Fragments

Programming the Android Platform
- Tablets have larger displays than phones do.
- Can therefore support multiple UI panes / user behaviors at the same time.
  - May not need “one activity per screenful of data” rule of thumb.
**Fragment**

- Represents a behavior / portion of UI within an Activity
- Multiple Fragments can be embedded in an Activity to create a multi-pane UI
- A single Fragment can be reused across multiple Activities
Email Viewer Example

- On phone – 2 Activities
  - List email messages
  - View selected email message
- On tablet – 2 Fragments embedded in 1 Activity
Fragments have their own lifecycles and receive their own events

But Fragment lifecycle interacts with containing Activity’s lifecycle, e.g.,
- When Activity pauses, its Fragments are paused
- When Activity is destroyed, its Fragments are destroyed
Fragment Lifecycle States

- Similar to Activity Lifecycle States
- Resumed
  - Fragment is visible in the running activity
- Paused
  - Another activity is in the foreground and has focus
  - The containing activity is still visible
- Stopped
  - The fragment is not visible
Fragment Lifecycle Callbacks

- **onCreate()**
  - Initial creation of the fragment
- **onStart()**
  - Fragment is visible to the user
- **onResume()**
  - Fragment is visible to the user and actively running
- **onPause()**
  - Fragment is visible, but does not have focus
- **onStop()**
  - Fragment is no longer visible
- **onDestroy()**
  - Fragment is no longer in use
Fragment Lifecycle Callbacks (cont.)

- **onAttach()**
  - Fragment is first attached to its activity
- **onCreateView()**
  - Fragment instantiates its user interface view
- **onActivityCreated()**
  - Fragment's activity created and Fragment’s view hierarchy instantiated
- **onDestroyView()**
  - View previously created by onCreateView() detached from the Fragment
- **onDetach()**
  - Fragment no longer attached to its activity
Fragment & Activity Lifecycles

**Activity Callbacks:**

- `onCreate()`
- `onStart()`
- `onResume()`
- `onPause()`
- `onStop()`
- `onDestroy()`

**Fragment Callbacks:**

- `onCreate()`
- `onDestroy()`
- `onCreateView()`
- `onDetach()`
- `onActivityCreated()`
Fragments usually, but not always, have a UI

Layout can be inflated/implemented in `onCreateView()`

- `onCreateView()` must return the View at the root of the Fragment’s layout
- The returned View will be added to the containing Activity
  - Container represented as a ViewGroup within the containing Activity’s view hierarchy
Two ways to add Fragments to an Activity’s layout

- Declare it statically in the Activity’s layout file
- Add it programmatically to a ViewGroup in the Activity’s layout
### Fragment Layout (cont.)

<table>
<thead>
<tr>
<th>FragmentStaticLayout</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hamlet</strong></td>
</tr>
<tr>
<td><strong>King Lear</strong></td>
</tr>
<tr>
<td><strong>Julius Ceasar</strong></td>
</tr>
<tr>
<td><strong>As flies to wanton boys, are we to the gods; they kill us for their sport.</strong></td>
</tr>
</tbody>
</table>
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    ...
}
<LinearLayout ...>
    <fragment
        class="edu.umd.cs.cmsc436.Fragments.TitlesFragment"
        android:id="@+id/titles" android:layout_weight="1" ...
    />
    <fragment
        class="edu.umd.cs.cmsc436.Fragments.DetailsFragment"
        android:id="@+id/details" android:layout_weight="2" ...
    />
</LinearLayout>
public View onCreateView(LayoutInflater inflater,
             ViewGroup container,
             Bundle savedInstanceState) {

    return inflater.inflate(R.layout.detail_fragment,
                container, false);

}
detail_fragment.xml

<ScrollView ...>
  <TextView android:id="@+id/quoteView"
  </Textview>
</ScrollView>
While Activity’s running you can add a Fragment to the Activity’s layout

- Specify a containing ViewGroup
- Get reference to FragmentManager
- Execute a FragmentTransaction
<LinearLayout xmlns:android="android:id=""@+id/activityFrame"
    android:orientation="horizontal" />
<FrameLayout android:id="@+id/titleFrame" ...
    android:layout_width="0dp" android:layout_weight="1">
</FrameLayout>
<FrameLayout android:id="@+id/detailFrame" ...
    android:layout_width="0dp" android:layout_weight="2">
</FrameLayout>
</LinearLayout>
private final mTitlesFragment = new TitlesFragment();
private final mDetailsFragment = new DetailsFragment();
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    ...
FragmentManager fragmentManager = getFragmentManager();
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
fragmentTransaction.add(R.id.titleFrame, mTitlesFragment);
fragmentTransaction.add(R.id.detailFrame, mDetailsFragment);
fragmentTransaction.commit();
}
TitlesFragment mTitlesFragment = new TitlesFragment();
DetailsFragment mDetailsFragment = new DetailsFragment();
...
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    ...
    setContentView(R.layout.main);
    ...
}
<LinearLayout>
    android:id="@+id/activityFrame"
    android:orientation="horizontal"
    ...
</LinearLayout>
mFragmentManager = getFragmentManager();
FragmentTransaction fragmentTransaction =
    mFragmentManager.beginTransaction();
// adding TitlesFragment
fragmentTransaction.add(R.id.activityFrame, mTitlesFragment);
fragmentTransaction.commit();
}
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    LinearLayout ll = new LinearLayout(getActivity());
    ll.setLayoutParams(new LinearLayout.LayoutParams(0, ViewGroup.LayoutParams.MATCH_PARENT, 1.0f));
    ll.addView(super.onCreateView(inflater, container, savedInstanceState));
    return ll;
}
... public void onListSelection(int index) {
    if (!mDetailsFragment.isAdded()) {
        FragmentTransaction fragmentTransaction =
            mFragmentManager.beginTransaction();
        // adding DetailsFragment
        fragmentTransaction.add(R.id.activityFrame, mDetailsFragment);
        // reverse this transaction when Back button is hit
        fragmentTransaction.addToBackStack(null);
        fragmentTransaction.commit();
        mFragmentManager.executePendingTransactions();
    }
...
public View onCreateView(LayoutInflater inflater,
    ViewGroup container, Bundle savedInstanceState) {
    return inflater.inflate(R.layout.detail_fragment, container, false);
}
detail_fragment.xml

<LinearLayout ...
    android:id="@+id/detail_linear_layout" ...
    android:layout_width="0dp" android:layout_weight="2">
    <ScrollView android:id="@+id/ScrollView1"
        android:orientation="vertical" ...>
        <TextView android:id="@+id/quoteView"
            ... />
    </ScrollView>
</LinearLayout>
Fragment Lifecycle Summary

- **Fragment Added**
  - onCreateView()
  - onCreateView()
  - onCreate()
  - onAttach()
  - onStart()
  - onResume()
  - onPause()
  - onStop()

- **Fragment Active**
  - Fragment added to backstack, then removed/replaced
  - User navigates back or Fragment removed/replaced

- **Fragment returns to layout**

- **Fragment destroyed**
  - onDestroyView()
  - onDestroy()
  - onDetach()
Action Bar

- Menu widget for Activities
  - Places the traditional title bar at the top of the screen.
- By default, Action Bar includes
  - Application icon
  - Activity title
  - Items from any Options Menus
<table>
<thead>
<tr>
<th>FragmentDynamicLayoutWithActionBar</th>
<th>ActivityMenuItem</th>
<th>TitleMenuItem</th>
<th>DetailMenuItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hamlet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>King Lear</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Julius Ceasar</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

There is a tide in the affairs of men, Which taken at the flood, leads on to fortune. Omitted, all the voyage of their life is bound in shallows and in miseries.
ActionBar Items

- ActionBar items work like menus
- Will discuss menus in more detail in a later class
<menu ...>
  <item android:id="@+id/activity_menu_item"
       android:title="ActivityMenuItem"
       android:showAsAction="always">
  </item>
</menu>
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.activity_menu, menu);
    return true;
}

public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.activity_menu_item:
        ...
        return true;
    default:
        return super.onOptionsItemSelected(item);
    }
}

public boolean onOptionsItemSelectedListener(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.activity_menu_item:
        ...
        return true;
    default:
        return super.onOptionsItemSelectedListener(item);
    }
}
setHasOptionsMenu(true);

public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
    inflater.inflate(R.menu.detail_menu, menu);
}

public boolean onOptionsItemSelected(MenuItem item) {
    ...
}

public void onOptionsItemSelected(MenuItem item) {
    ...
}
Source Code Examples

- FragmentStaticLayout
- FragmentProgrammaticLayout
- FragmentDynamicLayout
- FragmentDynamicLayoutWithActionBar