FPT Software

ANDROID TRAINING
LESSON 3

Version 0.1

Agenda

- ImageButton
- ProgressBar
- SeekBar
- UI Events
- Option menu and Context Menu
- Inflate Xml for Menu

ImageButton

- Create a button with a custom image instead of text, using the Button widget and an XML file that defines three different images to use for the different button states.

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:drawable="@drawable/android_pressed"
        android:state_pressed="true" />
   <item android:drawable="@drawable/android_focused"
        android:state_focused="true" />
   <item android:drawable="@drawable/android_normal" />
</selector>
```

ProgressBar

- Progress bar is useful to tell user that the task is takes longer time to finish.
- The key to use progress bar is using “Thread” to run your time consume task and another “Thread” to update the progress bar status accordingly.

SeekBar

- A SeekBar is an extension of ProgressBar that adds a draggable thumb. The user can touch the thumb and drag left or right to set the current progress level or use the arrow keys.

UI Events

- **Define an event listener and register it with the View.**
  The View class contains a collection of nested interfaces named OnxxxListener, each with a callback method called On().
  For example:
  - **View.OnClickListener:** for handling "clicks" on a View.
  - **View.OnTouchListener:** for handling touch screen events in a View.
  - **View.OnKeyListener**: for handling device key presses within a View.
- So if we want our View to be notified when it is "clicked" (such as when a button is selected),
  - implement OnClickListener
  - define its onClick() callback method (where we perform the action upon click)
  - register it to the View with setOnClickListener().

UI Events

- **Override an existing callback method for the View.**
  This is what we should do when we've implemented our own View class and want to listen for specific events that occur within it.
  - Example events we can handle include:
    - **onTouchEvent()**
      Called when the touchscreen is pressed or released, or when it detects movement.
    - **onTrackballEvent()**
      Called when the device's trackball is moved.
    - **onKeyDown()**
      Called when any device key is pressed; includes the D-pad, keyboard, hang-up, call, back, and camera buttons.
    - **onKeyUp()**
      Called when a user releases a pressed key.

Option menu and Context Menu

- **Options menu and action bar**
  - The options menu is the primary collection of menu items for an activity.
    - Android 2.3 or lower, users can reveal the options menu panel by pressing the *Menu* button.
    - On Android 3.0 and higher, items from the options menu are presented by the action bar as a combination of on-screen action items and overflow options. Beginning with Android 3.0, the *Menu* button is deprecated (some devices don't have one).

Option menu and Context Menu

- **Context menu and contextual action mode**
  - A context menu is a floating menu that appears when the user performs a long-click on an element.
  - When developing for Android 3.0 and higher, you should instead use the contextual action mode to enable actions on selected content. This mode displays action items that affect the selected content in a bar at the top of the screen and allows the user to select multiple items.

Option menu and Context Menu

- A popup menu displays a list of items in a vertical list that's anchored to the view that invoked the menu.

Inflate Xml for Menu

- For all menu types, Android provides a standard XML format to define menu items.
- Using a menu resource is a good practice for a few reasons:
  - 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.

Inflate Xml for Menu

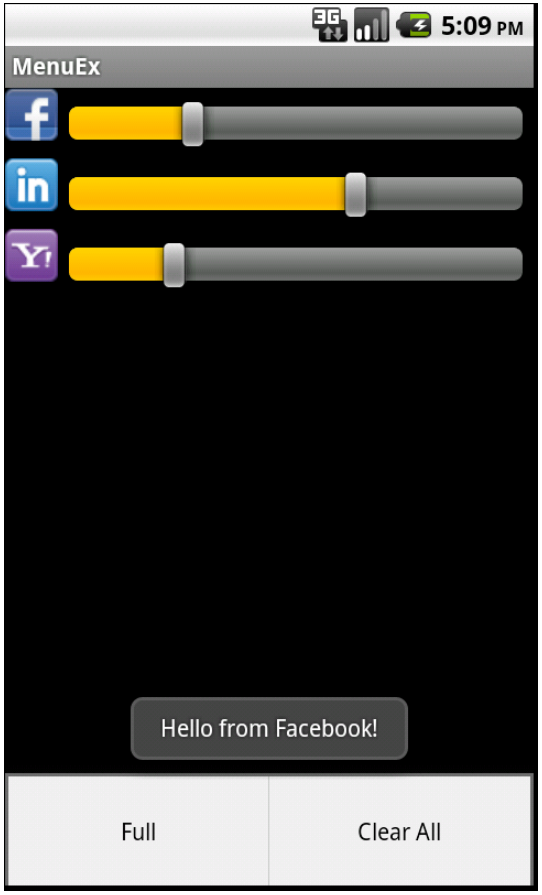
- To define the menu, create an XML file inside your project's res/menu/ directory and build the menu with the following elements:
  - <menu>
    - Defines a Menu, which is a container for menu items. A <menu> element must be the root node for the file and can hold one or more <item> and <group> elements.
  - <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.

Inflate Xml for Menu

```
<?xml version="1.0" encoding="utf-8"?>
    <menu xmlns:android="http://schemas.android.com/apk/res/android">
      <item android:id="@+id/new_game"
          android:icon="@drawable/ic_new_game"
          android:title="@string/new_game"
          android:showAsAction="ifRoom"/>
      <item android:id="@+id/help"
          android:icon="@drawable/ic_help"
          android:title="@string/help" />
    </menu>
```

Homework

- Develop an application:
  - With GUI the same with Picture
  - Option menu:
    - Full: all progress of seekbar reach 100%
    - Clear ALL: all progress of seekbar return 0
  - Click on image button: say hello from FB, or Linkin…



Thank you!