAsAction` attribute inside the `<item>`.
- So, when a particular item with the `android:showAsAction="ifRoom"` value arrives, the system checks if the Action Bar has space. If it does, then that element is placed in the Action Bar.
- If no mention of the `showAsAction` attribute is given, then the element is straightly moved to the Stack.

## Adding New Items into the Action Bar Contd.

- Usually, only the icon (android:icon) of the item is displayed in the Action Bar but if you want to make the android:title, you can do it.
- If you use the following declaration,

```
android:showAsAction="ifRoom|withText"
```

It means that the text string in android:title will also get mentioned along with the icon of the element.



## How to Choose Action Items?

- 1 So, how do you choose which items to appear in the Action Bar and which to be delegated to the stack?
- 2 You have to watch a few traits of the items to analyze that.
- 3 For instance, if an item is frequently visited, then it should be placed in the Action Bar. Similarly, if you feel that it is very important to your app, you can place it in the Action Bar.
- 4 But, if you feel that the item is a typical one, you can move that to the stack.

## How to Split the Action Bars?

- | In upgraded Android versions higher than 4.0, there is an way using which you can split the action bars.
- | Once you do it, a new action bar will appear at the bottom of the touchscreen giving you two actions bars in total. That is, all your items can now appear without overflowing into a stack.
- | To do this, you need to add the following code to your <application> or <activity> container.

```
uiOptions="splitActionBarWhenNarrow"
```

# Navigating Using the Action Bar

- | The Action Bar will obviously have the icon of your Application in its left corner.
- | You can use this Icon for two purposes,
  - ♣ When clicked, you could load the Home Page through that navigation using `onOptionsItemSelected()` method.
  - ♣ Else, you could use the click to move the page to the top of the application using `Flag_Activity_Clear_Top` intent.

## References

Android Developers :

<http://developer.android.com/guide/topics/ui/actionbar.html>


Lars Vogel, Vogella:

[www.vogella.com/articles/AndroidActionBar/article.html](http://www.vogella.com/articles/AndroidActionBar/article.html)

Android Hive : [www.androidhive.info/2011/09/how-to-create-android-menus/](http://www.androidhive.info/2011/09/how-to-create-android-menus/)

Android Developers:

[developer.android.com/guide/topics/ui/ui-events.html](http://developer.android.com/guide/topics/ui/ui-events.html)



*Taking People To The Next Level...*