# CSC 143 Java

Event-Driven Programming

## Event-Driven Programming (Review)

- Idea: program initializes itself then accepts *events* in whatever random order they occur
- Kinds of events
  - Mouse move/drag/click, Keyboard, Touch screen, Joystick, game controller
  - Window resized or components changed
  - Activity over network or file stream
  - Sensors, lab experiments
  - Timer interrupt
- First demonstrated in the 1960s(!);
- Major developments at Xerox PARC in the 1970s (Alto workstation, Smalltalk, Xerox Star)
- Appeared outside research community in Apple Macintosh (1984)

## Event Objects

- An event is represented in Java by an object
  - AWT/Swing events are subclasses of AWTEvent. Examples:
    - **ActionEvent – button pressed**
    - **KeyEvent – keyboard input**
    - **MouseEvent – mouse move/drag/click/button press or release**
- Event objects contain information about the event
  - User interface object that triggered the event (JButton, JPanel)
  - Other information appropriate for the event. Examples:
    - **ActionEvent – text string describing button (if from a button)**
    - **MouseEvent – mouse coordinates of the event**
- Package is java.awt.event

## Event Handler (or Listener)

Event Handler: An object that is interested in reacting to an event.

✓An event handler must implement the appropriate *interface* for the events it wishes to respond to:

ActionListener, KeyListener, MouseListener (buttons), MouseMotionListener (move/drag), others …

✓An event handler must *register* with the object that generates the event (e.g. user interface component).

- An event handler may register to respond to many kinds of events generated by many different objects
- There may be many event handlers registered to listen for particular kinds of events from a single object

## Reacting to Events in Java

When an event occurs, all registered event handlers are notified by sending them an appropriate message

(Just like the model/viewer architecture)

➢The appropriate method from the interface is called

actionPerformed, keyPressed, keyReleased, keyTyped, mouseClicked, mouseDragged, etc.

➢ An event object describing the event is passed in as an argument

## A First Example – Simple Button Handler

Idea: React to button clicks

- Create a handler object to receive clicks on the button and print a message when events happen
- Register the handler object with the button

## Button Listener

- Simplest part of setup
- Need to implement ActionListener interface and actionPerformed method declared in that interface
- Doesn't do much – just printing

```
public class ButtonListener implements ActionListener {
    /** Respond to events generated by the button. */
    public void actionPerformed(ActionEvent e) {
        System.out.println("A button was clicked\n" + e);
    }
}
```

## Hook Things Up

- Instantiate the button and a event handler
- Register the event handler with the button
- Add the button to a container

```
JButton button = new JButton("Hit me!");
 button.addActionListener(new ButtonListener());
// add it to the JFrame or a JPanel within the JFrame
```

Let's take a look at this little program.