Programming Languages and Techniques (CIS120)

Lecture 32

November 18, 2015

Histogram Demo / Swing

Announcements

- HW8: Spellchecker
  - Available on the web site
  - Due: Tuesday, November 24th
  - Parsing, working with I/O, more practice with collections

- Next Week: No Lab Sections

- Next Wednesday: Bonus Lecture "Consequences of Code as Data"
  - Attendance not required (but encouraged if you are around!)

Poll

How many hours did you spend on Chat HW?

1. 0-4
2. 5-9
3. 10-14
4. 15-19
5. 20-24
6. 25+

Design Example: Histogram.java

A design exercise using java.io and
the generic collection libraries

Problem Statement

Write a program that, given a filename for a text file as input, calculates the frequencies (i.e. number of occurrences) of each distinct word of the file. The program should then print the frequency distribution to the console as a sequence of “word: freq” pairs (one per line).

Histogram result:

```
The : 1
Write : 1
a : 4
as : 2
calculates : 1
command : 1
console : 1
distinct : 1
distribution : 1
e : 1
each : 1
file : 2
filename : 1
for : 1
freq : 1
frequencies : 1
frequency : 1
given : 1
i : 1
input : 1
line : 2
number : 1
occurrences : 1
of : 4
one : 1
pairs : 1
per : 1
print : 1
program : 2
sequence : 1
should : 1
text : 1
that : 1
the : 4
then : 1
to : 1
word : 2
```

Interactive Demo

Histogram.java and WordScanner.java

Java's Swing Libraries

GUIs, take two

Why study GUIs (yet again)?

- Most common example of *event based programming*
- Heavy and effective use of OO inheritance
- Case study in library organization
  - (and advanced Java features)
- Ideas applicable everywhere:
  - Web apps
  - Mobile apps
  - Desktop apps
- Fun!

Terminology overview

	GUI (OCaml)	Swing
Graphics Context	Gctx.gctx	Graphics
Widget type	Widget.widget	JComponent
Basic Widgets	button label checkbox	JButton JLabel JCheckBox
Container Widgets	hpair, vpair	JPanel, Layouts
Events	event	ActionEvent MouseEvent KeyEvent
Event Listener	mouse_listener mouseclick_listener (any function of type event -> unit)	ActionListener MouseListener KeyListener

Swing practicalities

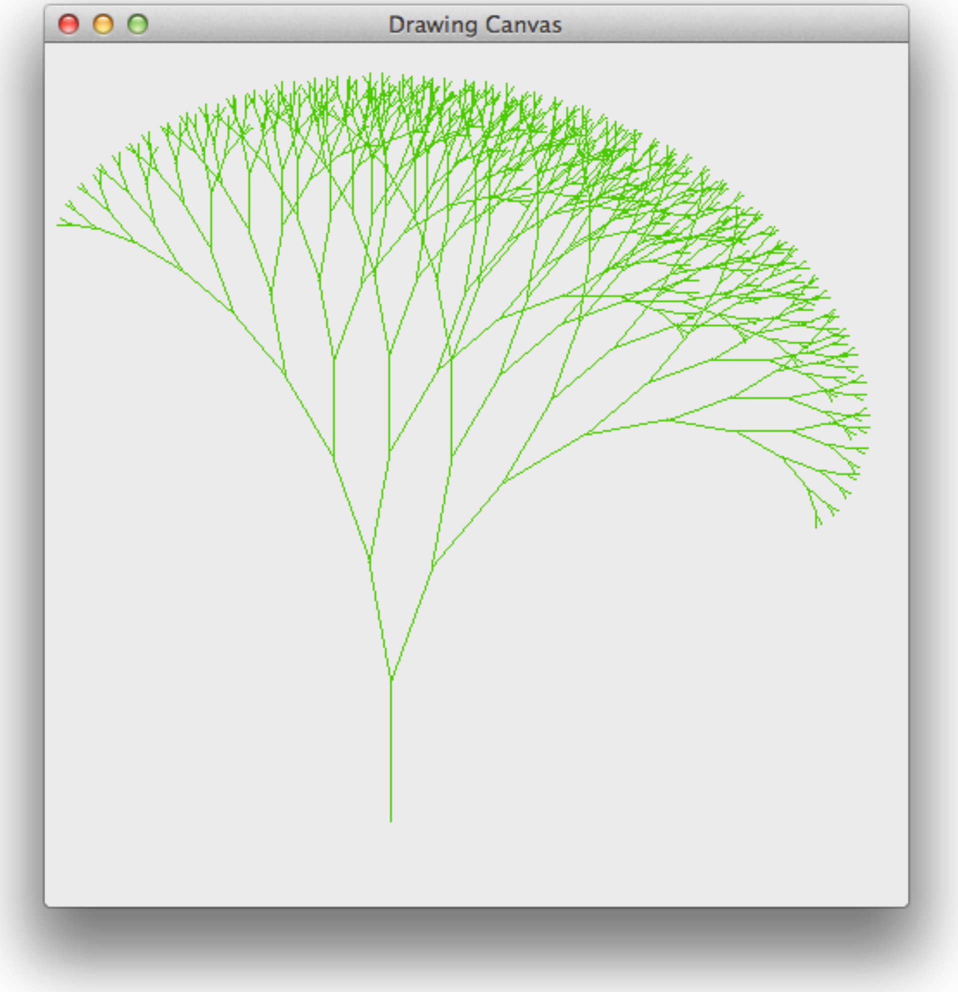
- Java library for GUI development
  - javax.swing.*
- Built on existing library: AWT
  - java.awt.*
  - If there are two versions of something, use Swing’s. (e.g., java.awt.Button vs. javax.swing.JButton)
    - The “Jxxx” version is usually the one you want, rather than “xxx”.
- Portable
  - Communicates with OS's native window system
  - Same Java program looks different when run on PC, Linux and Mac

Simple Drawing

DrawingCanvas.java

DrawingCanvasMain.java

Fractal Drawing Demo



How do we draw a picture?

- In GUI HW, create a widget where the repaint function uses the graphics context to draw an image

```
let w_draw : widget =
{
   repaint = (fun (gc:gctx) ->
       Gctx.draw_line gc (0, 0) (100, 100);
       Gctx.draw_point gc (3,4)) ;

   size    = (fun () -> (200,200));

   handle  = (fun () -> ())
}
```

- In Swing, *extend* from class JComponent …

Fundamental class: JComponent

- Analogue to widget type from GUI project
  - (*Terminology*: widget == component)
- Subclasses *override* methods
  - paintComponent (like repaint, displays the component)
  - getPreferredSize (like size, calculates the size of the component)
  - Events handled by listeners
- Much more functionality available
  - minimum/maximum size
  - font
  - foreground/background color
  - borders
  - what is visible
  - many more…

Simple Drawing Component

```java
public class DrawingCanvas extends JComponent {

    public void paintComponent(Graphics gc) {
        super.paintComponent(gc);

        // set the pen color to green
        gc.setColor(Color.GREEN);

        // draw a fractal tree
        fractal (gc, 75, 100, 270, 15);
    }

    // get the size of the drawing panel
    public Dimension getPreferredSize() {
        return new Dimension(150,150);
    }
```

How to display this component?

JFrame

- Represents a top-level window
  - Displayed directly by OS (looks different on Mac, PC, etc.)
- Contains JComponents
- Can be moved, resized, iconified, closed

```
public void run() {
    JFrame frame = new JFrame("Tree");

    // set the content of the window to be the drawing
    frame.getContentPane().add(new DrawingCanvas());

    // make sure the application exits when the frame closes
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // resize the frame based on the size of the panel
    frame.pack();

    // show the frame
    frame.setVisible(true);
}
```

Poll

Are you here today?

1. yes