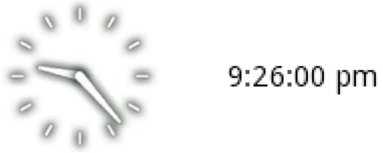
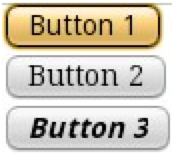
GUI Widgets

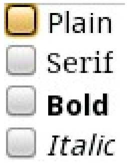
Recall: Android widgets



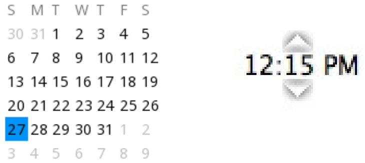
Analog/DigitalClock



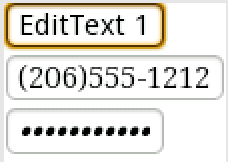
Button



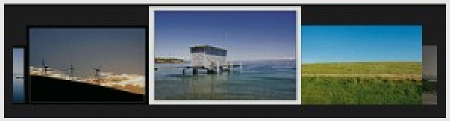
Checkbox



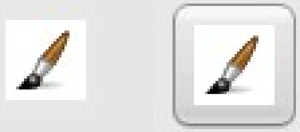
Date/TimePicker



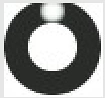
EditText



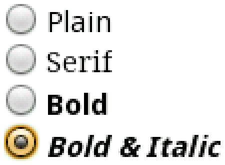
Gallery



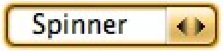
ImageView/Button



ProgressBar

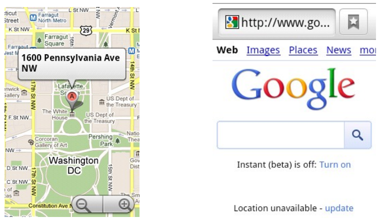


RadioButton



Spinner

TextView


1600 Pennsylvania Ave NW
Washington DC
http://www.go...
Web Images Places News mo
Google
Instant (beta) is off: Turn on
Location unavailable - update

MapView, WebView

Button (link)

A clickable widget with a text label

- key attributes:

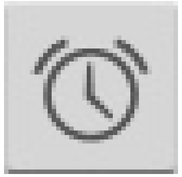
`android:clickable="`***`bool`***`"`	set to false to disable the button
`android:id="@+id/`***`theID`***`"`	unique ID for use in Java code
`android:onClick="`***`function`***`"`	function to call in activity when clicked (must be public, void, and take a View arg)
`android:text="`***`text`***`"`	text to put in the button

- represented by `Button` class in Java code

```
Button b = (Button) findViewById(R.id.theID);
...
```

ImageButton

A clickable widget with an image label



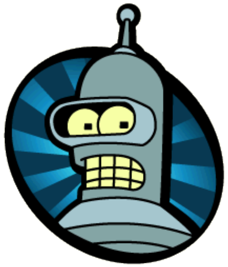
- key attributes:

android:clickable="***bool***"	set to false to disable the button
android:id="@+id/***theID***"	unique ID for use in Java code
android:onClick="***function***"	function to call in activity when clicked (must be public, void, and take a View arg)
android:src="@drawable/***img***"	image to put in the button (must correspond to an image resource)

- to set up an image resource:
  - put image file in project folder **app/src/main/res/drawable**
  - use `@drawable/foo` to refer to `foo.png`
    - use simple file names with only letters and numbers

ImageView

Displays an image without being clickable



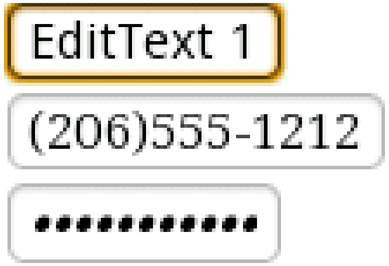
- key attributes:

android:id="@+id/***theID***"	unique ID for use in Java code
android:src="@drawable/***img***"	image to put in the screen (must correspond to an image resource)

- to change the visible image, in Java code:
  - get the ImageView using findViewById
  - call its **setImageResource** method and pass R.drawable.***filename***

EditText (link)

An editable text input box



EditText 1

(206)555-1212

••••••••••••

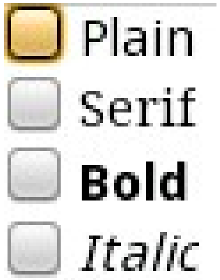
- key attributes:

android:hint="***text***"	gray text to show before user starts to type
android:id="@+id/***theID***"	unique ID for use in Java code
android:inputType="***type***"	what kind of input is being typed; number,phone,date,time,...
android:lines="***int***"	number of visible lines (rows) of input
android:maxLines="***int***"	max lines to allow user to type in the box
android:text="***text***"	initial text to put in box (default empty)
android:textSize="***size***"	size of font to use (e.g. "20dp")

  - others: capitalize, digits, fontFamily, letterSpacing, lineSpacingExtra, minLines, numeric, password, phoneNumber, singleLine, textAllCaps, textColor, typeface

CheckBox (link)

An individual toggleable on/off switch



- key attributes:

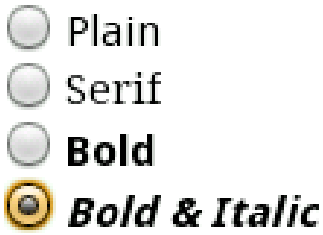
android:checked="***bool***"	set to true to make it initially checked
android:clickable="***bool***"	set to false to disable the checkbox
android:id="@+id/***theID***"	unique ID for use in Java code
android:onClick="***function***"	function to call in activity when clicked (must be public, void, and take a View arg)
android:text="***text***"	text to put next to the checkbox

- In Java code:

```
CheckBox cb = (CheckBox) findViewById(R.id.theID);
cb.toggle();
cb.setChecked(true);
cb.performClick();
```

RadioButton (link)

A toggleable on/off switch; part of a group



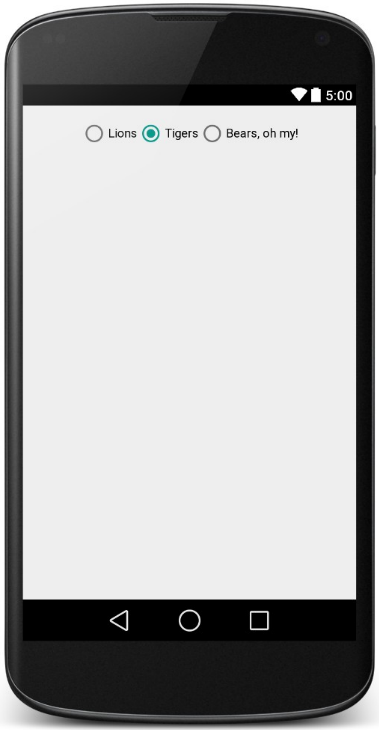
- key attributes:

`android:checked="`***`bool`***`"`	set to true to make it initially checked
`android:clickable="`***`bool`***`"`	set to false to disable the button
`android:id="@+id/`***`theID`***`"`	unique ID for use in Java code
`android:onClick="`***`function`***`"`	function to call in activity when clicked (must be public, void, and take a View arg)
`android:text="`***`text`***`"`	text to put next to the button

- need to be nested inside a `RadioGroup` tag in XML so that only one can be selected at a time

RadioGroup example

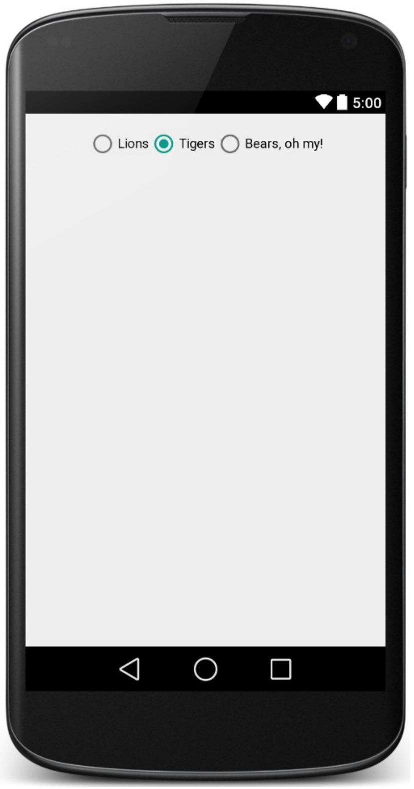
```
<LinearLayout ...
        android:orientation="vertical"
        android:gravity="center|top">
    <RadioGroup ...
            android:orientation="horizontal">
        <RadioButton ... android:id="@+id/lions"
                         android:text="Lions"
                         android:onClick="radioClick" />
        <RadioButton ... android:id="@+id/tigers"
                         android:text="Tigers"
                         android:checked="true"
                         android:onClick="radioClick" />
        <RadioButton ... android:id="@+id/bears"
                         android:text="Bears, oh my!"
                         android:onClick="radioClick" />
    </RadioGroup>
</LinearLayout>
```



○ Lions ● Tigers ○ Bears, oh my!

Reusing onClick handler

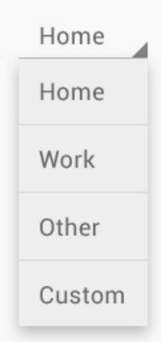
```
// in MainActivity.java
public class MainActivity extends Activity {

    public void radioClick(View view) {
        // check which radio button was clicked
        if (view.getId() == R.id.lions) {
            // ...
        } else if (view.getId() == R.id.tigers) {
            // ...
        } else {
            // bears ...
        }
    }
```



○ Lions ● Tigers ○ Bears, oh my!

Spinner (link)

A drop-down menu of selectable choices



- key attributes:

android:clickable="***bool***"	set to false to disable the spinner
android:id="@+id/***theID***"	unique ID for use in Java code
android:entries="@array/***array***"	set of options to appear in spinner (must match an array in strings.xml)
android:prompt="@string/***text***"	title text when dialog of choices pops up

- also need to handle events in Java code (see later)
  - must get the Spinner object using findViewById
  - then call its setOnItemSelectedListener method (see example)

String resources

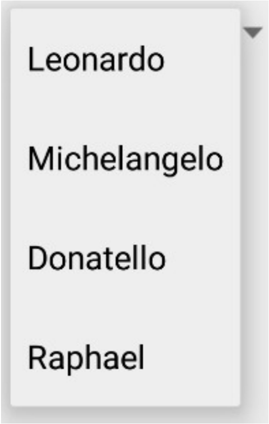
- Declare constant strings and arrays in `res/values/strings.xml`:

```
<resources>
    <string name="name">value</string>
    <string name="name">value</string>

    <string-array name="arrayname">
        <item>value</item>
        <item>value</item>
        <item>value</item>   <!-- must escape ' as \' in values -->
        ...
        <item>value</item>
    </string-array>
</resources>
```

- Refer to them in Java code:
  - as a resource: `R.string.`***`name`***`,` `R.array.`***`name`***
  - as a string or array: `getResources().getString(R.string.`***`name`***`),`
    `getResources().getStringArray(R.array.`***`name`***`)`

Spinner example

```
<LinearLayout ...>
    <Spinner ... android:id="@+id/tmnt"
        android:entries="@array/turtles"
        android:prompt="@string/choose_turtle" />
    <TextView ... android:id="@+id/result" />
</LinearLayout>
```

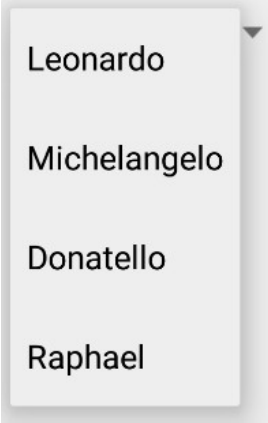


- in res/values/strings.xml:

```
    <resources>
        <string name="choose_turtle">Choose a turtle:</string>
        <string-array name="turtles">
            <item>Leonardo</item>
            <item>Michelangelo</item>
            <item>Donatello</item>
            <item>Raphael</item>
        </string-array>
    </resources>
```

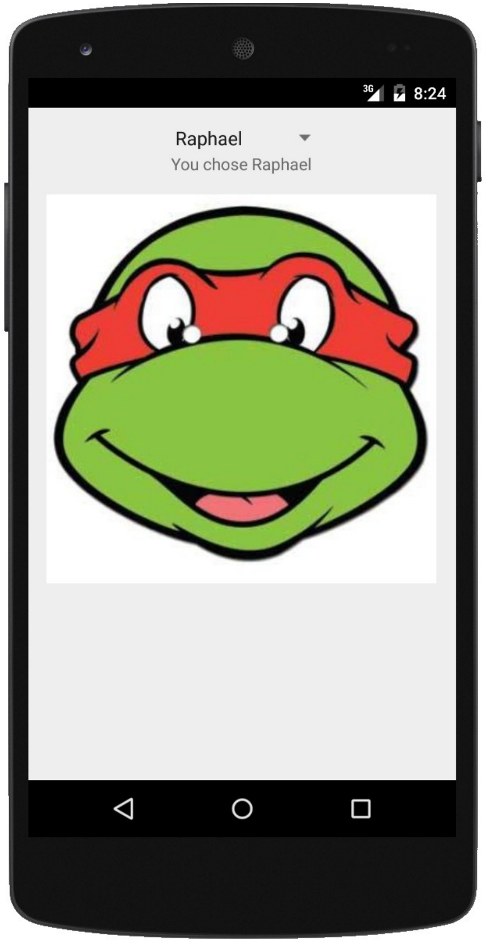
Spinner event example

```java
// in MainActivity.java
public class MainActivity extends Activity {
  ...
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Spinner spin = (Spinner) findViewById(R.id.tmnt);
    spin.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
      public void onItemSelected(AdapterView<?> spin, View v, int i, long id) {
        TextView result = (TextView) findViewById(R.id.turtle_result);
        result.setText("You chose " + spin.getSelectedItem());
      }

      public void onNothingSelected(AdapterView<?> parent) {}  // empty
    });
  }
}
```



TMNT app exercise

- Write an app to select TMNT characters from a spinner.
  - When a character is selected, an image about that character and other information is presented to the user.
  - Assume that relevant image files are already available for each character.

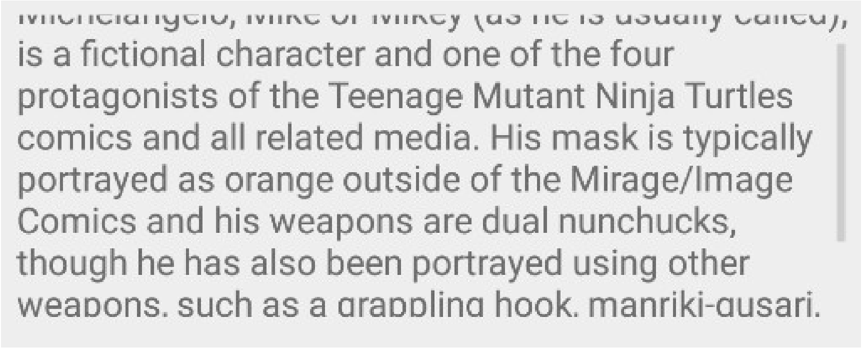


Raphael

You chose Raphael

ScrollView

A container with scrollbars around another widget or container



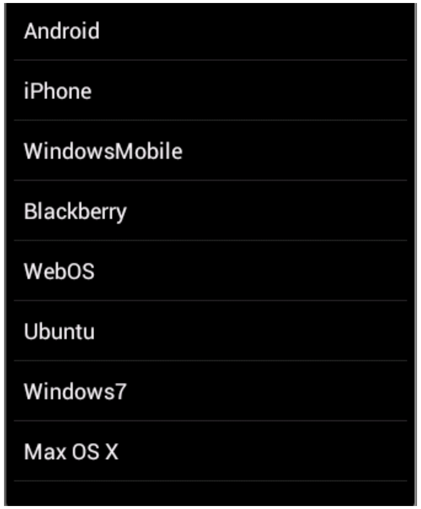
Michelangelo, Mike or Mikey (as he is usually called), is a fictional character and one of the four protagonists of the Teenage Mutant Ninja Turtles comics and all related media. His mask is typically portrayed as orange outside of the Mirage/Image Comics and his weapons are dual nunchucks, though he has also been portrayed using other weapons, such as a grappling hook, manriki-gusari,

```
<LinearLayout ...>
    ...
    <ScrollView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
        <TextView ... android:id="@+id/turtle_info" />
    </ScrollView>
</LinearLayout>
```

List (link)

A visible menu of selectable choices

- lists are more complicated, so we'll cover them later ...


Android
iPhone
WindowsMobile
Blackberry
WebOS
Ubuntu
Windows7
Max OS X