

Android for .NET Developers Series

Adopting the Android Mindset

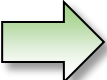
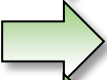



Moving from menu thinking to action thinking

Jim Wilson
hedgehogjim.wordpress.com
@hedgehogjim
jimw@jwhh.com



pluralsight 
hardcore developer training

Outline

-  Moving from menu to ActionBar
-  Activity and Fragment ActionBar cooperation
-  Linking to the app “Home” screen
-  Making more actions directly accessible
-  Learning more about the ActionBar

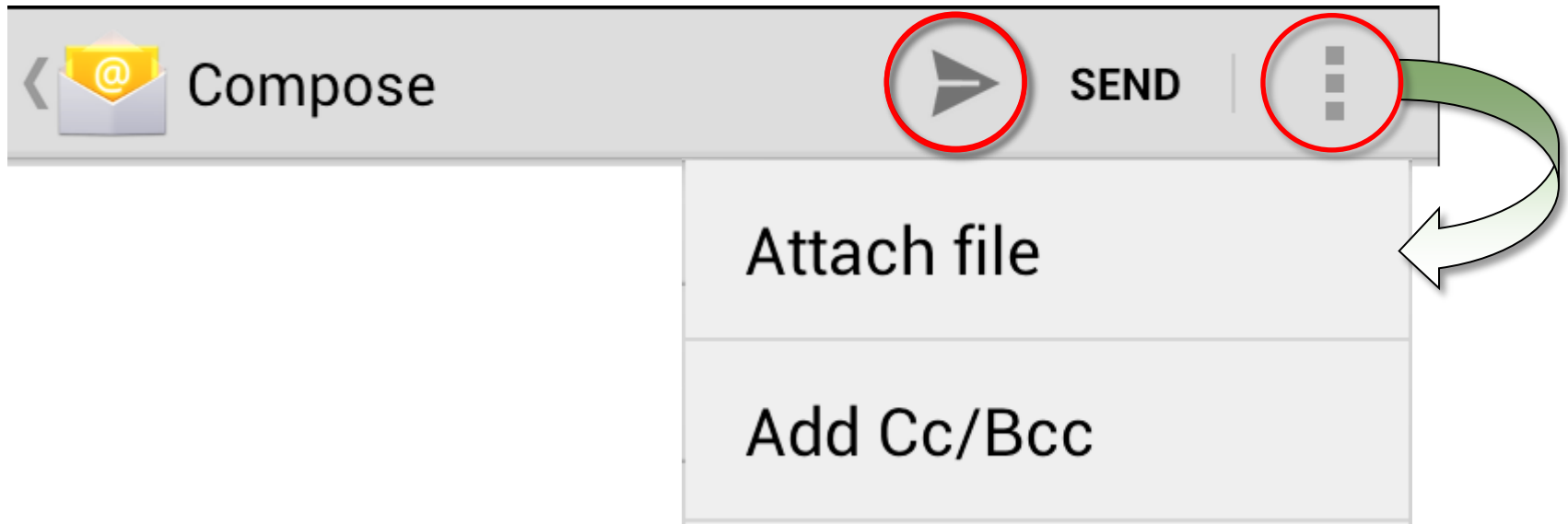


Adopting the android mindset

*Provide immediate access
to commonly used
features*

Minimize menu use on mobile

- ➔ **Make common actions directly accessible**
- ➔ Take advantage of the ActionBar
 - Primary point of interaction starting in API Level 11
- ➔ Put most frequent items directly on the ActionBar
 - Provides a “just push the button” experience
- ➔ Move less frequently used items into menus
 - Avoid deeply nesting menus



ActionBar

- ➡ **ActionBar provides a highly adaptable solution to user actions**
- ➡ Basic behavior handled as a menu
 - ❑ Populated from menu resource through onCreateOptionsMenu
 - ❑ Selections handled with onOptionsItemSelected
- ➡ Item element's showAsAction attribute provides hints on ActionBar behavior
 - ❑ always: Always show on ActionBar
 - ❑ ifRoom: Show on ActionBar if space available, otherwise in action overflow
 - ❑ withText: Include text with icon if space allows
 - ❑ Multiple values separated by vertical bar (|)

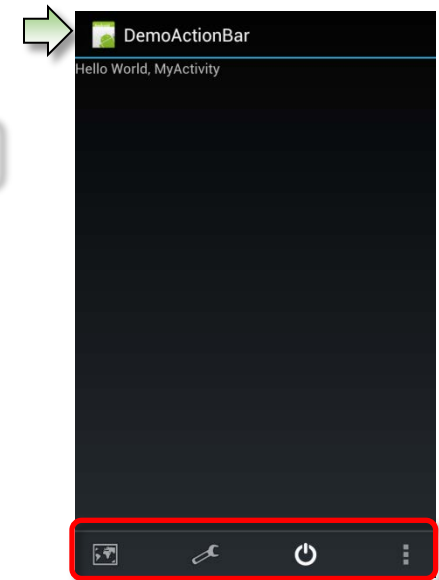
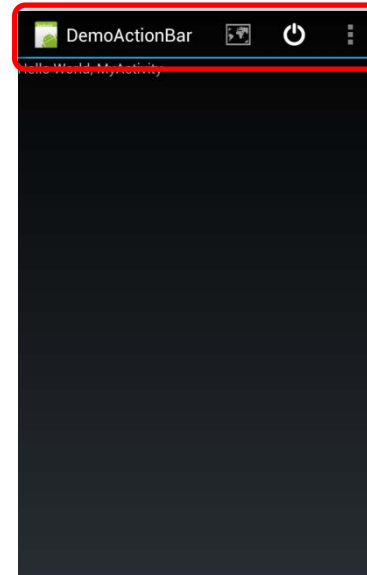
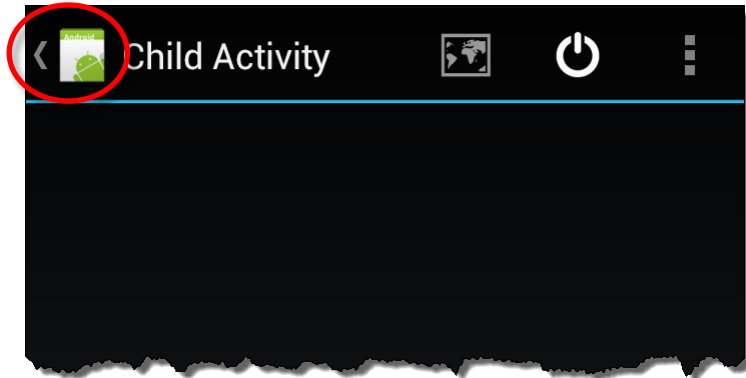
```
<menu xmlns:android="...">
  <item
    android:id="@+id/action_edit"
    android:title="@string/action_edit"
    android:icon="@drawable/ic_action_edit"
    android:showAsAction="ifRoom" />
</menu>
```

Activity and Fragment ActionBar cooperation

- ➡ **Fragment can handle and add ActionBar items**
 - ➡ By default only Activity is involved in ActionBar items
 - ➡ Call Fragment's `setHasOptionsMenu` to add Fragment participation
 - ➡ Allows Fragment to handle ActionBar items
 - ❑ Fragment's `onOptionsItemSelected` called before Activity's
 - ❑ If Fragment returns false, then Activity's is called
- ➡ **Allows Fragment to add ActionBar items**
 - ❑ Fragment's appear after Activity's
 - ❑ ActionBar automatically updates as Fragments are add & removed

Adding special behaviors

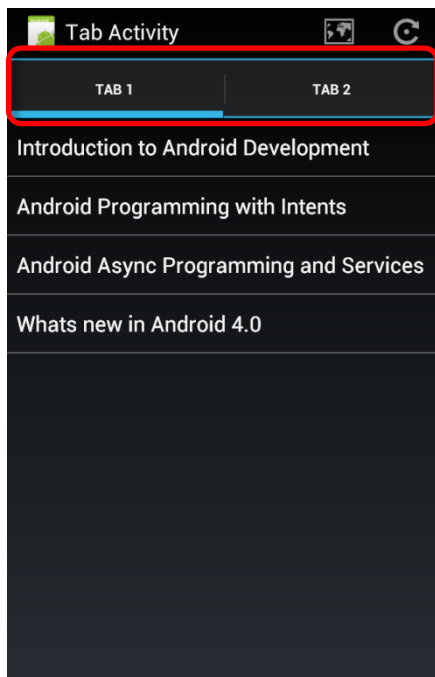
- ➔ **ActionBar capabilities go beyond simple menu-like behavior**
- ➔ ActionBar icon can serve as a “Up” or “Home” button
 - ❑ Known as “up affordance”
 - ❑ Provides a standard way to return a home screen or move up in hierarchy
- ➔ Split ActionBar into 2 parts
 - ❑ On narrow screens, icons and menu moved to a bar at bottom of screen
 - ❑ On wide screens, remains as a single bar



This is just the beginning

➔ **The ActionBar is loaded with capabilities**

- ➔ Tabbed display behavior
- ➔ Dropdown list navigation
- ➔ Auto-hide capabilities
- ➔ Much, much more



Summary

- ➡ **The ActionBar is at the heart of user interaction**
 - Highly adaptable to different application needs
 - Highly adaptable to device differences
- ➡ **ActionBar managed similar to menu**
 - Uses same resources and methods as menus
 - The showAsAction attribute adds ActionBar appearance hints
- ➡ **Fragments can participate in ActionBar**
 - Fragment must call setHasOptionsMenu passing true
 - Fragment handles action items before Activity
 - Fragment added action items appear after Activity added action items
- ➡ **Add an app “Home” feature with up affordance**
- ➡ **Make more icons visible on narrow displays by splitting the ActionBar**