Accelerating Information Technology Innovation

http://aiti.mit.edu

India Summer 2012
Lecture 3 – Android User Interface (Navigation)
Overview

• Models of app navigation
  ○ The activity stack
  ○ ListView/GridView
  ○ Menus
  ○ Dialogs and Notifications

• Using navigation in your app
The Activity Stack

The activity stack is usually used to separate **modal** actions (i.e. different modes: “reading mail” → “writing reply”)

Wednesday, July 4, 12
How to use the Activity Stack

Generally: Navigate from lists/overview to details!
How to use the Activity Stack

Also useful for temporary states (e.g. replying to e-mail)
Using the Activity Stack

Navigation Examples

Another Activity
public class NavigationExamples extends Activity {
    public void onClickMe(View button) {
        /* Create an Intent to start AnotherActivity */
        Intent startAnotherActivity =
            new Intent(NavigationExamples.this, 
                        AnotherActivity.class);

        /* Start the activity. */
        startActivity(startAnotherActivity);
    }
}

The Code:
onClickMe()
Using the Activity Stack: Demo
Aside: The Manifest and Intent Filters
Aside: The Manifest and Intent Filters

This will appear as a separate launcher! →
Aside: The Manifest and Intent Filters

`android.intent.category.ALTERNATIVE` makes it an “alternative” activity (which won’t appear in the menu)
ListView and GridView
ListView and GridView

Useful to organize lists of items for further detail (e.g. lists of nearby restaurants, lists of train times...)
Adapters

Adapters “wrap” a data source (database, array, etc.) for use in a View (ListView, GridView, Spinner, etc.)
Using a ListView

Navigation Examples

Item 1
Sub Item 1

Item 2
Sub Item 2

Item 3
Sub Item 3

Item 4
Sub Item 4

Item 5
Sub Item 5

Item 6
Sub Item 6

Item 7
Sub Item 7

Navigation Examples
The Code:

`onCreate()`

```java
public class NavigationExamples extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        /* Initialize the Activity */
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_navigation_examples);

        /* Set up the ListView's adapter */
        ListView myListView =
            (ListView) findViewById(R.id.myListView);
        ArrayAdapter<String> stateList =
            new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1,
                getResources().getStringArray(R.array.states));
        myListView.setAdapter(stateList);

        /* ... */
    }
}
```
The Code:

onCreate()

/* ... */

/* Listen for clicks on each item in the ListView */
myListView.setOnItemClickListener(
    new OnItemClickListener() {
        public void onItemClick(AdapterView<?> parent,
                                View view, int position, long id) {
            /* Start AnotherActivity... */
            Intent anotherActivityIntent =
                new Intent(NavigationExamples.this,
                            AnotherActivity.class);
            startActivity(anotherActivityIntent);
        }
    });
Using a ListView:
Demo
Menus
Menus

Option Menu

Floating context menu and contextual action bar

Useful for common modes or actions related to an Activity

(Context menus are useful for items IN a View)
public class NavigationExamples extends Activity {
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_navigation_examples, menu);
        return true;
    }
}
Option Menus
The Code:

```java
public class NavigationExamples extends Activity {
    public boolean onOptionsItemSelected(MenuItem item) {
        /* Choose your action based on the item selected. */
        switch (item.getItemId()) {
            case R.id.menu_settings:
                /* Make the settings appear. */
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```
Context Menus

- Very similar to Option Menus
  - `onCreateContextMenu()` instead of `onCreateOptionsMenu()`
    - Include `super.onCreateContextMenu()` first
  - `onContextItemSelected()` instead of `onOptionsItemSelected()`
    - `menuItem.getItemId().id` gets the item context in `onOptionsItemSelected()`
Using Menus: Demo
I Taught You That So I Could Teach You This:
The Action Bar
The Action Bar

At the top of the screen, this is better than a menu
The Action Bar

At the top of the screen, this is better than a menu

Downside: It’s only supported on Android 3.0+ (API Level 11+)
The Action Bar

At the top of the screen, this is better than a menu

Downside: It’s only supported on Android 3.0+ (API Level 11+)

But there is some sample compatibility code in
<sdk>/samples/<android-version>/ActionBarCompat/
The Action Bar

Just like Option Menus!

• Uses `onCreateOptionsMenu()` and `onOptionsItemSelected()`

• Make Action Bar Items using menu resources
Activating the Action Bar

Min SDK or Target SDK > 11
Activating the Action Bar
The Action Bar: Demo
Dialogs and Notifications
Dialogs and Notifications

AlertDialogs ask for simple user input.
Other dialogs are used to show progress of an action.

Toast Notification
Status Notification

AlertDialogs ask for simple user input.
Other dialogs are used to show progress of an action.

Notifications let the user know of a background event (e.g. “2 new e-mails!” or “An update has been downloaded”)
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("You clicked the context menu!");
    .setCancelable(false)
    .setPositiveButton("Yes",
        new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                dialog.cancel();
            }
        })
    .setNegativeButton("No",
        new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                dialog.cancel();
            }
        });
AlertDialog dialog = builder.create();
dialog.show();
Toast Notifications: Code

Toast.makeText(this,
        "This is a toast notification!",
        Toast.LENGTH_SHORT)
    .show();
/* Get the NotificationManager service. */
String ns = Context.NOTIFICATION_SERVICE;
NotificationManager mNotificationManager =
    (NotificationManager) getSystemService(ns);

/* Create the notification. */
int icon = R.drawable.ic_launcher; // Icon
CharSequence tickerText = "The ticker text"; // Ticker Text
long when = System.currentTimeMillis(); // When to show

Notification notification = new Notification(
    icon, tickerText, when);
/* Prepare the event info (message) */

Context context = getApplicationContext();
CharSequence contentTitle = "My notification";  // The title
CharSequence contentText = "Hello World!";  // The text

/* This intent to restart is "delayed" for a later date. */
Intent notificationIntent = new Intent(
    this, NavigationExamples.class);
PendingIntent contentIntent = PendingIntent.getActivity(
    this, 0, notificationIntent, 0);

notification.setLatestEventInfo(
    context, contentTitle, contentText, contentIntent);

/* Send the notification. */

final int HELLO_ID = 1;
mNotificationManager.notify(HELLO_ID, notification);
Resources

- Android Developer Site

- Android Design (on the Developer Site)
More Resources

• Android Training (on the Developer Site)
  ○ “Designing Effective Navigation”
    <http://developer.android.com/training/design-navigation/index.html>
  ○ “Implementing Effective Navigation”

• Android Patterns (a list of useful interaction patterns for Android apps)
  <http://www.androidpatterns.com/>

• Giorgio Venturi has a presentation about Android Design Patterns
  <http://www.closertag.com/thought-pieces/android-design-patterns/>