# CS260 Intro to Java & Android
# 06.AndroidEvents

## Winter 2014

# Input Controls

- Android has a wide variety of input controls for designing sophisticated UIs including

  - Buttons

  - Text Fields

  - Checkboxes

  - Radio Buttons

# Button

- Consists of text or icon (or both)



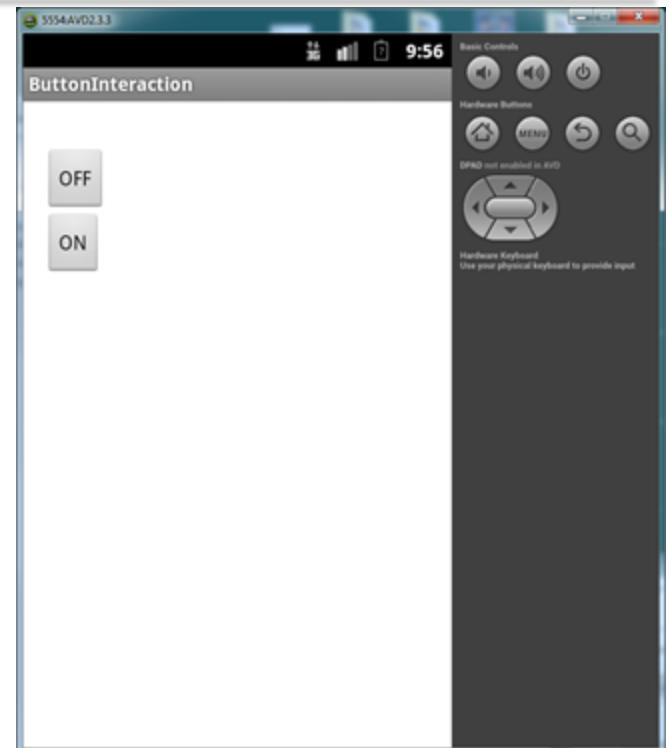- Communicates an action when the user touches the button

# Event-handling

- Events are created through user interaction

- Events are captured from a View object interacted with by the user

Example: When a button is touched, the method onTouchEvent () is called on the touched object

# Button Example



- Create a project called ButtonInteraction that looks exactly like the following

- Button names in main.xml are btnOff and btnOn

- Strings are sButtonOff is OFF and sButtonOn is ON

# Button Events

void android.view.View.setOnClickListener(OnClickListener l)

**public void setOnClickListener** (*View.OnClickListener l*)
Added in API level 1

Register a callback to be invoked when this view is clicked. If this view is not clickable, it becomes clickable.

**Parameters**
l The callback that will run
**See Also**
- setClickable (boolean)

- Method #1 for handling a button click

```
mButtonOff = (Button) findViewById (R.id.btnOff);
mButtonOff.setOnClickListener (new View.OnClickListener ()
{
   public void onClick (View view)
   {
      Log.d ("ButtonInteraction","Button Off");
   }
});
```

# What might a Button look like?

```
class Button
{
  private View.OnClickListener mListener;

  public Button ()
  {
    mListener = null;
  }

  public void setOnClickListener (View.OnClickListener listener)
  {
    mListener = listener;
  }

  private void handleEvent (Event e)
  {
    paintButton();
    if( mListener != null)
    {
      mListener.onClick (this);
    } …
```

# Button Events

- Method#2 for handling a button click

```java
public class ButtonExampleActivity extends Activity
                                    implements View.OnClickListener

mButtonOff = (Button) findViewById (R.id.btnOff);
mButtonOff.setOnClickListener (this);


public void onClick (View view)
{
  Log.d ("ButtonInteraction", "Button Select");
}
```
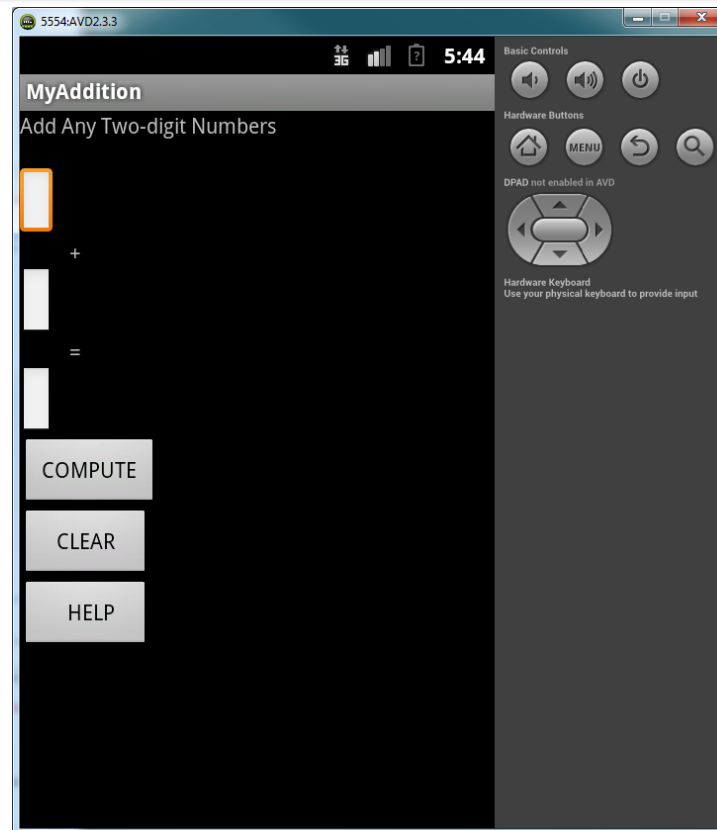
# Button Events

```java
public void onClick (View view)
{
    Log.d ("ButtonInteraction", "Button Select");

    if (mButtonSelect == view)
    {
        // do something else
    }
}
```

# Problem

- You are to design a simple calculator that does addition of two digit numbers. The calculator is displayed on the next slide and details are given on slides thereafter.

# Calculator

# Class Calculator

- Has private members
  1. EditText mEditNumber1
  2. EditText mEditNumber2
  3. EditText mEditSum
  4. Button mButtonCompute
  5. mButtonClear
  6. mButtonHelp

# main.xml ids

- main.xml has ids
  1. btnClear
  2. btnCompute
  3. btnHelp
  4. editNumber1
  5. editNumber2
  6. editSum

# Step to Complete Calculator

1. Create all private instance variables

2. Set each instance variable equal to its' associated widget

3. Button widgets need to set the appropriate onClickListener

4. Add functionality to the onClick method such that when the Clear button is pressed, all text in each EditText field is cleared

e.g. mEditNumber1.setText ("");

# Step to Complete Calculator

1. Program the Compute button such that you will add the two numbers entered by the user and output the result in mEditSum

```java
int num1, num2;

try
{
  num1 = Integer.parseInt (mEditNumber1.getText ().toString());
}
catch (NumberFormatException e)
{
  // we will eventually pop up an alert dialog
  num1 = 0;
}
```

# Step to Complete Calculator

1. Create all private instance variables

2. Set each instance variable equal to its associated widget

3. Button widgets need to set the appropriate onClickListener

4. Add functionality to the onClick method such that when the Clear button is pressed, all text in each EditText field is cleared

e.g. mEditNumber1.setText ("");

# Challenge

- If you get this far with time to spare, try and figure out how to display an alert if the user enters Invalid Input