Android – Building user interfaces

Erlend Stav
SINTEF

SINTEF Android workshop, November 17, 2009

Overview

- Two ways of creating user interfaces
- Steps for a simple example
  - Adding a button to the user interface
  - Adding a new string to use from the button
  - Adding identifier
  - Handling user interface events from the code
- Views in Android
- Exploring the Android examples
  - Importing examples in Eclipse

Two ways of creating UI

- Instantiate layout elements from Java
- Declare UI elements in XML
  - Visual aspects separate from behaviour
  - Customization for different languages, and for different screen orientations and sizes

- In Eclipse the UI Editor simplifies defining the XML
- In this presentation we will use the Eclipse UI Editor

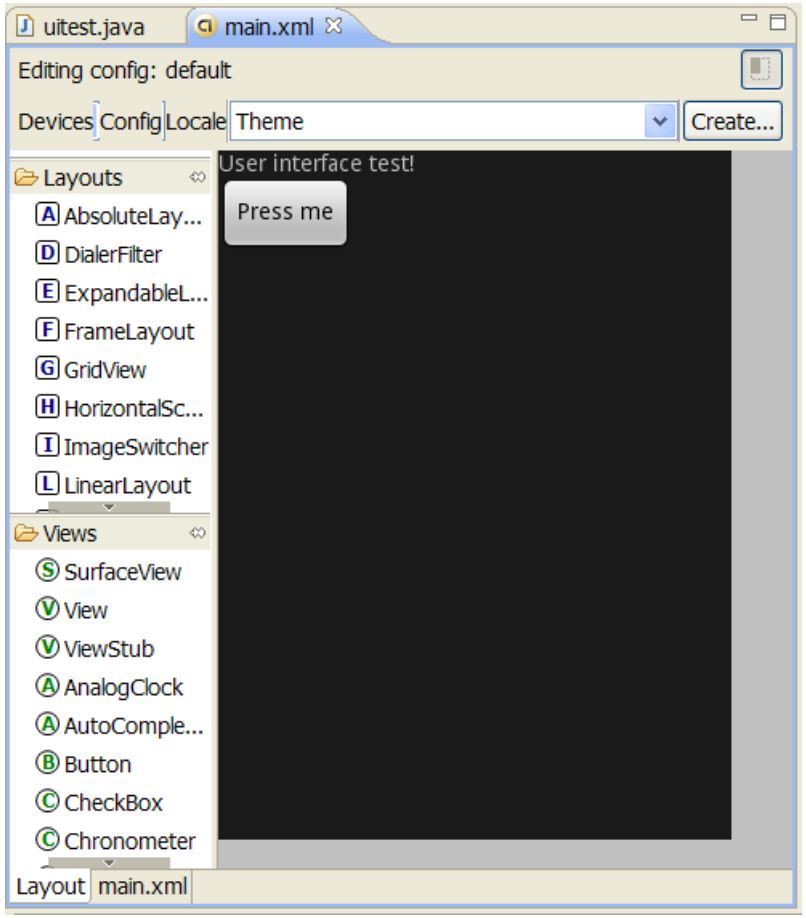
Example

- Simple example including a text label and a button
- When the button is pressed, change the text of the label

- Four steps to set up example

Step 1: Add button to user interface

- The first step is to add a button to the layout
- Drag Button from View palette into the layout

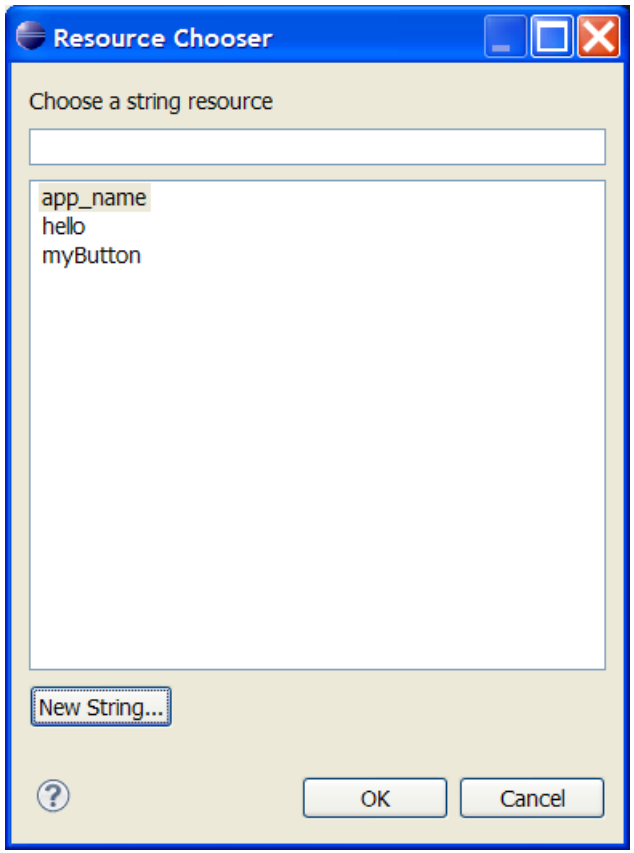


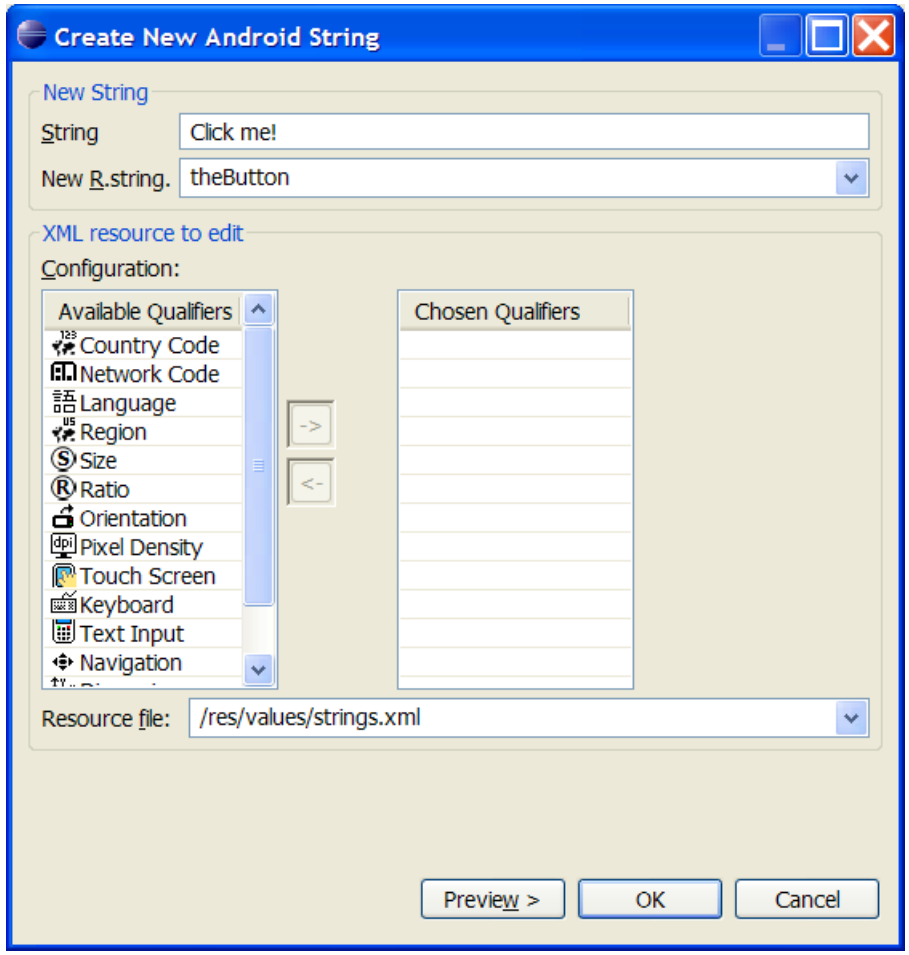
XML view for main.xml


```
<?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"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<Button
    android:id="@+id/Button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/theButton"
    ></Button>
</LinearLayout>
```

Step 2: Add a text for the button

- Click "…" for the text property of the button
- Click "New String", and enter e.g. "theButton" as name and "Click me!" as string





Step 3: Set identifiers for button and label

- To access user interface elements from the code, we need to have an id to locate them by
- For the button, set the id property e.g. to @+id/theButton
- When a layout with a property set in this manner is saved, the R.java file will be updated and contain a constant called theButton in the id class of the file.

- Select the label and set the id property e.g. to @+id/theLabel

Step 4: Handling user interface events from the code

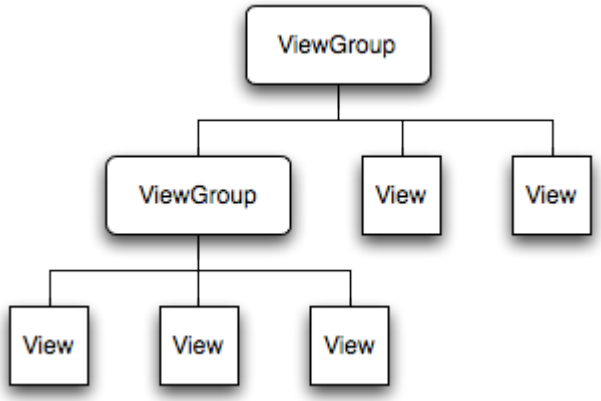

```
public class UItest extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button theButton = (Button)findViewById(R.id.theButton);
        theButton.setOnClickListener( new OnClickListener() {
                public void onClick(View arg0) {
                  theButtonPressed();
                }
        });
    }

    public void theButtonPressed() {
        TextView theLabel = (TextView)findViewById(R.id.theLabel);
        theLabel.setText("Hi there!");
    }
}
```

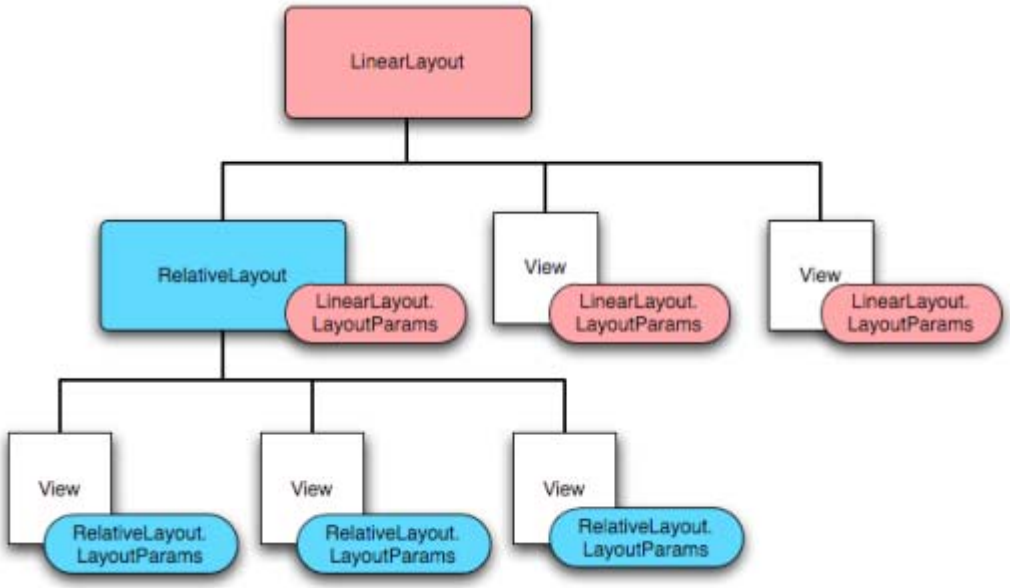
Views in Android

- Views are the basic units of the user interface in Android UI
- A ViewGroup is a kind of view that can contain other views (and ViewGroups). The main set of predefined ViewGroups in Android are called Layouts
- Concrete views for interaction with users, like buttons, check boxes, and text entry fields are called Widgets



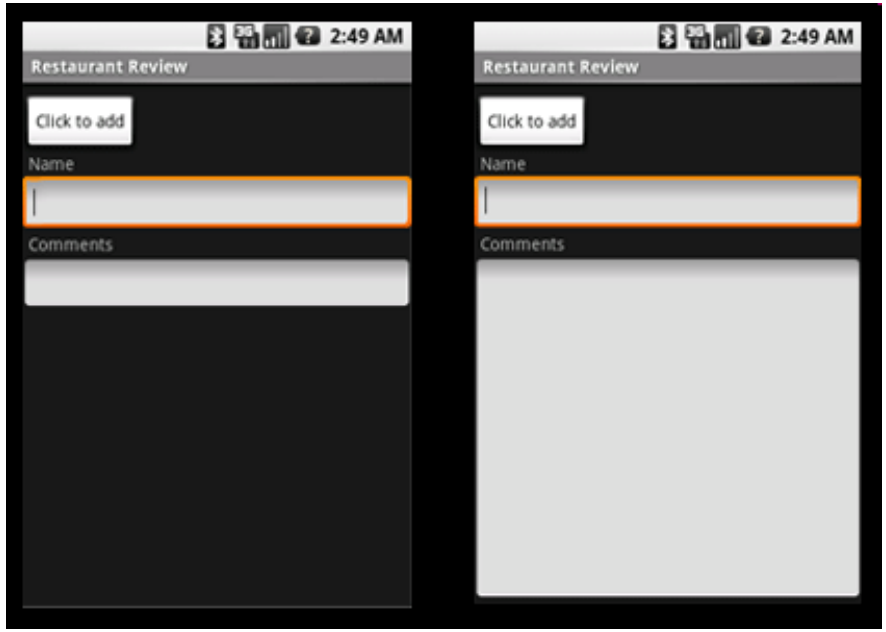
Layouts

- View groups, such as layouts, provide hierarchy and organization such as linear, tabular, etc.
- Views contained in a layout have associated parameters which control their presentation within the layout
- Overview of main layouts: http://developer.android.com/guide/topics/ui/layout-objects.html



Example: LinearLayout

- Default layout for the main.xml when creating in new Eclipse Android project
- Align children in a single direction – set orientation property to either horizontally or vertically
- Children can be set to fill the parent in a direction, to wrap based on their content, and may have a weight for importance



Adding a new activity and UI

- To add a resource containing a new GUI
  - From the “New…” menu select “Android XML file”
  - From the dialog
    - select filename for the new GUI
    - check the Layout radio button
    - select the root element to use; e.g. LinearLayout, RelativeLayout, or other of your preference
- To add a new activity
  - From the “New…” menu select “Class “
  - Fill in name for the class and the package
  - For Subclass, type/browse to select android.app.Activity
  - To add the standard onCreate method, you can copy from an existing activity, and adjust the setContentView() to refer to the new GUI resource you created

Exploring Android examples from Eclipse

- There is a number of official examples available for the Android platform
- The examples are downloaded as part of the SDK, and are available under:
  - .\platforms\Android-X.Y\samples
- The ApiDemos example contains a rich set of examples, and illustrates many of the available layouts and widgets
- The other examples contains more focused and complete applications

Import example

- Select “New” and “Android Project”
- From dialog, select “Create project from existing sample”
- Select project from samples list
- Click “Finish”