Android UI Adapters

Sang Shin
Michèle Garoche
www.javapassion.com
“Learn with Passion!”

Disclaimer

- Portions of this presentation are modifications based on work created and shared by the Android Open Source Project
  - http://code.google.com/policies.html
- They are used according to terms described in the Creative Commons 2.5 Attribution License
  - http://creativecommons.org/licenses/by/2.5/

Topics

- AdapterView & Adapter
- AdapterView responsibilities
- ListActivity, ListView, and ListAdapter
- Spinner
- Gallery

AdapterView & Adapter

What is AdapterView Class?

- The *AdapterView* is a child class of *ViewGroup*
  - A special kind of container of view objects (list items)
- Typically you are going to use subsclasses of *AdapterView* class instead of using it directly
- Example subclasses of *AdapterView* class
  - *ListView*
  - *Spinner*
  - *Gallery*
- An *AdapterView* access the data through *Adapter* object
  - Instead of accessing data directly itself

What is an Adapter?

- An *Adapter* object acts as a bridge between an *AdapterView* object and the underlying data for that view.
  - The Adapter provides access to the data items.
- The Adapter is also responsible for making a View for each item in the data set.
- Types of Adatpers - they implements *ListAdatper interface*
  - *ArrayAdatper*
  - *CursorAdatper*
  - *There are a few more*

Adapter Class Hierarchy

- *BaseAdatper* abstract class implements *ListAdapter* and *SpinnerAdatper* interfaces
- *ArrayAdapter* and *CursorAdapter* classes are subclasses of *BaseAdapter* class
- You can create a custom adaptor by extending *BaseAdapter* class

AdapterView Responsibilities

AdapterView Responsibilities

- Two main responsibilities of AdapterView
  - Filling the layout with data (it received through the help of an Adapter)
  - Handling user selections - when a user selects an item, perform some action

Filling the Layout with Data

- Inserting data into the layout is typically done by binding the AdapterView class to an Adapter, which retrieves data from an external source (perhaps a list that the code supplies or query results from the device's database).

Handling User Selections

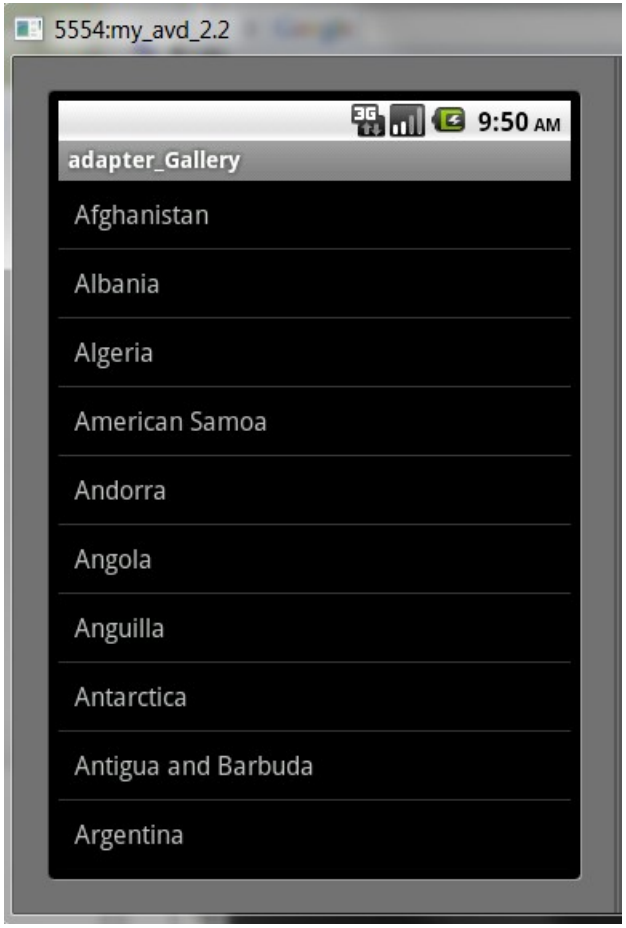
- You handle the user's selection by setting the class's *AdapterView.OnItemClickListener* member to a listener and catching the selection changes

```
// Create a message handling object as an anonymous class.
private OnItemClickListener mMessageClickedHandler = new
OnItemClickListener() {
    public void onItemClick(AdapterView parent, View v, int position, long id){
        // Display a messagebox.
        Toast.makeText(mContext,"You've got an event",
                                    Toast.LENGTH_SHORT).show();
    }
};

// Now hook into our object and set its onItemClickListener member
// to our class handler object.
mHistoryView = (ListView)findViewById(R.id.my_list_view);
mHistoryView.setOnItemClickListener(mMessageClickedHandler);
```

ListView

ListView Class

- A child class of AdapterView class
- Shows items in a vertically scrolling list.
- The items come from the ListAdapter associated with this view



Two Choices of Activity Class

- Option #1 - Your activity extends Activity class
  - You have to create ListView object yourself from resource file just like any other View object
- Option #2 - Your activity extends ListActivity class
  - ListView object gets created by the ListActivity's contructor, so you don't need to create it yourself

Option #1 - Extending Activity Class

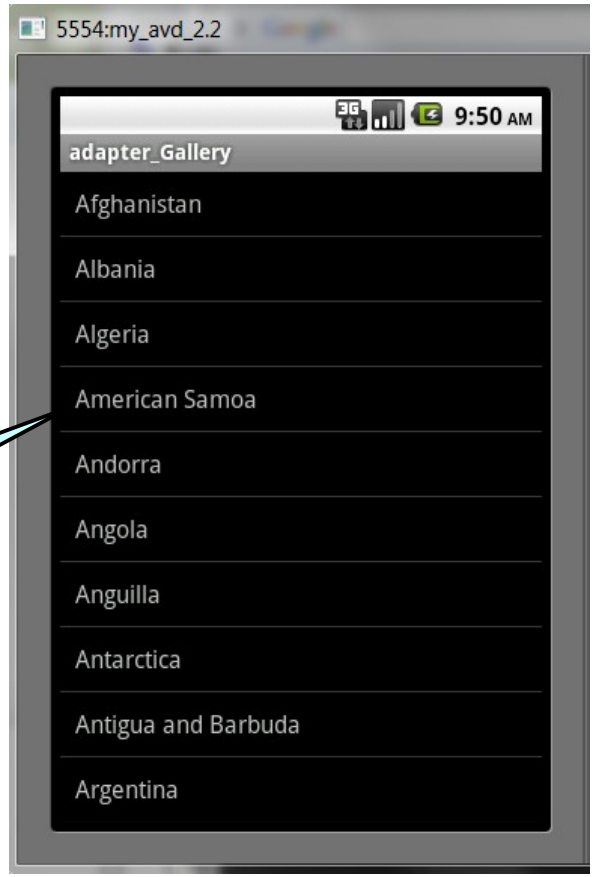
```
public class HelloListView extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        // Since HelloListView extends Activity (instead of ListActivity),
        // we have to create ListView object ourselves.
        ListView lv =(ListView)findViewById(R.id.listview);

        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(
                this,                   // Application context
                R.layout.list_item,     // layout description for each list item
                COUNTRIES);

        lv.setAdapter(arrayAdapter);

    }
```

Example of ListView Layout

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <ListView
        android:id="@+id/listview"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Example of List Item Layout

```
<?xml version="1.0" encoding="UTF-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="10dp"
    android:textSize="16sp"
    >
</TextView>
```



List Item Layout

Option #2: *ListActivity* Activity class

- Android-provided utility class specially designed for displaying a list of items by binding to a data source such as an array or Cursor, and exposes event handlers when the user selects an item.
  - *ListActivity* hosts a *ListView* object that can be bound through an *adatper* to different data sources, typically either an array or a Cursor holding query results.
  - *setListAdapter(ListAdatper adapter)* method automatically creates *ListView* object from the *ListAdapter* object
- Has a default layout that consists of a single, full-screen list in the center of the screen

Option #2: Extending ListActivity

```
public class HelloListView extends ListActivity {


    // Array as a data source
    static final String[] COUNTRIES = new String[] {
        "Yemen", "Yugoslavia", "Zambia", "Zimbabwe"
      };

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Create an adapter from Array data source object
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(
                this,                   // Application context
                R.layout.list_item,     // layout description for each list item
                COUNTRIES);             // String array of countries defined

        // Notice that this does not load a layout file for the Activity (which you
        // usually do with setContentView(int)). Instead, setListAdapter(ListAdapter)
        // automatically adds a ListView to fill the entire screen of the ListActivity.
        setListAdapter(arrayAdapter);

    }
}
```

Spinner

Spinner Class

- A child class of *AdapterView* class
- Displays one child at a time and lets the user pick among them.
- The items in the Spinner come from the Adapter associated with this view
- There is NO special SpinnerActivity class, so you have to create Spinner object yourself

Example of Spinner

```
public class HelloSpinner extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Spinner spinner = (Spinner) findViewById(R.id.spinner);
        ArrayAdapter<CharSequence> adapter =
                ArrayAdapter.createFromResource(
                        this,
                        R.array.planets_array,
                        android.R.layout.simple_spinner_item);

        adapter.setDropDownViewResource(
                        android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);
        spinner.setOnItemSelectedListener(new MyOnItemSelectedListener());
    }
```

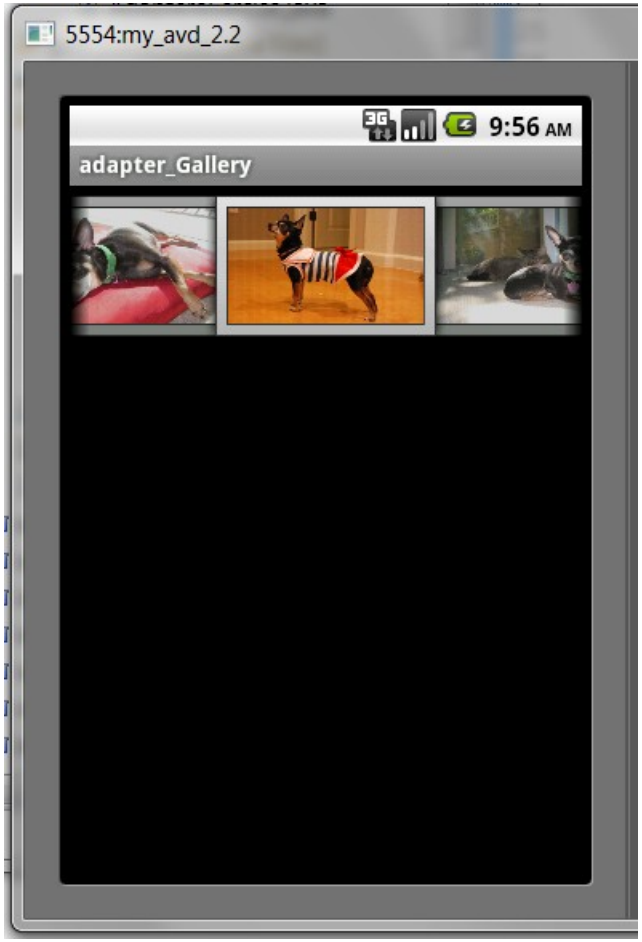
Example of Spinner Layout

```
<?xml version="1.0" encoding="UTF-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="10dip"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dip"
        android:layout_marginBottom="10dip"
        android:text="@string/planet_prompt"
    />
    <Spinner
        android:id="@+id/spinner"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:prompt="@string/planet_prompt"
    />
</LinearLayout>
```

Gallery

Gallery Class

- A child class of AdapterView class
- A view that shows items in a center-locked, horizontally scrolling list.



Example of Gallery

```
public class HelloGallery extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.main);
        Gallery g = (Gallery) findViewById(R.id.gallery);

        g.setAdapter(new ImageAdapter(this));

        g.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView parent,
                    View v, int position, long id) {
                Toast.makeText(HelloGallery.this, "" +
                    position, Toast.LENGTH_SHORT).show();
            }
        });
    }
```

Example of Gallery Layout

```
<?xml version="1.0" encoding="utf-8"?>
<Gallery xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gallery"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
/>
```

Thank you!

Check JavaPassion.com Codecamps!
http://www.javapassion.com/codecamps
"Learn with Passion!"