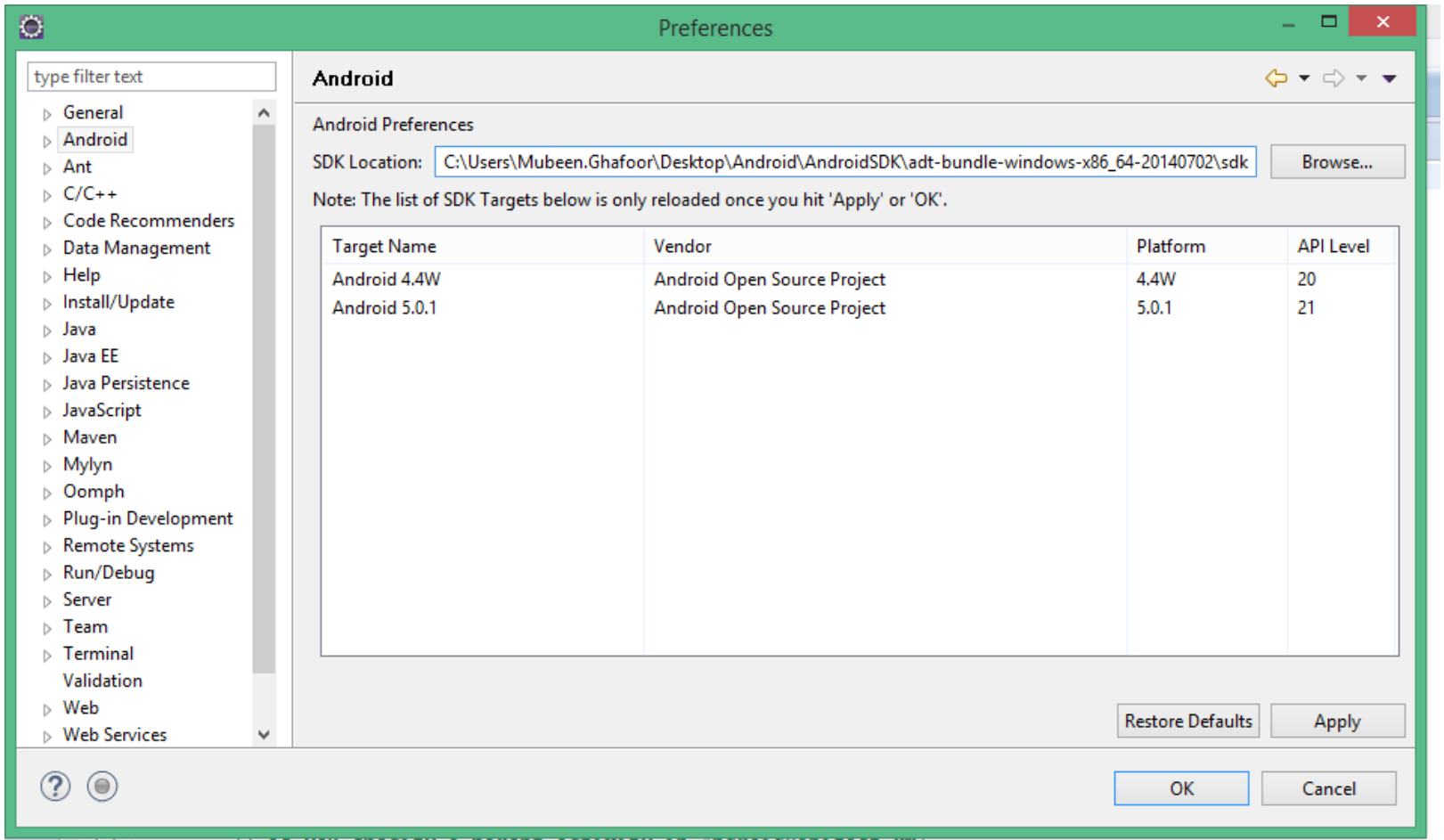
Application Development for Mobile Devices

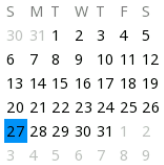
Lecture 6

Eclipse Settings

Windows -→ Preferences →
Set SDK Location:
Browse → Android → AndroidSDK → adt-bundle-windows-x86_64-20140702 →sdk



Android widgets

 9:26:00 pm Analog/DigitalClock	 Button	 Checkbox	  Date/TimePicker
EditText	 Gallery	 ImageView/Button	 ProgressBar
 RadioButton	 Spinner	 TextView	  MapView, WebView

View Class

View class represents the basic building block for user interface (UI) components.

Every item in a user interface (UI) is a subclass of the Android View class.

Typical examples include standard items such as the Button, CheckBox, ProgressBar and TextView classes.

Such views are also referred to as widgets.

Views may have an integer id associated with them. These ids are typically assigned in the layout XML files.

Define a Button in the layout file and assign it a unique ID.

```
<Button
    android:id="@+id/my_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/my_button_text"/>
```

Button

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);
...
```

findViewById

`findViewById(int):`
This function is used to retrieve the widgets in the UI that you need to interact with programmatically.

- findViewById(int id) is a method of the View and Activity classes.
- This method will take a resource Id usually in the form of R.id.mView and will return to you a View object that is a reference to that View.
- The returned object needs to be type casted to the correct type of View before you can start interacting with it.

- Example:

```
TextView answerLabel = (TextView) findViewById(R.id.textView1);

Button getAnswerButton = (Button) findViewById(R.id.button1);
```

OnClickListener

OnClickListener
Interface definition for a callback to be invoked when a view is clicked.

onClick(View v):
Called when a view has been clicked.

setOnClickListener():
Register a callback to be invoked when this view is clicked.

OnClickListener for Button

```
public class MainActivity extends ActionBarActivity implements OnClickListener {
Button myButton1 = null;
TextView myTextView = null;

@Override
protected void onCreate(Bundle savedInstanceState) {
.
.
        myTextView = (TextView) findViewById(R.id.textView1);
        myButton1 = (Button) findViewById(R.id.button1);
        myButton1.setOnClickListener(this);
}

@Override
public void onClick(View v) {
        if(v.getId() == myButton1.getId()){
                myTextView.setText("Button 1 Clicked");
        }
}
```

ImageButton

A clickable widget with an image label

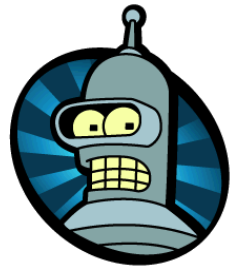
- key attributes:

`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
    - i.e. ImageView image = (ImageView) findViewById(R.id.imageView1);
  - call its **setImageResource** method and pass R.drawable.***filename***
    - i.e. image.setImageResource(R.drawable.Img1);

EditText

An editable text input box

- 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

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();          //Change the checked state of the view to the
                      //inverse of its current state
cb.performClick();    // Call this view's OnClickListener, if it is defined
cb.setChecked(true) ;    // Changes the checked state of this button
```