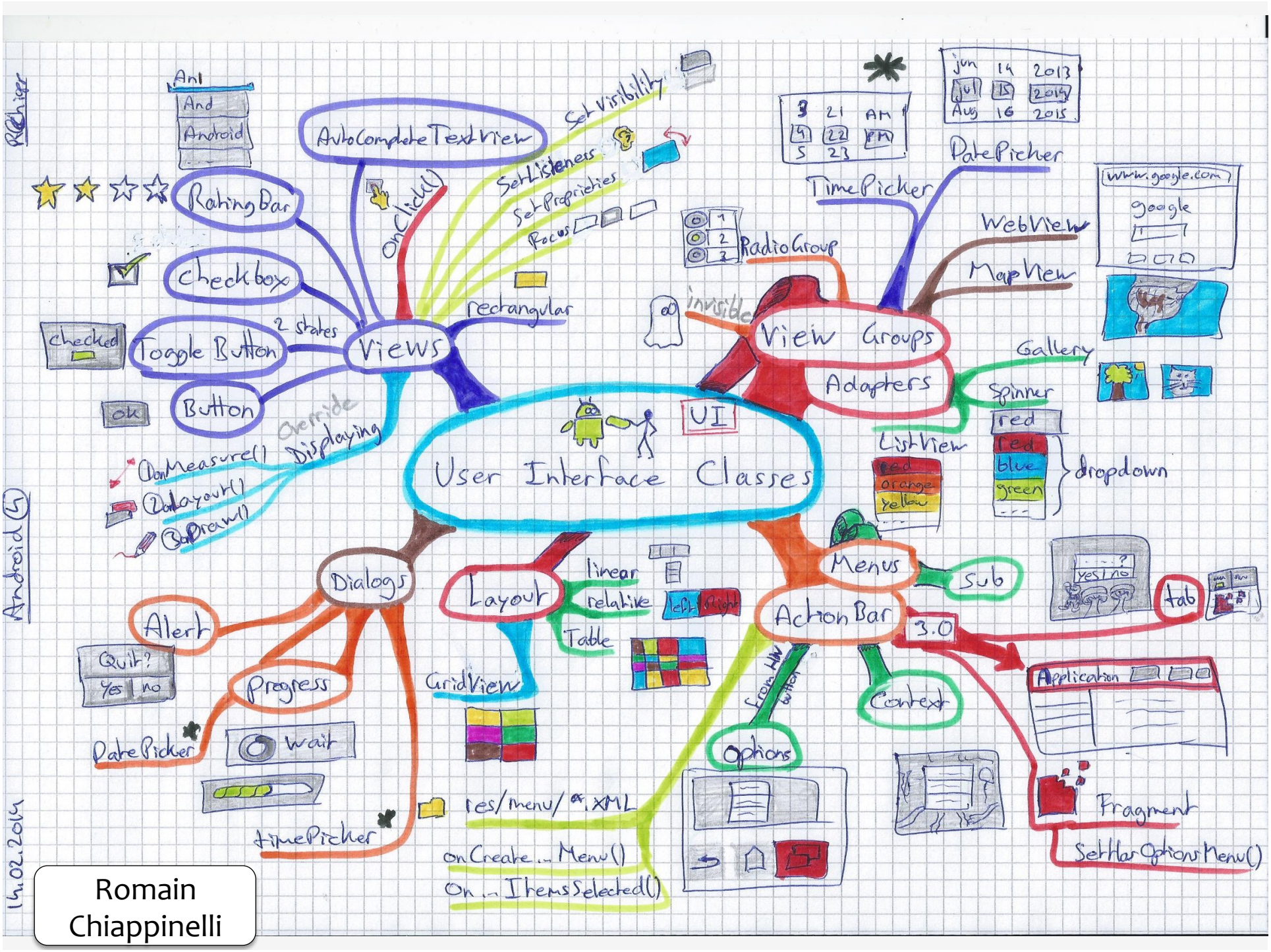


Romain Chiappinelli

Programming Handheld Systems

Adam Porter

User Interface Classes

Today's Topics

Views & View Events

View Groups, AdapterViews & Layouts

Menus & ActionBar

Dialogs

Android User Interfaces

Activities usually display a user interface

Android provides many classes for constructing user interfaces

View

Key building block for UI components

Occupy a rectangular space on screen

Responsible for drawing themselves and for handling events

Some Predefined Views

Button

ToggleButton

Checkbox

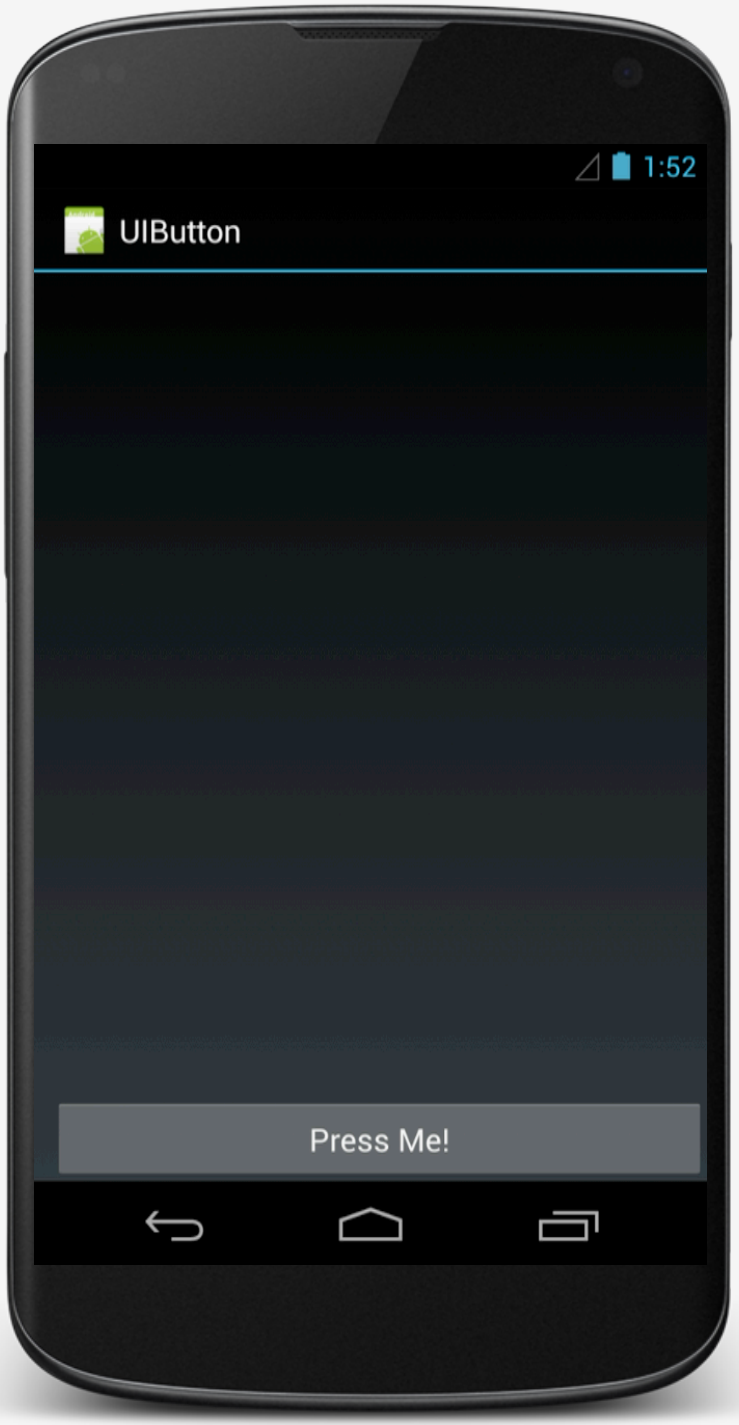
RatingBar

AutoCompleteTextView

Button

View that can be clicked on to perform an action

UIButton



UIButton

```
<Button
    android:id="@+id/button"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_marginLeft="10dip"
    android:text="@string/press_me_string" >
</Button>
```

```
// Get a reference to the Press Me Button
final Button button = (Button) findViewById(R.id.button);

// Set an OnClickListener on this Button
// Called each time the user clicks the Button
button.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {

        // Maintain a count of user presses
        // Display count as text on the Button
        button.setText("Got Pressed:" + ++count);

    }
});
```

ToggleButton

A 2-state Button

checked/not checked state

Light indicator showing state

UIToggleButton

UIToggleButton

```
<ToggleButton
    android:id="@+id/togglebutton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textOff="@string/start_string"
    android:textOn="@string/stop_string"
    android:textSize="24sp" />
```

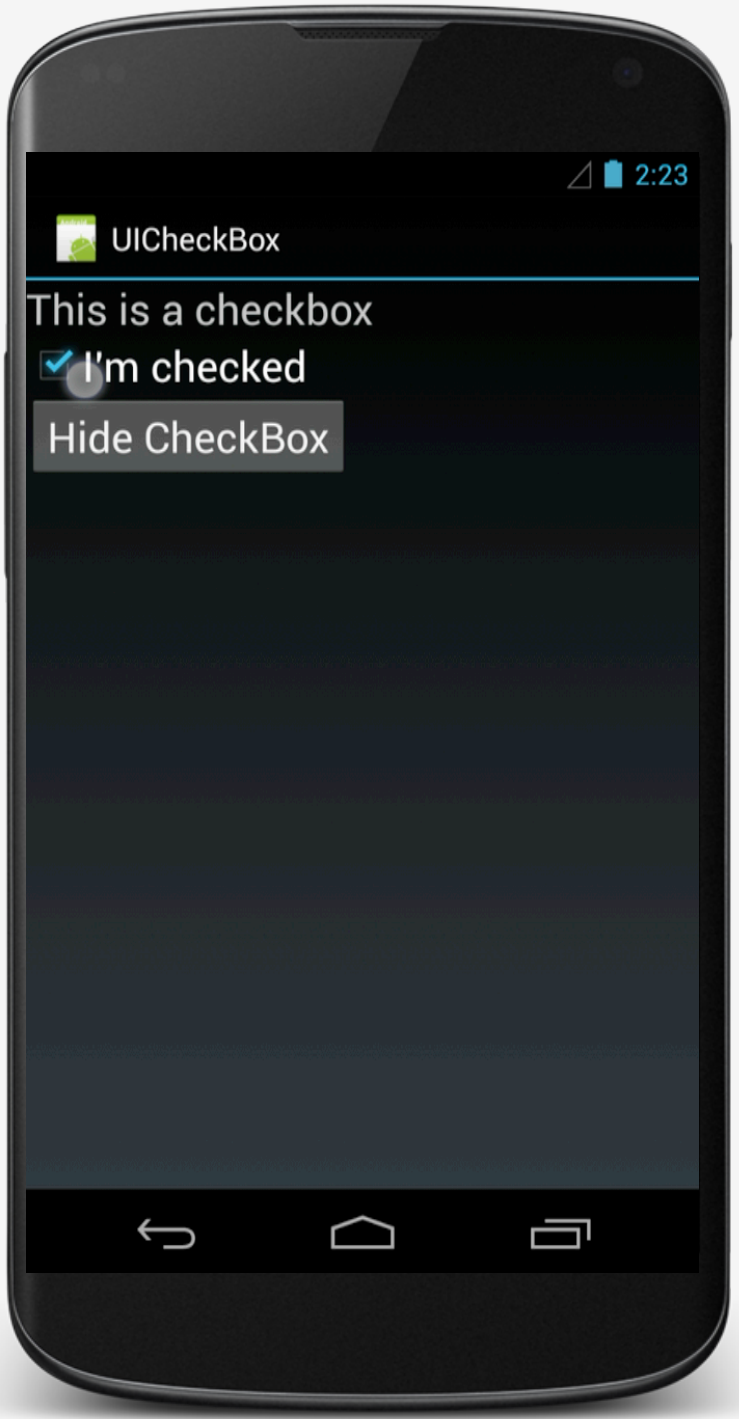
```
// Get a reference to the ToggleButton
final ToggleButton button = (ToggleButton) findViewById(R.id.togglebutton);

// Set an OnClickListener on the ToggleButton
button.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {

        // Toggle the Background color between a light and dark color
        if (button.isChecked()) {
            bg.setBackgroundColor(0xFFF3F3F3);
        } else {
            bg.setBackgroundColor(0xFF000000);
        }
    }
});
```

CHECKBOX

Another kind of 2-state button

Checked/not checked

UICheckbox



UICheckbox

```
<CheckBox
    android:id="@+id/checkbox"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/im_not_checked_string"
    android:textSize="24sp"/>
```

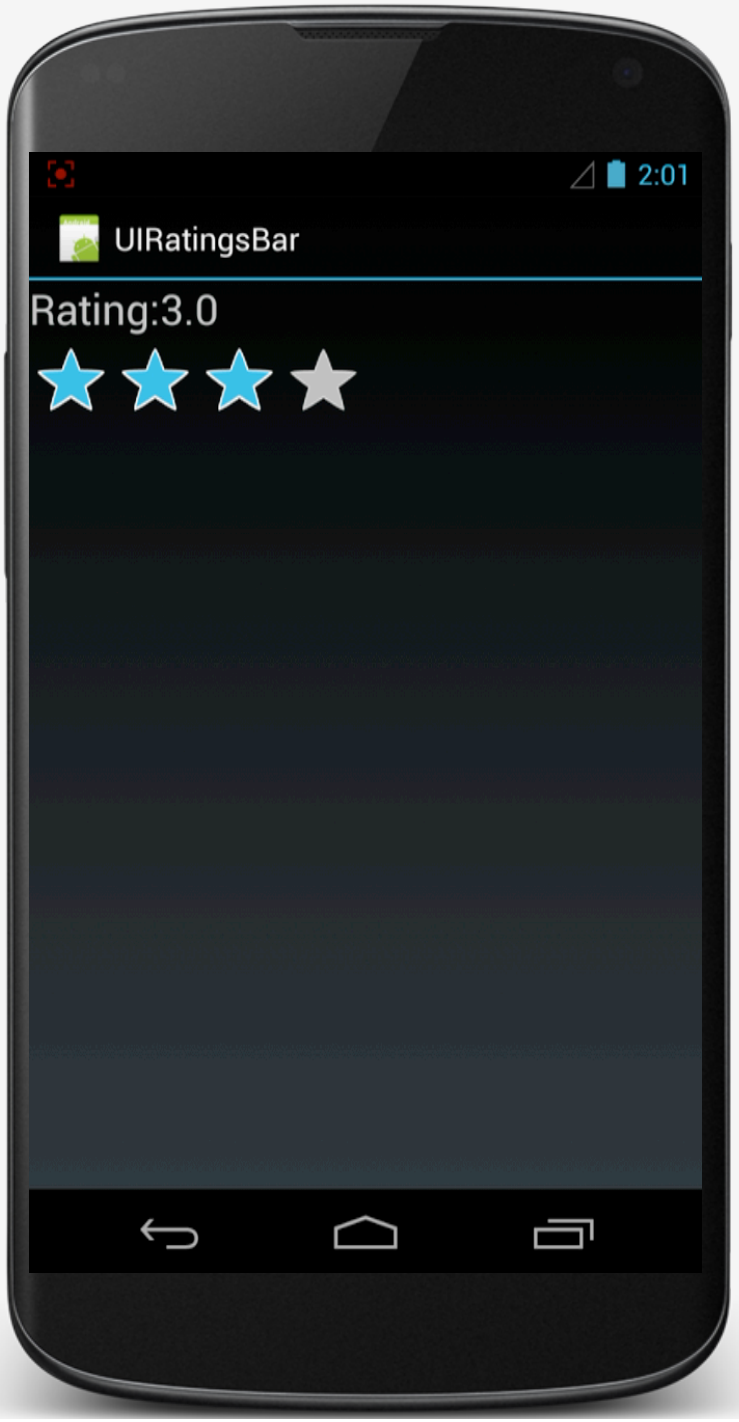
```
// Get a reference to the CheckBox
final CheckBox checkbox = (CheckBox) findViewById(R.id.checkbox);

// Set an OnClickListener on the CheckBox
checkbox.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {

        // Check whether CheckBox is currently checked
        // Set CheckBox text accordingly
        if (checkbox.isChecked()) {
            checkbox.setText("I'm checked");
        } else {
            checkbox.setText("I'm not checked");
        }
    }
});
```

RatingBar

A view comprising a row of stars

The user can click or drag the stars to highlight some number of them

UIRatingBar



UIRatingBar

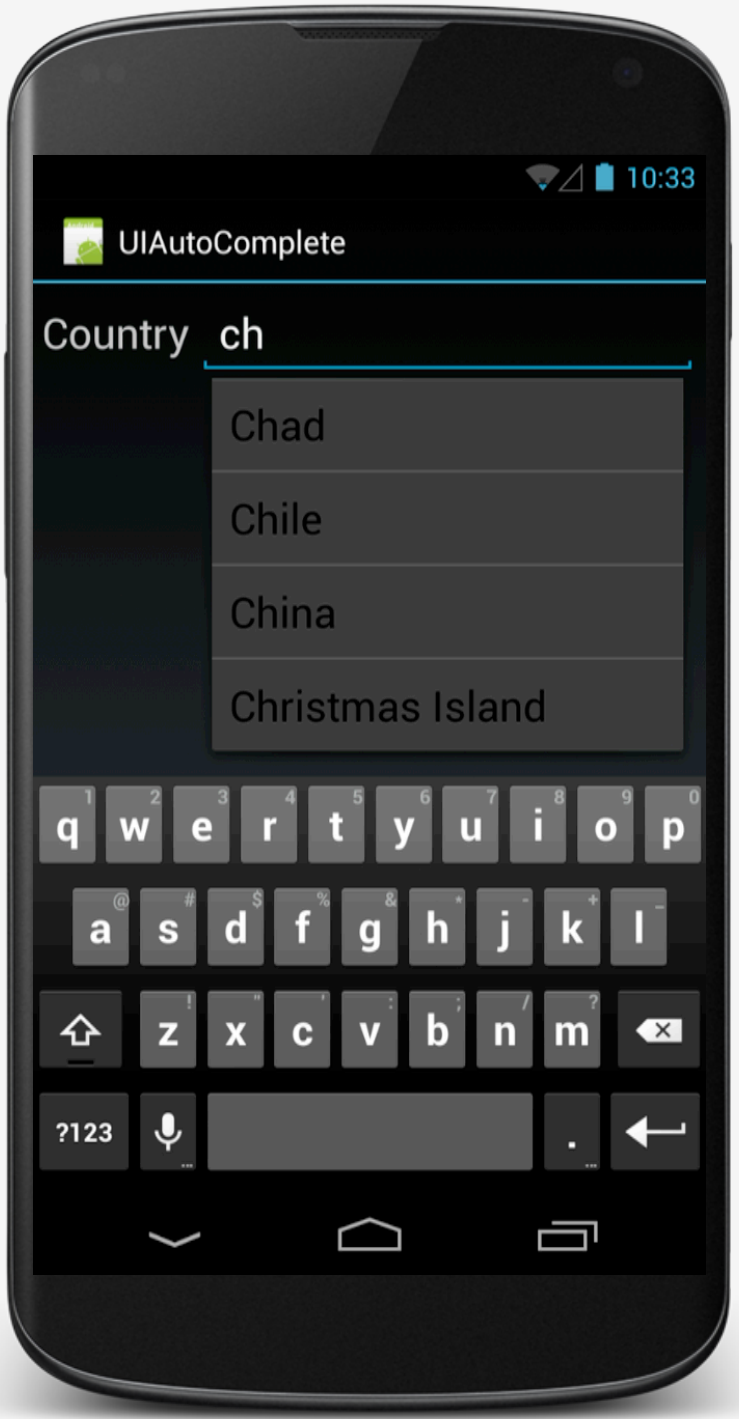
```
<RatingBar
    android:id="@+id/ratingbar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:numStars="4"
    android:stepSize="1.0" >
</RatingBar>
```

```
final RatingBar bar = (RatingBar) findViewById(R.id.ratingbar);

bar.setOnRatingBarChangeListener(new OnRatingBarChangeListener() {

    // Called when the user swipes the RatingBar
    @Override
    public void onRatingChanged(RatingBar ratingBar, float rating, boolean fromUser) {
        tv.setText("Rating:" + rating);
    }
});
```

AutoCompleteTextView

An editable text field that provides completion suggestions as the user types in text

UIAutoComplete TextView



UIAutoComplete

```
<AutoCompleteTextView
    android:id="@+id/autocomplete_country"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="5dp"
    android:textSize="24sp"/>
```

```
// Get a reference to the AutoCompleteTextView
AutoCompleteTextView textView = (AutoCompleteTextView) findViewById(R.id.autocomplete_country);

// Create an ArrayAdapter containing country names
ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
        R.layout.list_item, COUNTRIES);

// Set the adapter for the AutoCompleteTextView
textView.setAdapter(adapter);
```

Common View operations

Set visibility: show or hide view

Set Checked state

Set Listeners: Code that should be executed when specific events occur

Set properties: opacity, background, rotation

Manage input focus: allow view to take focus, request focus

View Event Sources

- User interaction
  - Touch
  - Keyboard/trackball/D-pad
- System control
  - Lifecycle changes

Handling View Events

Often handle events with listeners

Numerous Listener interfaces defined by the View class

View Listener interfaces

OnClickListener.onClick()

View has been clicked

OnLongClickListener.onLongClick()

View has been pressed & held

View Listener interfaces

OnFocusChangeListener.
onFocusChange()

View has received or lost focus

OnKeyListener.onKey()

View about to receive a hardware key press

Displaying Views

Views are organized in a Tree

Displaying has multiple steps

Measure – get dimensions of each View

Layout – Position each View

Draw – Draw each view

Handling View Events

Custom View subclasses can override various View methods

Handling View Events

onMeasure()

- Determine the size of this View and its children

onLayout()

- View must assign a size and position to all its children

onDraw()

- View should render its content

Handling View Events

onFocusChanged()

View's focus state has changed

onKeyUp(), onKeyDown()

A hardware key event has occurred

onWindowVisibilityChanged()

Window containing view has changed its visibility status

ViewGroup

An invisible View that contains other views

Used for grouping & organizing a set of views

Base class for view containers & layouts

Some Predefined ViewGroups

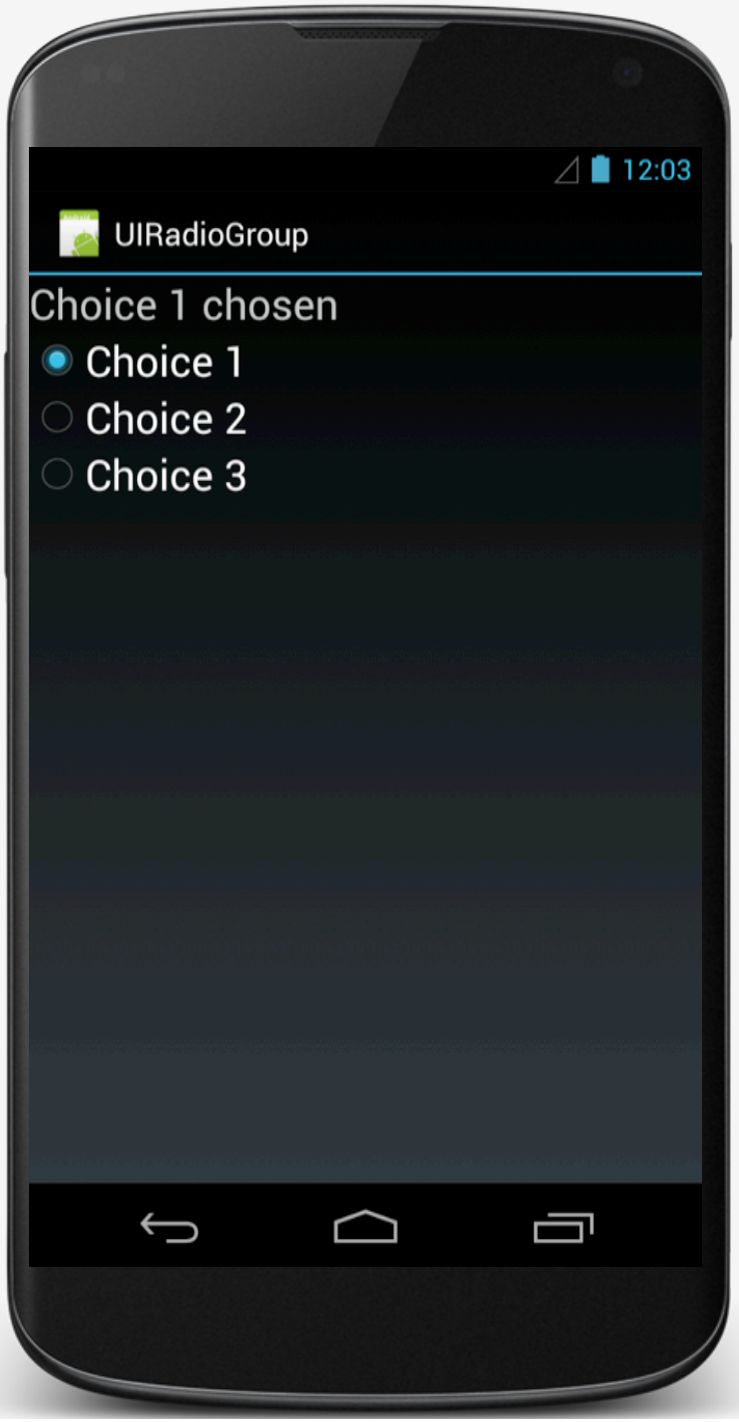
- RadioGroup 
- TimePicker
- DatePicker
- WebView
- MapView
- Gallery
- Spinner

RadioGroup

A ViewGroup containing a set of Radio Buttons (CheckBoxes)

Only one button can be selected at any one instant

UIRadioGroup



UIRadioGroup

```
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    >

    <RadioButton
        android:id="@+id/choice1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_1_string"
        android:textSize="24sp"/>

    <RadioButton
        android:id="@+id/choice2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_2_string"
        android:textSize="24sp"/>

    <RadioButton
        android:id="@+id/choice3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_3_string"
        android:textSize="24sp" />
</RadioGroup>
```

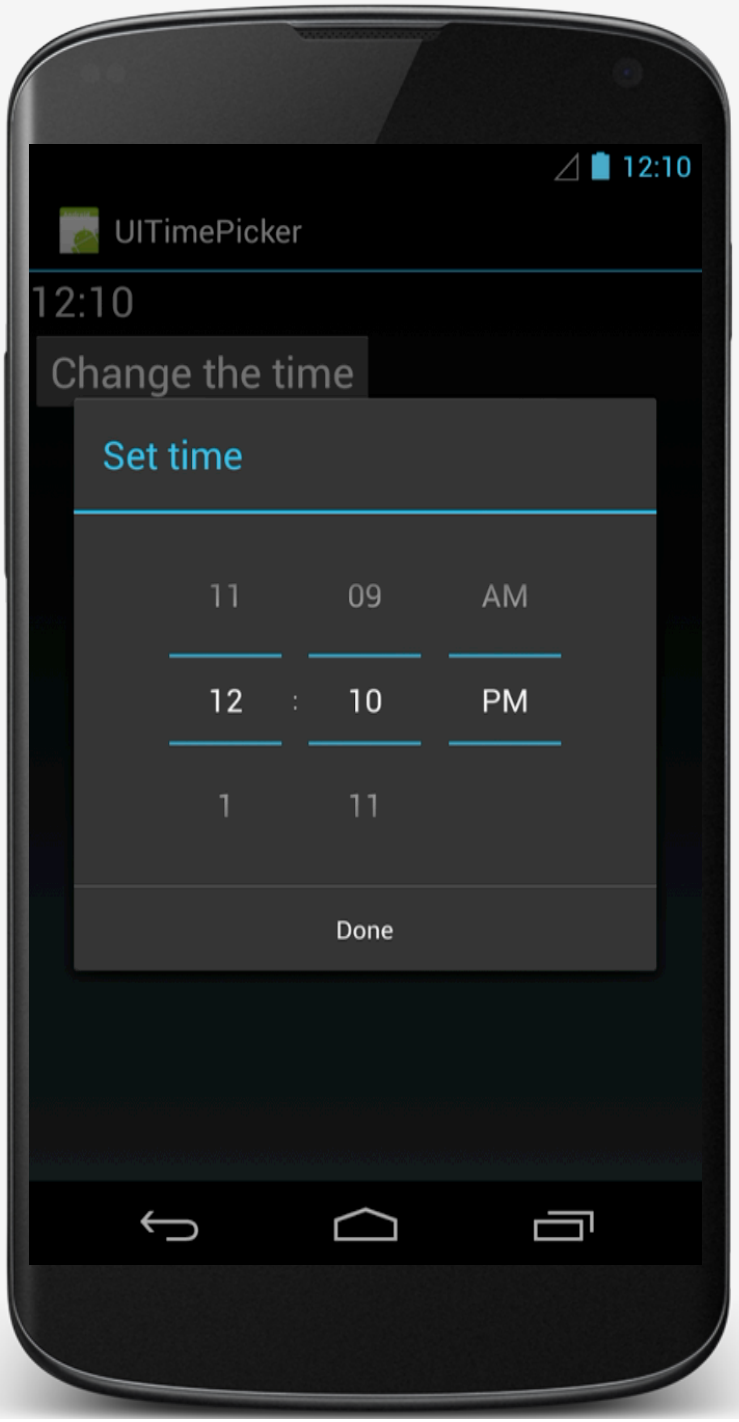
UIRadioGroup

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

    final TextView tv = (TextView) findViewById(R.id.textView);

    // Define a generic listener for all three RadioButtons in the RadioGroup
    final OnClickListener radioListener = new OnClickListener() {
        @Override
        public void onClick(View v) {
            RadioButton rb = (RadioButton) v;
            tv.setText(rb.getText() + " chosen");
        }
    };

    final RadioButton choice1 = (RadioButton) findViewById(R.id.choice1);
    // Called when RadioButton choice1 is clicked
    choice1.setOnClickListener(radioListener);

    final RadioButton choice2 = (RadioButton) findViewById(R.id.choice2);
    // Called when RadioButton choice2 is clicked
    choice2.setOnClickListener(radioListener);

    final RadioButton choice3 = (RadioButton) findViewById(R.id.choice3);
    // Called when RadioButton choice3 is clicked
    choice3.setOnClickListener(radioListener);

}
```

TimePicker

A ViewGroup that allows the user to select a time

UITimePicker



UITimePicker

```
<TextView
    android:id="@+id/timeDisplay"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:textSize="24sp" />

<Button
    android:id="@+id/pickTime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/change_the_time_string"
    android:textSize="24sp" />
```

UITimePicker

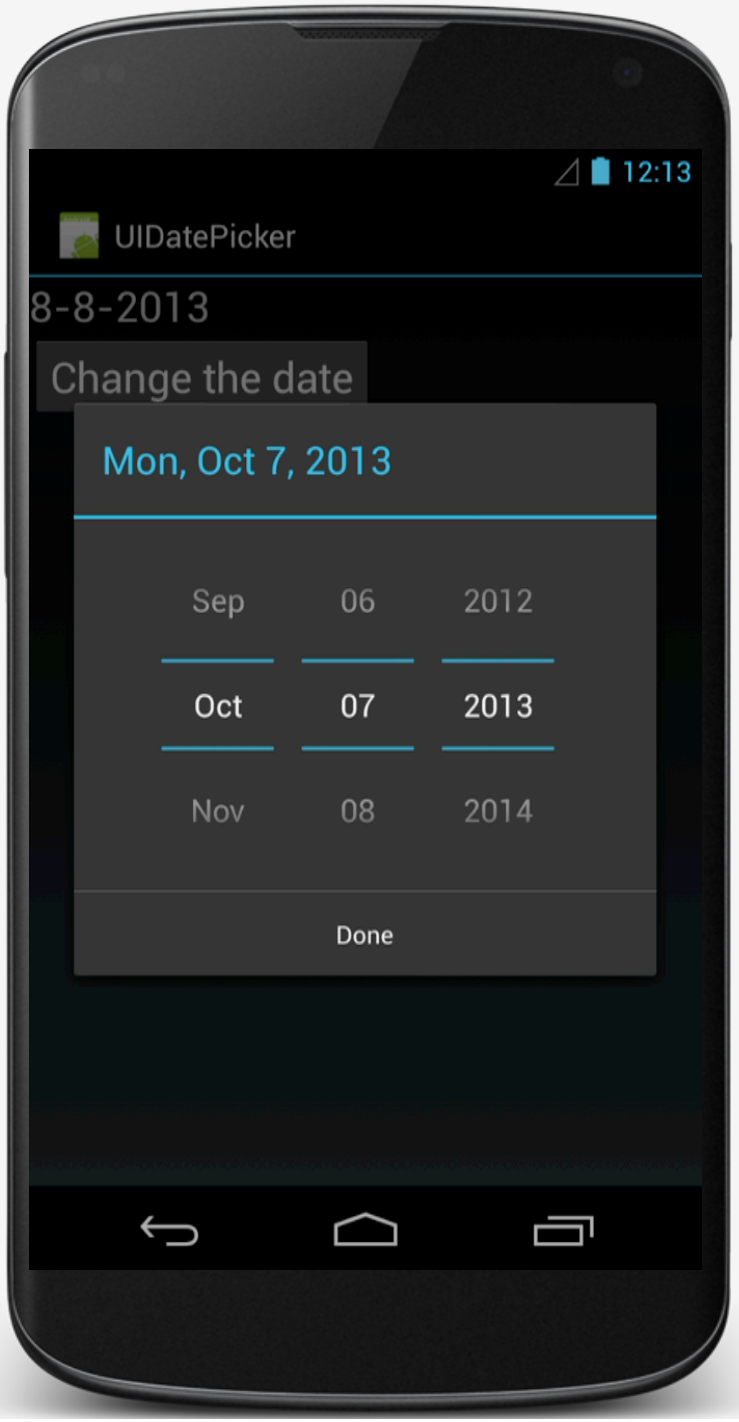
```
// The callback received when the user "sets" the time in the dialog
private TimePickerDialog.OnTimeSetListener mTimeSetListener = new TimePickerDialog.OnTimeSetListener() {
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        mHour = hourOfDay;
        mMinute = minute;
        updateDisplay();
    }
};
```

```
@Override
protected Dialog onCreateDialog(int id) {
    switch (id) {
    case TIME_DIALOG_ID:
        return new TimePickerDialog(this, mTimeSetListener, mHour, mMinute,
                false);
    }
    return null;
}
```

DatePicker

A ViewGroup that allows the user to select a date

UIDatePicker



UIDatePicker

```
<TextView
    android:id="@+id/dateDisplay"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:textSize="24sp" />

<Button
    android:id="@+id/pickDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/change_the_date_string"
    android:textSize="24sp" />
```

UIDatePicker

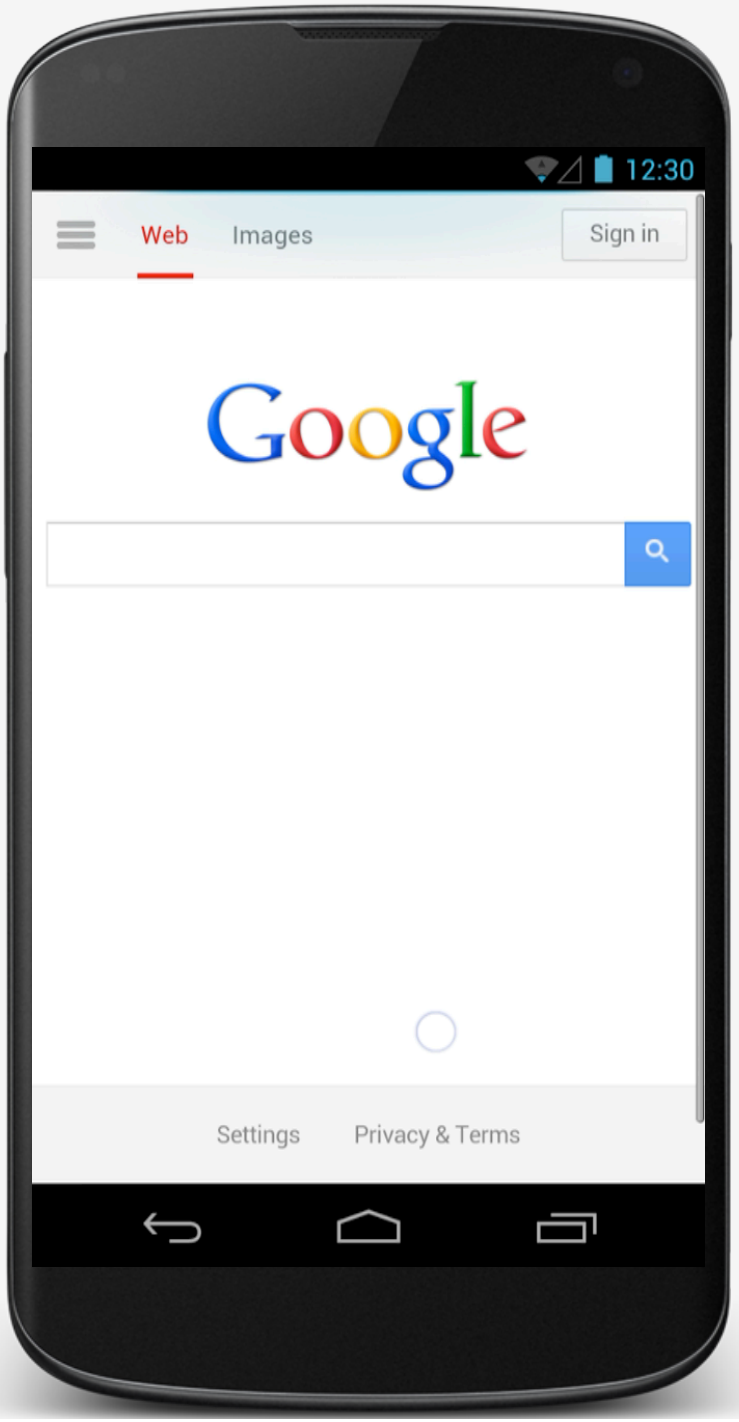
```
// The callback received when the user "sets" the date in the Dialog
private DatePickerDialog.OnDateSetListener mDateSetListener = new DatePickerDialog.OnDateSetListener() {

    public void onDateSet(DatePicker view, int year, int monthOfYear,
            int dayOfMonth) {
        mYear = year;
        mMonth = monthOfYear;
        mDay = dayOfMonth;
        updateDisplay();
    }
};
```

```
// Create and return DatePickerDialog
@Override
protected Dialog onCreateDialog(int id) {
    switch (id) {
    case DATE_DIALOG_ID:
        return new DatePickerDialog(this, mDateSetListener, mYear, mMonth,
                mDay);
    }
    return null;
}
```

WebView

A ViewGroup that displays a web page

UIWebView



UIWebView

```
<WebView  xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/webview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
/>
```

UIWebView

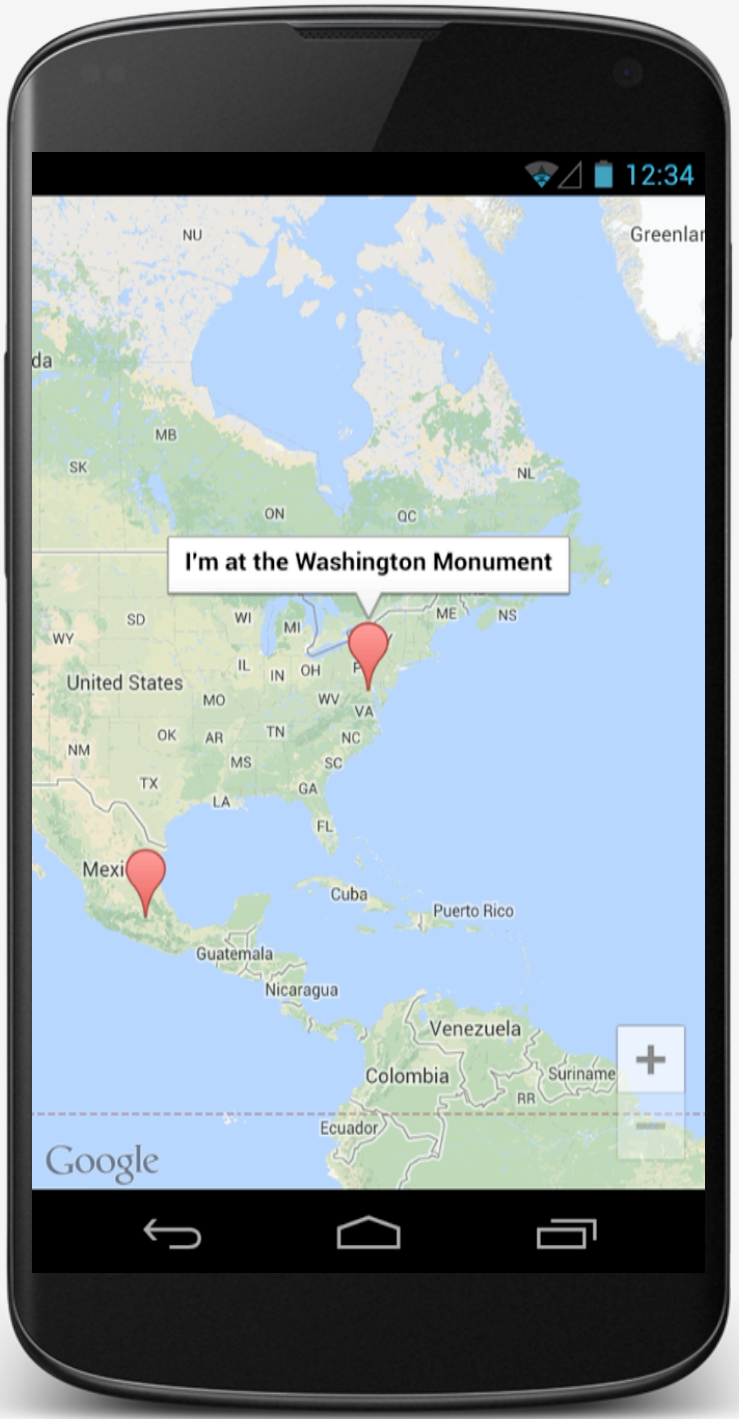
```
mWebView = (WebView) findViewById(R.id.webview);

// Set a kind of listener on the WebView so the WebView can intercept
// URL loading requests if it wants to

mWebView.setWebViewClient(new HelloWebViewClient());

mWebView.getSettings().setJavaScriptEnabled(true);
mWebView.loadUrl("http://www.google.com");
```

```
private class HelloWebViewClient extends WebViewClient {
    private static final String TAG = "HelloWebViewClient";;

    // Give application a chance to catch additional URL loading requests
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url) {
        Log.i(TAG, "About to load:" + url);
        view.loadUrl(url);
        return true;
    }
}
```

MapView

A ViewGroup that displays a Map

UIGoogleMaps



UIGoogleMaps

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
          android:id="@+id/map"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
          android:name="com.google.android.gms.maps.MapFragment"/>
```

```
// The GoogleMap instance underlying the GoogleMapFragment defined in main.xml
mMap = ((MapFragment) getFragmentManager().findFragmentById(R.id.map))
        .getMap();

if (mMap != null) {

    // Set the map position
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(new LatLng(29,
            -88), 0));

    // Add a marker on Washington, DC, USA
    mMap.addMarker(new MarkerOptions().position(
            new LatLng(38.8895, -77.0352)).title(
            getString(R.string.in_washington_string)));

    // Add a marker on Mexico City, Mexico
    mMap.addMarker(new MarkerOptions().position(
            new LatLng(19.13, -99.4)).title(
            getString(R.string.in_mexico_string)));
}
```

Adapters & AdapterViews

AdapterViews are views whose children and data are managed by an Adapter

Adapter manages the data and provides data views to AdapterView

AdapterView displays the data Views

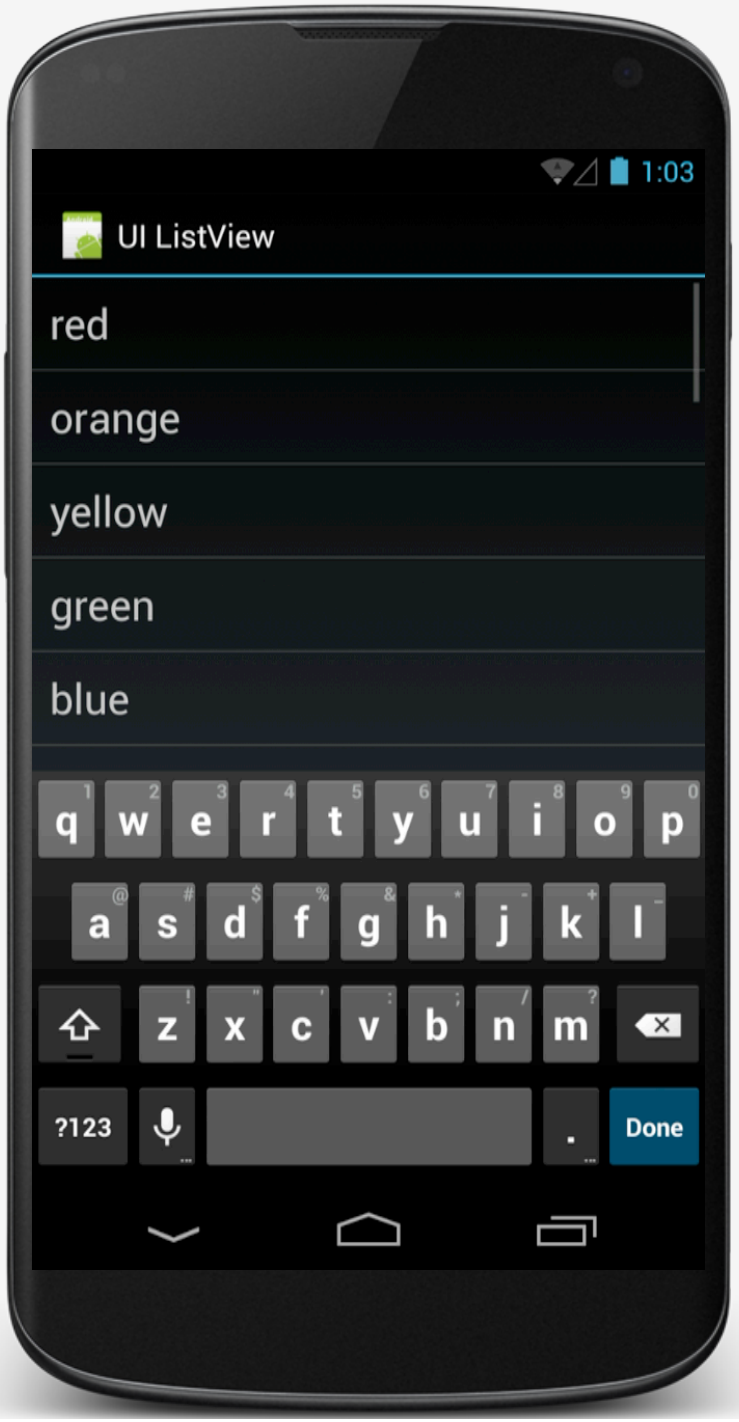
ListView

AdapterView displaying a scrollable list of selectable items

Items managed by a ListAdapter

ListView can filter the list of items based on text input

UIListView



UIListView

```
// Create a new Adapter containing a list of colors
// Set the adapter on this ListActivity's built-in ListView
setListAdapter(new ArrayAdapter<String>(this, R.layout.list_item,
        getResources().getStringArray(R.array.colors)));

ListView lv = getListView();

// Enable filtering when the user types in the virtual keyboard
lv.setTextFilterEnabled(true);

// Set an setOnItemClickListener on the ListView
lv.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {

        // Display a Toast message indicting the selected item
        Toast.makeText(getApplicationContext(),
                ((TextView) view).getText(), Toast.LENGTH_SHORT).show();
    }
});
```

Spinner

An AdapterView that provides a scrollable list of items

User can select one item from the list

Items managed by a SpinnerAdapter

UISpinner


12:26
UI Spinner
Pick a Color. Any Color.
red
red
orange
yellow
green
blue
indigo
violet

UISpinner

```
<Spinner
    android:id="@+id/spinner"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="24sp" />
```

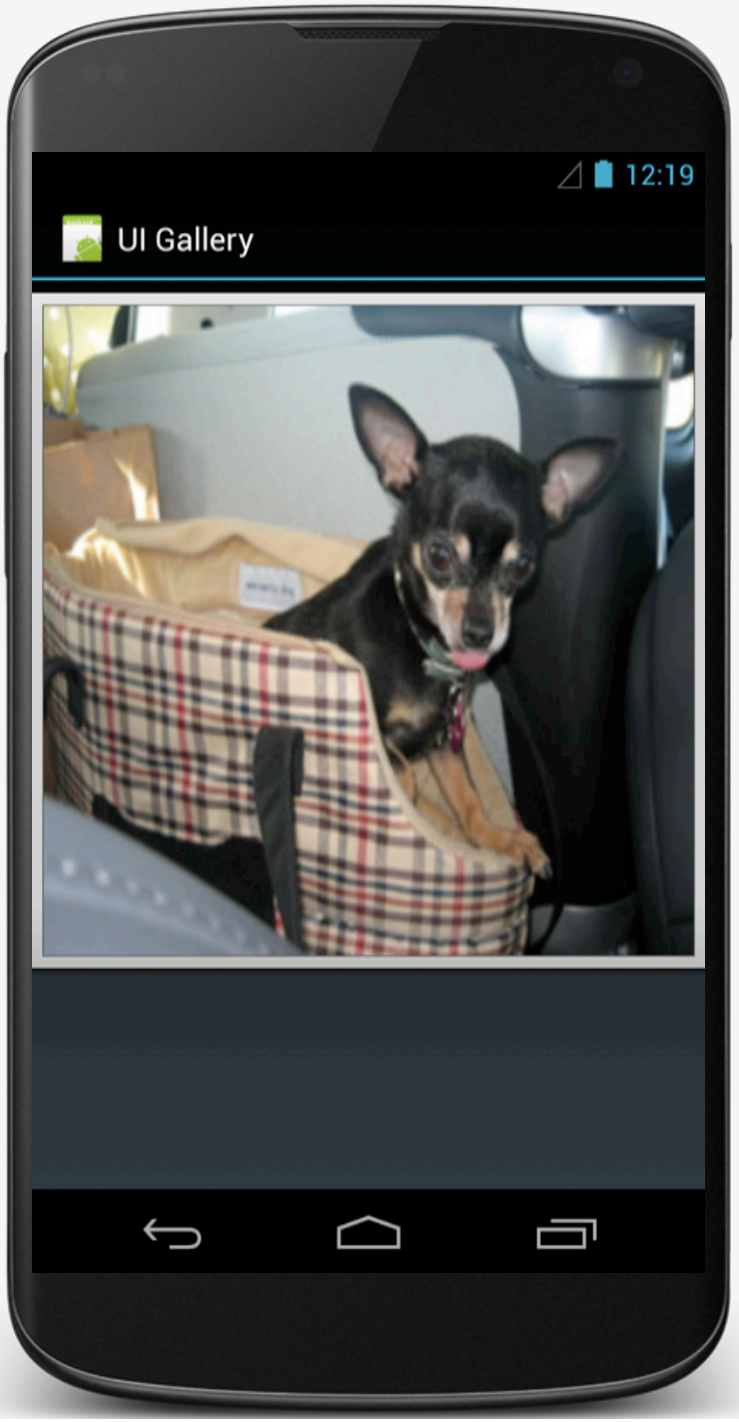
```
// Get a reference to the Spinner
Spinner spinner = (Spinner) findViewById(R.id.spinner);

// Create an Adapter that holds a list of colors
ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
        this, R.array.colors, R.layout.dropdown_item);

// Set the Adapter for the spinner
spinner.setAdapter(adapter);

// Set an setOnItemSelectedListener on the spinner
spinner.setOnItemSelectedListener(new OnItemSelectedListener() {
    public void onItemSelected(AdapterView<?> parent, View view,
            int pos, long id) {
```

Gallery

A ViewGroup showing a horizontally scrolling list

Items managed by a SpinnerAdapter

UIGallery



UIGallery

```
<Gallery xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gallery"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
/>
```

```
Gallery g = (Gallery) findViewById(R.id.gallery);

// Create a new ImageAdapter and set in as the Adapter for the Gallery
g.setAdapter(new ImageAdapter(this));

// Set an setOnItemClickListener on the Gallery
g.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View v,
            int position, long id) {
        // Display a Toast message indicate the selected item
        Toast.makeText(GalleryActivity.this, "" + position,
                Toast.LENGTH_SHORT).show();
    }
});
```

UIGallery

```
// The Adapter class used with the Gallery
public class ImageAdapter extends BaseAdapter {

    private static final int IMAGE_DIM = 800;

    private int mGalleryItemBackground;
    private Context mContext;

    // List of IDs corresponding to the images
    private Integer[] mImageIds = { R.drawable.sample_1,
            R.drawable.sample_2, R.drawable.sample_3, R.drawable.sample_4,
            R.drawable.sample_5, R.drawable.sample_6, R.drawable.sample_7 };

    public ImageAdapter(Context c) {
        mContext = c;
        TypedArray a = obtainStyledAttributes(R.styleable.GalleryActivity);
        mGalleryItemBackground = a.getResourceId(
                R.styleable.GalleryActivity_android_galleryItemBackground,
                0);
        a.recycle();
    }

    public int getCount() {
        return mImageIds.length;
    }

    public Object getItem(int position) {
        return mImageIds[position];
    }

    public long getItemId(int position) {
        return position;
    }
```

UIGallery

```
public View getView(int position, View convertView, ViewGroup parent) {

    ImageView imageView = (ImageView) convertView;

    // If convertView is not recycled set it up now
    if (null == imageView) {
        imageView = new ImageView(mContext);

        imageView.setLayoutParams(new Gallery.LayoutParams(IMAGE_DIM,
                IMAGE_DIM));
        imageView.setScaleType(ImageView.ScaleType.FIT_XY);
        imageView.setBackgroundResource(mGalleryItemBackground);

    }

    // Set the image for the imageView
    imageView.setImageResource(mImageIds[position]);

    return imageView;
}
```

Layouts

A generic ViewGroup that defines a structure for the Views it contains

LinearLayout

Child views arranged in a single horizontal or vertical row

UILinearLayout

UILinearLayout

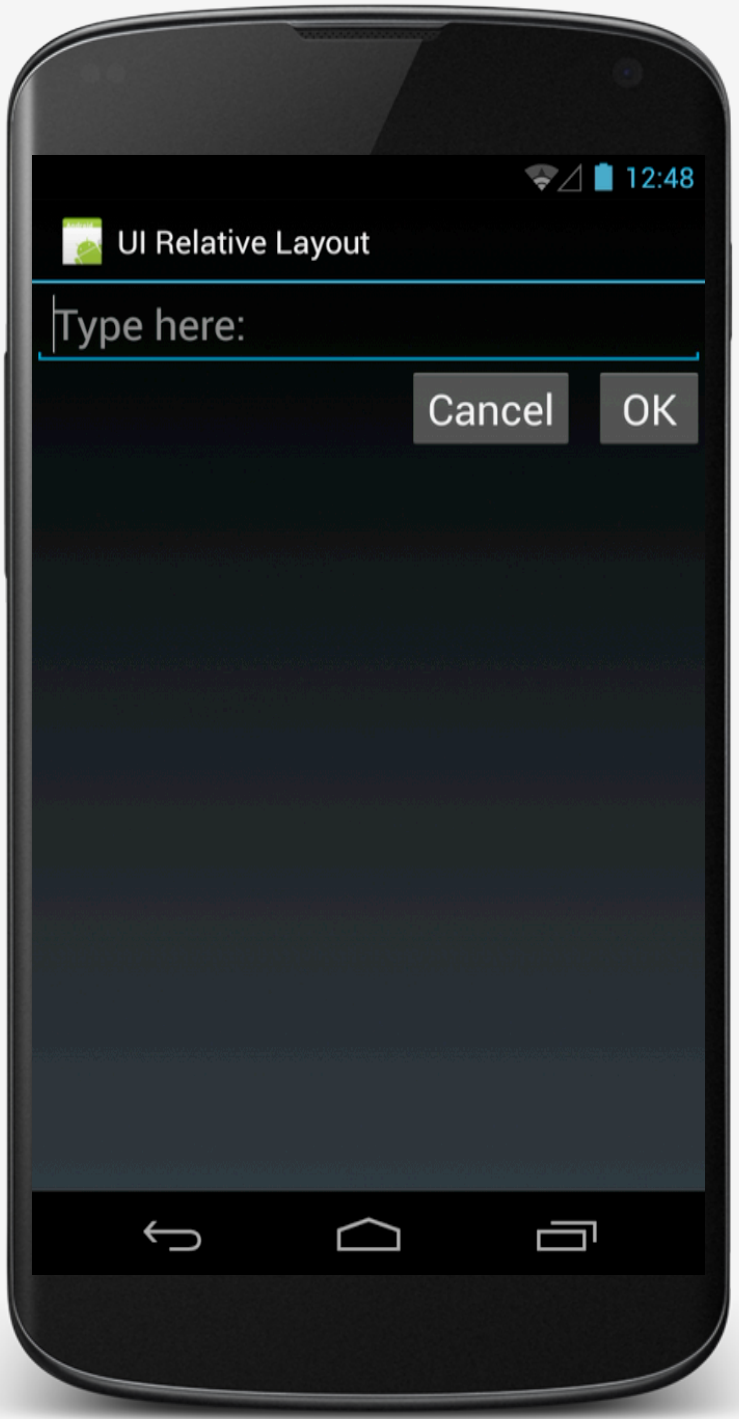
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:orientation="horizontal" >
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="3"
    android:orientation="vertical" >
```

RelativeLayout

Child views are positioned relative to each other and to parent view

UIRelativeLayout



UIRelativeLayout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <!-- Note the use of android:hint to put explanatory text in the EditText  -->
    <EditText
        android:id="@+id/entry"
        android:layout_width="match_parent"
```

TableLayout

Child views arranged into rows & columns

UITableLayout

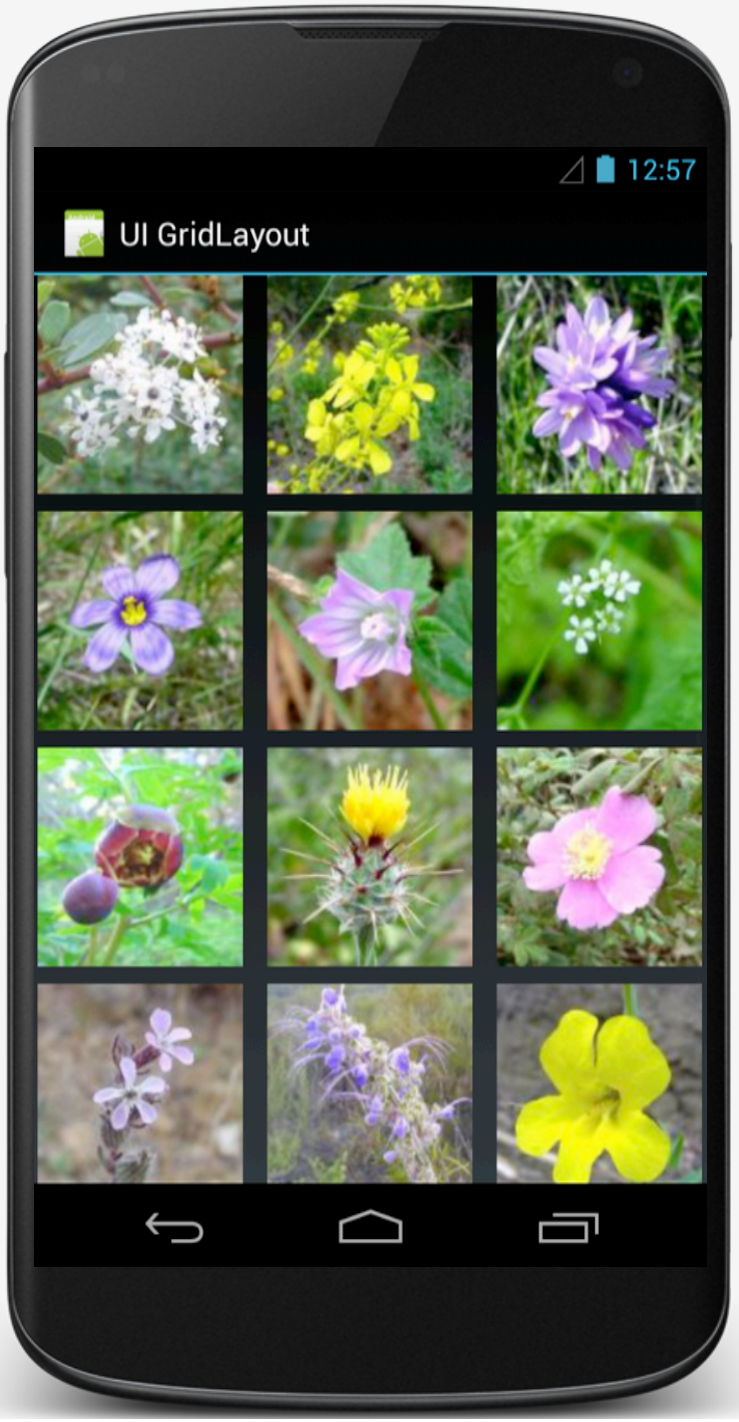
UITableLayout

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="1" >

    <!-- First row -->
    <TableRow>

        <!-- start in column 1 -->
        <TextView
            android:layout_column="1"
            android:padding="3dip"
            android:text="@string/open_string"
            android:textSize="24sp" />

        <TextView
            android:gravity="right"
            android:padding="3dip"
            android:text="@string/ctrl_o_string"
            android:textSize="24sp" />
    </TableRow>
```

GridView

Child views arranged in a two-dimensional, scrollable grid

UIGridView



UIGridView

```
<GridView
    android:id="@+id/gridview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:columnWidth="90dp"
    android:gravity="center"
    android:horizontalSpacing="10dp"
    android:numColumns="auto_fit"
    android:stretchMode="columnWidth"
    android:verticalSpacing="10dp" />
```

```
GridView gridview = (GridView) findViewById(R.id.gridview);

// Create a new ImageAdapter and set it as the Adapter for this GridView
gridview.setAdapter(new ImageAdapter(this, mThumbIdsFlowers));

// Set an setOnItemClickListener on the GridView
gridview.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View v,
            int position, long id) {
```

Menus and ActionBar

- Activities support menus
- Activities can
  - Add items to a menu
  - handle clicks on the menu items

Menu Types

Options

menu shown when user presses the menu button

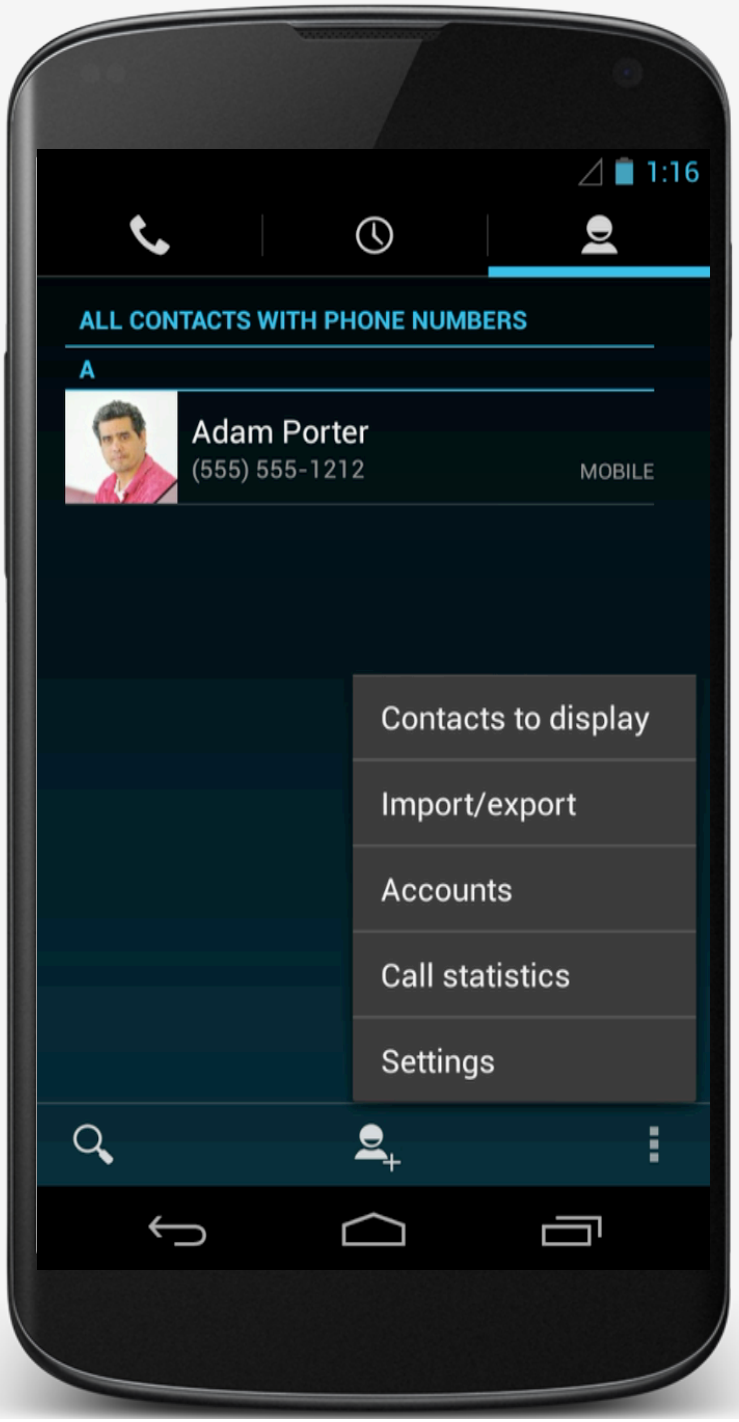
Context

View-specific menu shown when user touches and holds the view

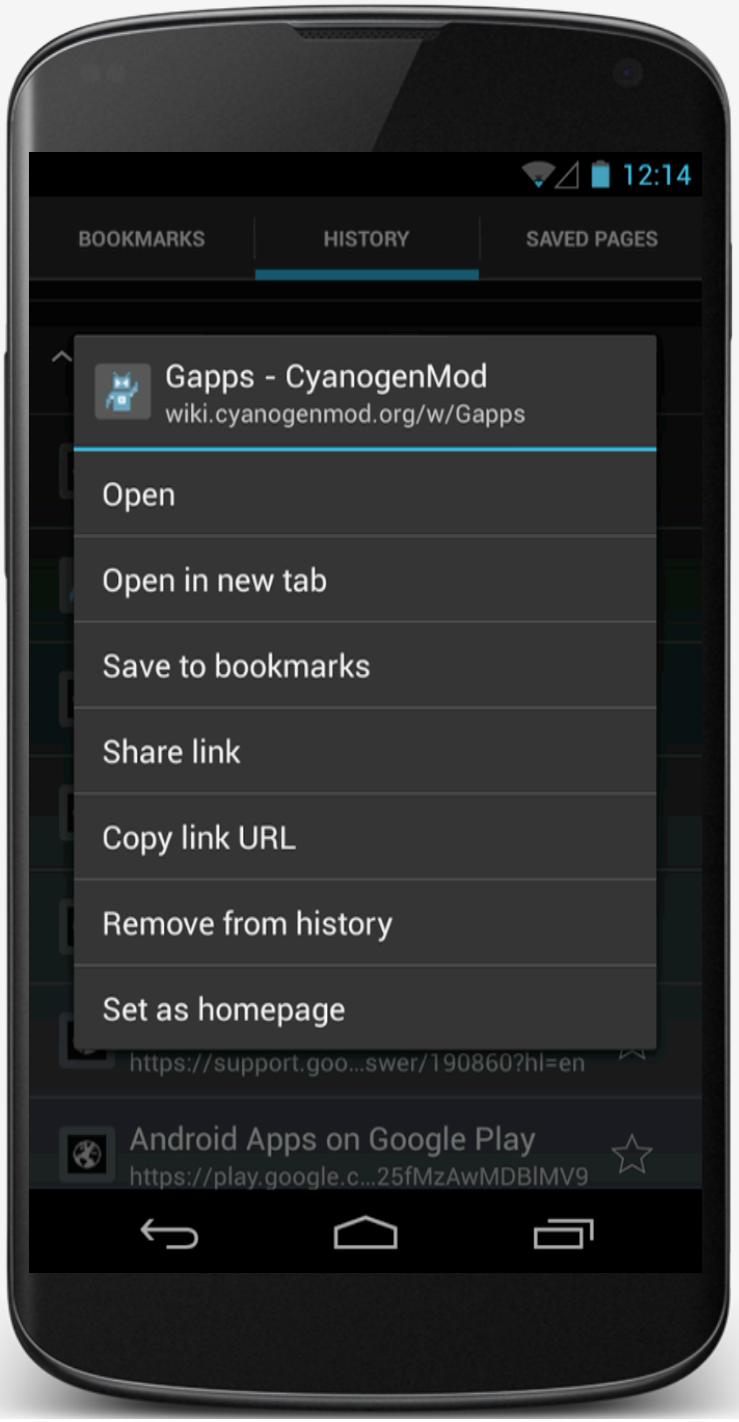
Submenu

A menu activated when user touches a visible menu item

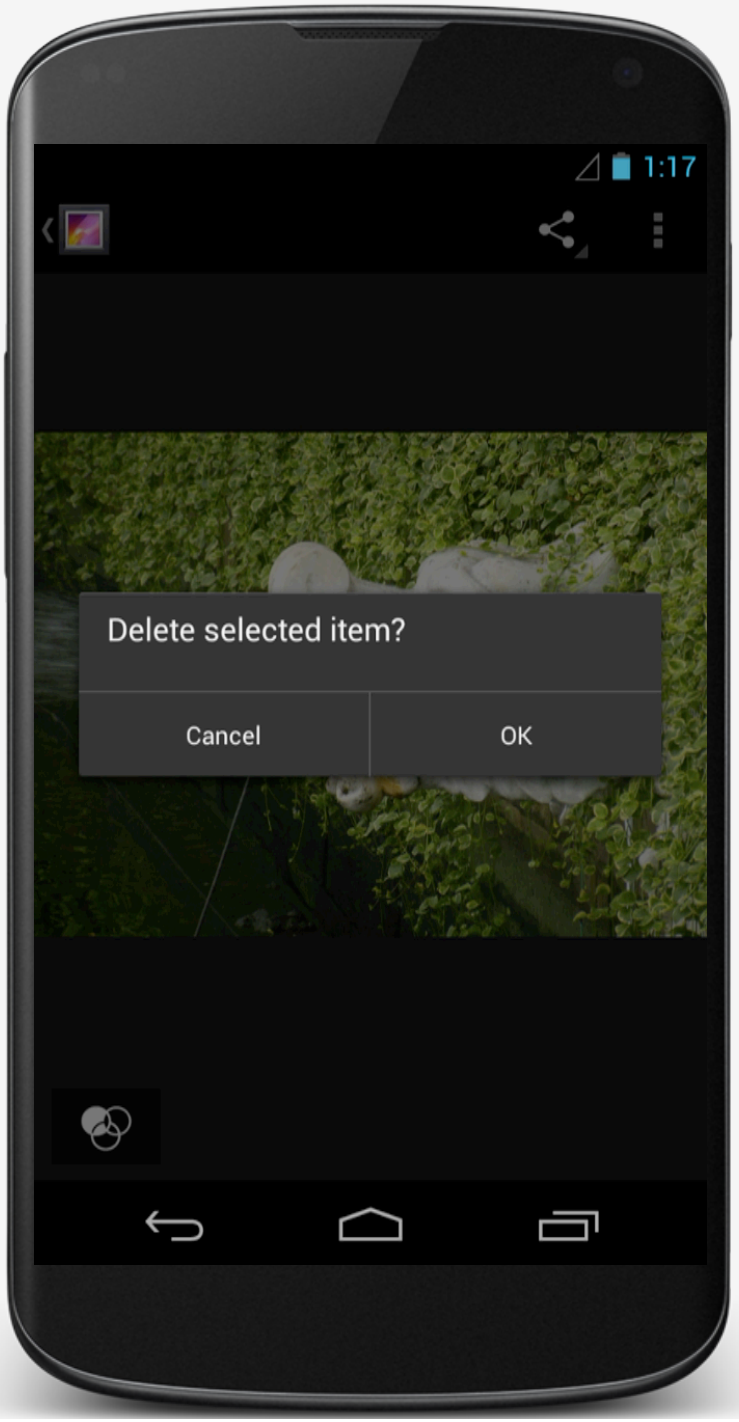
Options Menus



Context Menus



SubMenus



Creating Menus

Define menu resource in XML file

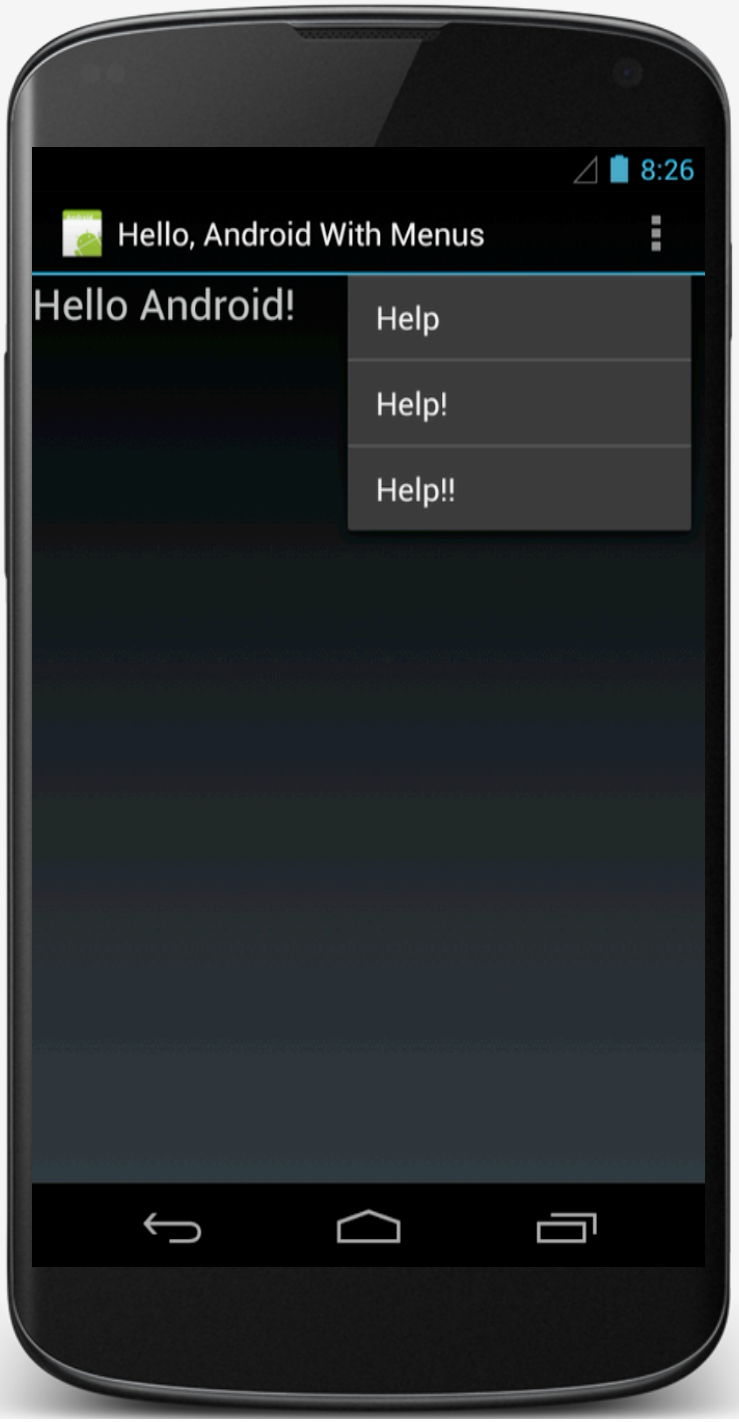
- Store in res/menu/filename.xml

Creating Menus

Inflate menu resource using Menu Inflater in onCreate...Menu() methods

Handling item selection in appropriate on...ItemsSelected() methods

HelloAndroid WithMenus



HelloAndroidWithMenus

```java
// Create Options Menu
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.top_menu, menu);
    return true;
}

// Process clicks on Options Menu items
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.help:
        Toast.makeText(getApplicationContext(), "you've been helped",
                Toast.LENGTH_SHORT).show();
        return true;
    case R.id.more_help:
        Toast.makeText(getApplicationContext(), "you've been helped more",
                Toast.LENGTH_SHORT).show();
        return true;
    case R.id.even_more_help:
        return true;
    default:
        return false;
    }
}
```

HelloAndroidWithMenus

```java
// Create Context Menu
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.context_menu, menu);
}

// Process clicks on Context Menu Items
@Override
public boolean onContextItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.help_guide:
        Toast.makeText(getApplicationContext(), "ContextMenu Shown",
                Toast.LENGTH_SHORT).show();
        return true;
    default:
        return false;
    }
}
```

Menus

Many other features supported

- Grouping menu items
- Binding shortcut keys to menu items
- Binding Intents to menu items

ActionBar

Similar to Application Bar in many desktop applications

Enables quick Access to common operations

FragmentDynamicLayout WithActionBar

Shows play titles and one quote from the selected play

Provides actions for the ActionBar

Three main objects

- QuoteViewerActivity
- TitleFragment
- QuoteFragment

ActionBar.Tab

Screen is divided into tab & content areas

Allows multiple Fragments to share single content area

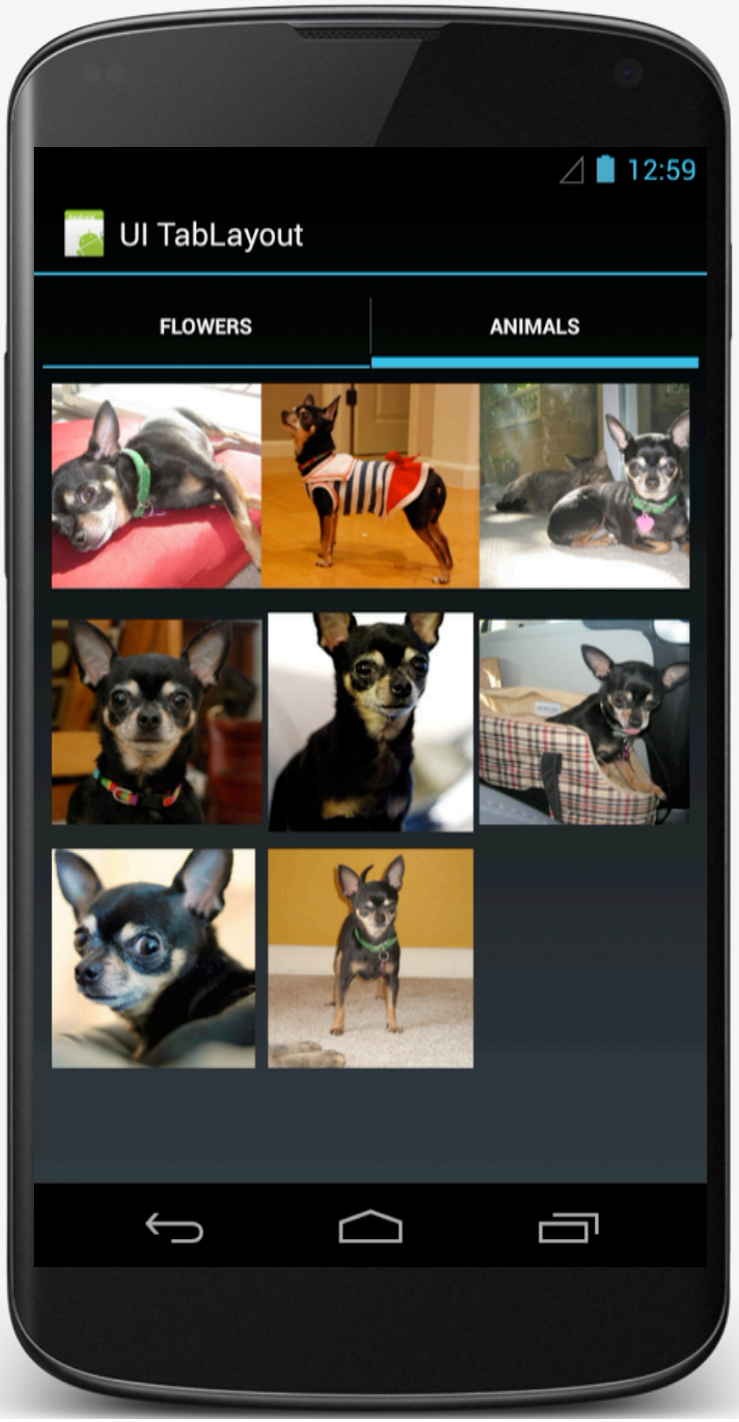
ActionBar.Tab

Each tab is associated with one Fragment

Exactly one tab is selected at any given time

Fragment corresponding to the selected tab is visible in the content area

UITabLayout



UITabLayout

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

    setContentView(R.layout.main);

    // Put ActionBar in Tab Mode
    final ActionBar tabBar = getActionBar();
    tabBar.setNavigationMode(ActionBar.NAVIGATION_MODE_TABS);

    // Create a GridFragment for the Flower thumbnails
    GridFragment flowerFrag = new GridFragment();

    // Store the list of thumbnails as an argument to the GridFragment
    Bundle args = new Bundle();
    args.putIntegerArrayList(THUMBNAIL_IDS, mThumbIdsFlowers);
    flowerFrag.setArguments(args);

    // Configure a tab for the Flower thumbnail GridFragment
    tabBar.addTab(tabBar.newTab().setText(FLOWERS_TABSTRING)
            .setTabListener(new TabListener(flowerFrag)));

    // Create a GridFragment for the Animal thumbnails
    GridFragment animalFrag = new GridFragment();

    // Store the list of thumbnails as an argument to the GridFragment
    args = new Bundle();
    args.putIntegerArrayList(THUMBNAIL_IDS, mThumbIdsAnimals);
    animalFrag.setArguments(args);

    // Configure a tab for the Animal thumbnail GridFragment
    tabBar.addTab(tabBar.newTab().setText(ANIMALS_TABSTRING)
            .setTabListener(new TabListener(animalFrag)));

}
```

UITabLayout

```java
// This class handles user interaction with the tabs
public static class TabListener implements ActionBar.TabListener {
    private static final String TAG = "TabListener";
    private final Fragment mFragment;

    public TabListener(Fragment fragment) {
        mFragment = fragment;
    }

    @Override
    public void onTabReselected(Tab tab, FragmentTransaction ft) {
    }

    // When a tab is selected, change the currently visible Fragment
    @Override
    public void onTabSelected(Tab tab, FragmentTransaction ft) {
        Log.i(TAG, "onTabSelected called");

        if (null != mFragment) {
            ft.replace(R.id.fragment_container, mFragment);
        }
    }

    // When a tab is unselected, remove the currently visible Fragment
    @Override
    public void onTabUnselected(Tab tab, FragmentTransaction ft) {
        Log.i(TAG, "onTabUnselected called");

        if (null != mFragment)
            ft.remove(mFragment);
    }
}
```

Dialogs

Independent subwindows used by Activities to communicate with user

Dialog subclasses

AlertDialog

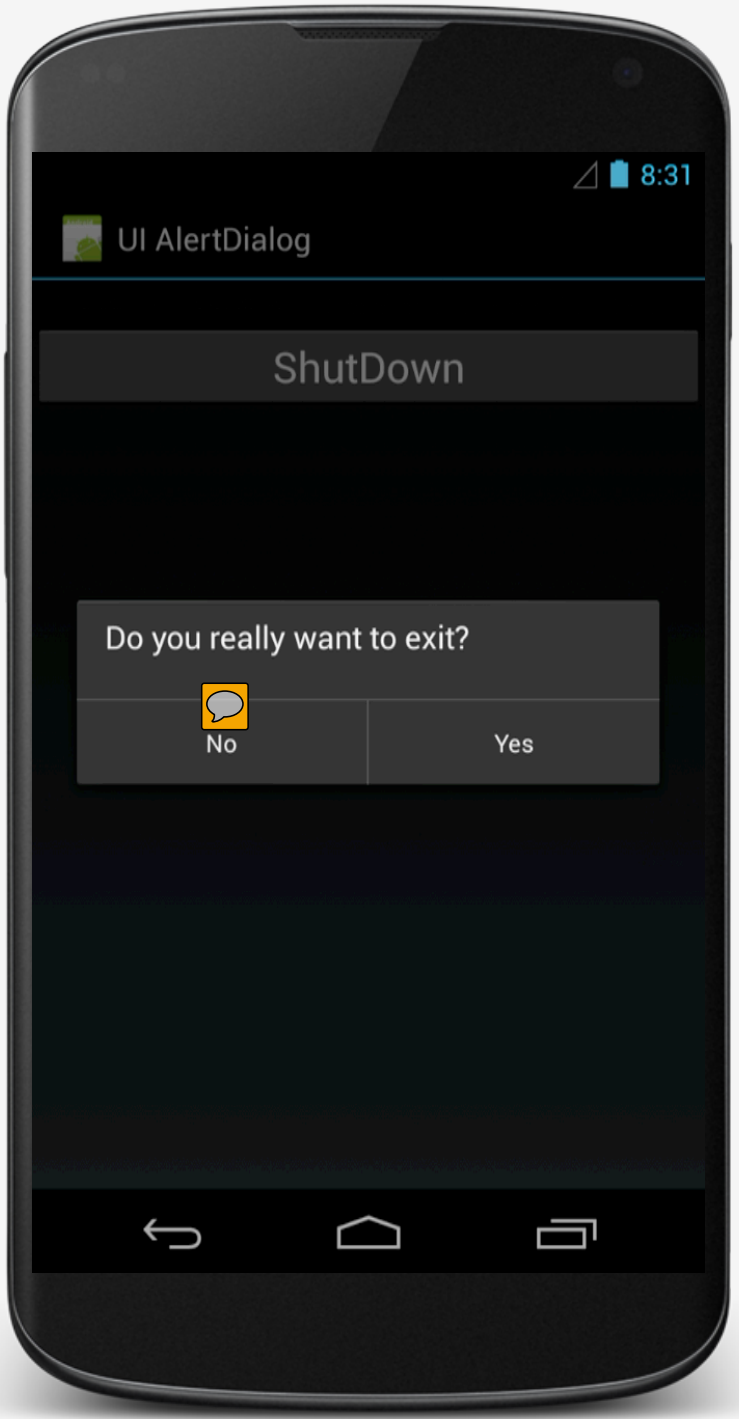
ProgressDialog

DatePickerDialog

TimePickerDialog

UIAlertDialog

ProgressDialog



UIAlertDialog

```java
// Class that creates the AlertDialog
public static class AlertDialogFragment extends DialogFragment {

    public static AlertDialogFragment newInstance() {
        return new AlertDialogFragment();
    }

    // Build AlertDialog using AlertDialog.Builder
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        return new AlertDialog.Builder(getActivity())
                .setMessage("Do you really want to exit?")

                // User cannot dismiss dialog by hitting back button
                .setCancelable(false)

                // Set up No Button
                .setNegativeButton("No",
                        new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog,
                                    int id) {
                                ((AlertDialogActivity) getActivity())
                                        .continueShutdown(false);
                            }
                        })

                // Set up Yes Button
                .setPositiveButton("Yes",
                        new DialogInterface.OnClickListener() {
                            public void onClick(
                                    final DialogInterface dialog, int id) {
                                ((AlertDialogActivity) getActivity())
                                        .continueShutdown(true);
                            }
                        }).create();
    }
}
```

Next Time

User Notifications