Graphical User Interface Programming

Robert M. Dondero, Ph.D.
Princeton University
Objectives

You will learn about:

- Graphical user interface (GUI) programming
- Specifically...
- For Java: The AWT/Swing GUI library
- For Python: The Tkinter GUI library
Objectives

- For each:
  - Interface structure
  - Program structure
  - Components
    - Dialog Boxes
  - Layout managers
  - Event handling
- First Java/AWT, then Python/Tkinter
Running GUI Apps

- Option 1: Run on your computer
  - Requirement:
    - Install Java or Python on your computer
  - Caveats:
    - Make sure you install proper version
    - Must test Asgt 3 on penguins
Running GUI Apps

- Option 2: Run on penguins
  - Requirement:
    - Run **X Window System Server** on local computer
    - That is, on your computer or lab computer
  - Instructions available through "Topics" Web page
Part 1:
Java AWT and Swing
Abstract Window Toolkit (AWT)

- Java 1.0
- Delegates creation/behavior of each GUI element to native GUI toolkit on each target platform
- Difficult to write highly-portable library
- Applications constrained to greatest common denominator approach
AWT/Swing Brief History

- **Swing**
  - Java 1.2
  - Relies on underlying windowing system only to display and paint windows
  - Paints components (buttons, menus, etc.) itself onto blank windows
  - Easy to write portable library
  - Application/user can choose look-and-feel
  - Built on top of AWT architecture
AWT/Swing Window Structure

JFrame

JPanel

JPanel
See `FrameTest.java`

- Create class to implement Runnable
- Create `run()` method within class
- Create `JFrame` object within `run()` method

`main()` method:

- Creates new Runnable object
- Passes to `EventQueue.invokeLater()`

`EventQueue.invokeLater()` method starts event loop
AWT/Swing Pgm Structure

- Generalizing
  - Suggestion: accept as a pattern
  - More details in "Threads" lecture
See `ComponentTest.java`

- JFrame, JPanel
- JButton, JLabel, JTextField, JTextArea, JScrollPane, JScrollBar, JSlider, JCheckBox
- ButtonGroup, JRadioButton, Border
- JComboBox
- JMenuBar, JMenu, JMenuItem, JPopupMenu
- JToolBar
- Tool tips
Swing Dialog Components

- See `OptionPaneMessageTest.java`
- See `OptionPaneConfirmTest.java`
- See `OptionPaneOptionTest.java`
- See `OptionPaneInputTest.java`
  - `JOptionPane`
- See `FileChooserTest.java`
  - `JFileChooser`
- Generalizing...
Swing Components

java.lang.Object
  java.awt.Component
    java.awt.Container
      java.awt.Window
        java.awt.Frame
          javax.swing.JFrame
        java.awt.Dialog
          javax.swing.JDialog
          javax.swing.JComponent
            javax.swing.JPanel
              javax.swing.JTextField
              javax.swing.JTextArea
              javax.swing.JLabel
              javax.swing.JScrollPane
              javax.swing.JScrollBar
              javax.swing.JSlider
              javax.swing.JComboBox
              java.awt.AbstractButton
                javax.swing.JButton
                javax.swing.JToggleButton
                  javax.swing.JRadioButton
                javax.swing.JMenuItem
                  javax.swing.JMenu
                    javax.swing.JMenuBar
                      javax.swing.JOptionPane
                      javax.swing.JFileChooser
AWT Layout Managers

- See `FlowLayoutTest.java`
  - `FlowLayout`
  - Flow is maintained as window resizes
- See `GridLayoutTest.java`
  - `GridLayout`
  - Components arranged in 2-D grid
AWT Layout Managers

- See `BorderLayoutTest.java`
  - `BorderLayout`
  - **North** and **south** border: sizes are fixed vertically, expand horizontally
  - **East** and **west** border: sizes are fixed horizontally, expand vertically
  - **Center** expands both horizontally and vertically
AWT Layout Managers

- And others; example...
- GridBagLayout
  - "The mother of all layout managers" – Horstmann and Cornell
  - Beyond COS 333 scope
  - (Unnecessary for assignments)
- Generalizing...
AWT Layout Managers

java.lang.Object
   java.awt.FlowLayout
   java.awt.BorderLayout
   java.awt.GridLayout
   java.awt.GridBagLayout
   java.awt.BoxLayout
   java.awt.CardLayout
   java.awt GroupLayout
   java.awt GroupLayout
   java.awt.OverlayLayout
   java.awt.ScrollPaneLayout
   java.awt.SpringLayout
   java.awt.ViewportLayout
AWT Event Handling

- Show `EventTest1.java`
  - JButton event handling
  - Event handling for other components is similar
  - Listener object must access panel
- Generalizing...
Problem:

- Listener object often must access Components other than the one generating the event
- Awkward to pass Components to Listener constructor

Solution:

- Make the Runnable object the Listener...
AWT Event Handling

- See EventTest2.java
  - EventTestRunnable2 implements ActionListener
  - EventTestRunnable2 class is large; serves multiple purposes
Problem:
- Previous approach can yield large complex classes

Solution:
- Inner classes...
See **EventTest3.java**

- ColorActionListener3 is inner class
  - Defined within EventTestRunnable3
  - ColorActionListener3 object can access fields of EventTestRunnable3 objects

---

**Editorial:**

- Are inner classes worth the added complexity?

---

**Generalizing...**
## AWT Event Handling

### Semantic events:

<table>
<thead>
<tr>
<th>Component</th>
<th>Listener Interface</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbstractButton</td>
<td>ActionListener</td>
<td>ActionEvent</td>
</tr>
<tr>
<td>JComboBox</td>
<td>actionPerformed()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td>JPasswordField</td>
<td></td>
<td>getActionCommand(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getModifiers()</td>
</tr>
<tr>
<td>JTextField</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ActionEvent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>getSource(), getID(),</td>
<td></td>
</tr>
<tr>
<td></td>
<td>getActionCommand(),</td>
<td></td>
</tr>
<tr>
<td></td>
<td>getModifiers()</td>
<td></td>
</tr>
<tr>
<td>JScrollBar</td>
<td>AdjustmentListener</td>
<td>AdjustmentEvent</td>
</tr>
<tr>
<td></td>
<td>adjustmentValueChanged()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getAdjustable(), getValue(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getAdjustmentType()</td>
</tr>
<tr>
<td>JSlider</td>
<td>ChangeListener</td>
<td>ChangeEvent</td>
</tr>
<tr>
<td></td>
<td>stateChanged()</td>
<td>getSource(), getID()</td>
</tr>
<tr>
<td>AbstractButton</td>
<td>ItemListener</td>
<td>ItemEvent</td>
</tr>
<tr>
<td>JComboBox</td>
<td>itemStateChanged()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getItem(), getStateChange(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getItemSelectable()</td>
</tr>
</tbody>
</table>
## AWT Event Handling

### Low-level events:

<table>
<thead>
<tr>
<th>Component</th>
<th>Listener Interface</th>
<th>Its Methods</th>
<th>Parameter</th>
<th>Its Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>FocusListener</td>
<td>focusGained()</td>
<td>FocusEvent</td>
<td>getID(), getComponent(), isTemporary()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>focusLost()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>KeyListener</td>
<td>keyPressed()</td>
<td>KeyEvent</td>
<td>getID(), getComponent(), getWhen(), getModifiers(), isAltDown(), isControlDown(), isMetaDown(), isShiftDown(), getKeyChar(), keyCode(), getKeyModifiersText(), getKeyText(), isActionKey()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>keyReleased()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>keyTyped()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>MouseListener</td>
<td>mousePressed()</td>
<td>MouseEvent</td>
<td>getID(), getWhen(), getModifiers(), isAltDown(), isControlDown(), isMetaDown(), isShiftDown(), getClickCount(), getX(), getY(), getPoint(), translatePoint(), isPopupTrigger()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mouseReleased()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mouseEntered()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mouseExited()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mouseClicked()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>MouseMotionListener</td>
<td>mouseDragged()</td>
<td>MouseEvent</td>
<td>getID()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mouseMoved()</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## AWT Event Handling

### Low-level events:

<table>
<thead>
<tr>
<th>Component</th>
<th>Listener Interface</th>
<th>Parameter Its Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>MouseWheelListener</td>
<td>MouseWheelEvent</td>
</tr>
<tr>
<td></td>
<td>mouseWheelMoved()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getWheelRotation(), getScrollAmount()</td>
</tr>
<tr>
<td>Window</td>
<td>WindowListener</td>
<td>WindowEvent</td>
</tr>
<tr>
<td></td>
<td>windowClosing()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td>windowOpening()</td>
<td>getComponent(), getWindow()</td>
</tr>
<tr>
<td></td>
<td>windowOpen()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>windowIconified()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>windowDeiconified()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>windowClosed()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>windowActivated()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>windowDeactivated()</td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>WindowFocusListener</td>
<td>WindowEvent</td>
</tr>
<tr>
<td></td>
<td>windowGainedFocus()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td>windowLostFocus()</td>
<td>getComponent(), getWindow()</td>
</tr>
<tr>
<td>Window</td>
<td>WindowStateListener</td>
<td>WindowEvent</td>
</tr>
<tr>
<td></td>
<td>windowStateChanged()</td>
<td>getSource(), getID(),</td>
</tr>
<tr>
<td></td>
<td></td>
<td>getOldState(), getNewState()</td>
</tr>
</tbody>
</table>
AWT Event Handling

- See EventTestAll.java
  - AWT informs JComponent and Window objects of **many** events
An AWT/Swing Example

- See ColorDisplayer.java
  - Slightly larger and more realistic
  - Multiple interacting components
Part 2: Python Tkinter
Tkinter Brief History

- **Tcl**
  - Very successful scripting language
  - John Ousterhout (Stanford U.)
Tkinter Brief History

- **Tk**
  - Cross-platform GUI toolkit
  - Developed for Tcl
  - Ported to other scripting languages
    - Ruby, Perl, Python
Tkinter Brief History

- **Tkinter**
  - Python interface to TK ("Tk interface")
  - Python wrapper around Tk
  - The "standard" Python GUI library
    - Distributed with Python
- Other Python GUI options are available
  - http://wiki.python.org/moin/GuiProgramming
Tkinter Window Structure
See `frametest.py`

- TK constructor creates a Toplevel object
- `tk.mainloop()` method starts event handling loop

Generalizing

- Suggestion: accept as a pattern
Tkinter Components

- See `componenttest.py`
  - Label
  - Button, Checkbutton, Radiobutton
  - Entry
  - Listbox
  - Scale, Scrollbar
  - Menu
  - Text, ScrolledText
- Generalizing...
Tkinter Components

object
  BaseWidget (and Pack, Grid, Place)
  Widget
    Label
    Button
    Checkbutton
    Radiobutton
    Entry
    Listbox
    Scale
    Scrollbar
    Text
      ScrolledText
    Menu
    Frame
  BaseWidget (and Wm)
  Toplevel

No combo box
Tkinter Components

- Constructor arguments:
  - First argument is parent Frame
  - Subsequent arguments set options
- Each widget is a mapping
  - After construction, use `widget['option']` to get or set an option
Tkinter-Related Dialogs

- See messageboxtest.py
  - tkMessageBox module
- See simplesdialogtest.py
  - tkSimpleDialog module
- See filedialogtest.py
  - tkFileDialog module
- See colorchoosertest.py
  - tkColorChooser module
Tkinter-Related Dialogs

Generalizing:

**tkMessageBox** module
- askokcancel()
- askquestion()
- askretrycancel()
- askyesno()
- showerror()
- showinfo()
- showwarning()

**tkSimpleDialog**
- askfloat()
- askinteger()
- askstring()

**tkFileDialog** module
- askdirectory()
- askopenfilename()
- askopenfilenames()
- asksaveasfilename()

**tkColorChooser** module
- askcolor()
See packertest.py

- Pack class, pack() method
- Widget inherits from Pack
  - Can send pack() message to any Widget object to "pack" it within its parent Widget
Tkinter Layout Managers

- Same functionality as AWT BorderLayout...
- **North** and **south** border: sizes are fixed vertically, expand horizontally
- **East** and **west** border: sizes are fixed horizontally, expand vertically
- **Center** expands both horizontally and vertically
Tkinter Layout Managers

- See `griddertest.py`
  - Grid class, `grid()` method
  - Widget inherits from Grid
    - Can send `grid()` message to any Widget object to "grid" it within its parent object
  - Same functionality as AWT GridLayout...
  - Components arranged in 2-D grid
- Generalizing...
Tkinter Layout Managers

- And one more...
- Place
  - Low-level layout manager
  - Places widgets within parent exactly as the widget requires
  - Used to implement custom layout managers
  - Beyond the scope of COS 333
Tkinter Layout Managers

object
  Pack
  Grid
  Place
Tkinter Event Handling

- See `eventtest.py`
  - Inner functions
  - Similar to Java's inner classes
See `eventtesteach.py`

- Common patterns, per Widget

Generalizing...

- Three non-disjoint mechanisms
"Command" Mechanism

- For Button, Radiobutton, Checkbutton, Menu
- Define zero-arg callback function
- Assign callback function to Widget "command" property
- Similar to Java Swing "semantic" event handling
Tkinter Event Handling

- "Variable Object" Mechanism
  - For Checkbutton, Radiobutton, Entry, Scale
  - Define **Tkinter variable object**
    - IntVar, DoubleVar, StringVar
  - Assign Tkinter variable object to Widget variable property
  - User affects Widget => affects variable
  - Program affects variable => affects Widget
  - No Java Swing equivalent
"Bind" Mechanism

- For any Widget
- Define callback function with event as arg
- Send bind() message to Widget object with event pattern and callback function as args
- Similar to Java Swing "low-level" event handling
Tkinter Event Patterns

First argument to `bind()` specifies an event pattern

'<'MODIFIER-MODIFIER-TYPE-DETAIL>'

MODIFIER:
   Control, Alt, Double, Triple,
   Button1, B1, Button2, B2, Button3, B3, Button4, B4, Button5, B5
   M, Mod1, M1, Mod2, M2, Mod3, M3, Mod4, M4, Mod5, M5,
   Shift, Lock, Meta,

TYPE:
   KeyPress, Key, KeyRelease,
   ButtonPress, Button, ButtonRelease,
   Motion,
   Enter, Leave,
   FocusIn, FocusOut,
   Destroy,
   MouseWheel, Activate, Map, Expose, Circulate, Property,
   Colormap, Gravity, Reparent, Configure, Unmap, Deactivate,
   Visibility

DETAIL:
   For KeyPress, Key, KeyRelease: the Keysym
   For ButtonPress, Button, ButtonRelease: the button number
**Tkinter Event Patterns**

**Keysym**
- a, b, ...
- 0, 1, ...
- F1, F2, ...
- Left, Right, Up, Down, End, Home, Insert, Print, Tab, Escape, Return, Caps_Lock, Num_Lock, Scroll_Lock
- space, less

**Button Number**
- 1 (the left mouse button)
- 2 (the middle mouse button, or both mouse buttons)
- 3 (the right mouse button)
Tkinter Event Patterns

- `widget.bind('<Button>', f)`
  - Call `f()` when user clicks a mouse button
- `widget.bind('<Key>', f)`
  - Call `f()` when user types a key
- `widget.bind('<Double-Button-1>', f)`
  - Call `f()` when user double clicks the left mouse button
- `widget.bind('<Control-Key-a>', f)`
  - Call `f()` when user types Ctrl-a
Tkinter Event Handling

- See `eventtestall.py`
  - Tkinter informs component objects of many events
A Tkinter Example

- See `colordisplay.py`
  - Slightly larger and more realistic
  - Multiple interacting components
Getting More Information

tux:~$ python
>>> from Tkinter import *
>>> from ScrolledText import *
>>> help(Toplevel)
>>> help(Widget)
>>> help(Button)
>>> help(ScrolledText)
>>> help(Event)
...

GUI Editorial

- Java's AWT/Swing library is well-documented
- Python's Tkinter library is less so
- Avoid Tkinter for GUI applications?
- Avoid Python for GUI applications?
Summary

- We have covered:
  - Graphical user interface (GUI) programming
  - Specifically...
  - For Java: The AWT/Swing GUI library
  - For Python: The Tkinter GUI library
Summary

- For each:
  - Interface structure
  - Program structure
  - Components
    - Dialog Boxes
  - Layout managers
  - Event handling
Summary

- Not covered...
- Low-level drawing
  - For Java: the Graphics2D class and the 2D library
  - For Python: the Canvas class
Appendix 1:
Graphical User Interfaces in C
GUls in C

- For C/Unix/Linux....
- The X-Window GUI library
  - X-Window server runs on local computer
  - X-Window client runs on server computer
- See
  http://www.paulgriffiths.net/program/c/srsc/helloxsrc.html
  - Use -lX11 option to build
  - Explicit coding of event loop
Higher level GUI modules exist
Especially...
GTK+
  See http://www.gtk.org/
Appendix 2: Bad GUls
Bad GUIs

- A few collected by Dr. Kernighan...
Bad GUIs

Updates were unable to be successfully installed
You must restart your computer for the updates to take effect.
Bad GUIs

Acrobat Reader 5.0 is not currently configured to be the default application for PDF files.

Would you like to make PDF files open with Acrobat Reader 5.0 instead of Acrobat Reader 5.0?

☑ Do not show this message again

No Yes
Bad GUIs

Step 2: Inspector

You must answer all required questions before you submit your recommendation.

→ indicates a required question.

<table>
<thead>
<tr>
<th>Question</th>
<th>Original Answer</th>
<th>New Answer</th>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>TO BE COMPLETED BY THE RECOMMENDER:</td>
<td>(empty)</td>
<td>→ Go to Page 2</td>
<td>Required Item</td>
</tr>
</tbody>
</table>
Bad GUIs

<table>
<thead>
<tr>
<th>Institution</th>
<th>Princeton University</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department</td>
<td>Computer Science</td>
</tr>
<tr>
<td>Additional email</td>
<td></td>
</tr>
</tbody>
</table>

Please use comma "," to separate your additional email addresses

Office Address

<table>
<thead>
<tr>
<th>Address line1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Address line2</td>
<td></td>
</tr>
<tr>
<td>City</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Country</td>
<td></td>
</tr>
<tr>
<td>Zip</td>
<td></td>
</tr>
<tr>
<td>Office phone</td>
<td></td>
</tr>
</tbody>
</table>

If the above information is incorrect, please update it by correcting the information in the fields and clicking the "Update Information" button below. We ask for this information in case future correspondence is necessary. We appreciate your willingness to help in this important process.

The following error(s) occurred:
- undefined is required.
- undefined is required.
- undefined is required.

[Update Information] [Reset]
You have been redirected to this page for the following reason:

Error Copying File or Folder

Cannot copy 2 O Lord, thou hast searched me out: The specified network name is no longer available.

OK
Bad GUIs

Princeton University
Office of the Registrar

Thank you for taking time to complete the Classroom Survey!

Year & Term: temp
Instructor: temp

© 2006 The Trustees of Princeton University. Last modified 12/05/06
Email the Office of the Registrar: regisn@princeton.edu
Bad GUIs
Bad GUIs

- See also...
- The "Interface Hall of Shame"