Graphics & Applets

CSC 1051 – Data Structures and Algorithms I

Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
www.csc.villanova.edu/~map/1051/
Today

- Image representation
- Java Graphics class
- Java applets
What’s a picture?

• programs represent pictures as grids of picture elements or *pixels*

Stephanos with his eraser collection
Some Pixel encodings

Bitmap
1 bit

Grayscale
8 bits

RGB Color
3 colors: red, green, blue
8 bits/color
24 bits
Comparing file sizes for 300 x 200 image:

- **Bitmap**: 1 bit, 7.3 KB
- **Grayscale**: 8 bits (1 byte), 58.6 KB
- **RGB Color**: 3 colors: red, green, blue, 24 bits (3 bytes), 175.8 KB
Is that enough?

- We’re representing color in 24 (3 * 8) bits.
  - That’s 16,777,216 ($2^{24}$) possible colors
  - Our eye can discern millions of colors - so probably pretty close
Additive/Subtractive Color

We choose 3 primary colors that can be combined to produce all the visible colors:

Additive primaries
- combining light
  Red  Green  Blue

Subtractive primaries
- combining ink, thus subtracting light
  Cyan  Yellow  Magenta

Animation: http://dx.sheridan.com/advisor/cmyk_color.html
Encoding RGB

- Each component color (red, green, and blue) is encoded as a single byte
- Colors go from (0,0,0) to (255,255,255)
  - If all three components are the same, the color is in greyscale
    - (50,50,50) at (2,2)
  - (0,0,0) (at position (1,2) in example) is black
  - (255,255,255) is white
The Color Class

• A color in a Java program is represented as an object created from the `Color` class

• The `Color` class also contains several predefined colors, including the following:

<table>
<thead>
<tr>
<th>Object</th>
<th>RGB Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color.black</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>Color.blue</td>
<td>0, 0, 255</td>
</tr>
<tr>
<td>Color.cyan</td>
<td>0, 255, 255</td>
</tr>
<tr>
<td>Color.orange</td>
<td>255, 200, 0</td>
</tr>
<tr>
<td>Color.white</td>
<td>255, 255, 255</td>
</tr>
<tr>
<td>Color.yellow</td>
<td>255, 255, 0</td>
</tr>
</tbody>
</table>
Outline

Character Strings
Variables and Assignment
Primitive Data Types
Expressions
Data Conversion
Interactive Programs
Graphics
Applets
Drawing Shapes
Applets

- A Java *application* is a stand-alone program with a `main` method (like the ones we've seen so far).
- A Java *applet* is a program that is intended to be transported over the Web and executed using a web browser.
- An applet also can be executed using the appletviewer tool of the Java SDK.
- An applet doesn't have a `main` method.
- Instead, there are several special methods that serve specific purposes.
Applets

- The `paint` method is executed automatically whenever the applet’s contents are drawn.

- The `paint` method accepts a parameter that is an object of the `Graphics` class.

- A `Graphics` object defines a `graphics context` on which we can draw shapes and text.

- The `Graphics` class has several methods for drawing shapes.
Applets

- We create an applet by extending the `JApplet` class.
- The `JApplet` class is part of the `javax.swing` package.
- This makes use of `inheritance`, which is explored in more detail in Chapter 8.
- See `Einstein.java`
import javax.swing.JApplet;
import java.awt.*;

public class Einstein extends JApplet
{
  //---------------------------------------------------------------------
  //  Draws a quotation by Albert Einstein among some shapes.
  //  ---------------------------------------------------------------------
  public void paint (Graphics page)
  {
    page.drawRect (50, 50, 40, 40);   // square
    page.drawRect (60, 80, 225, 30);  // rectangle
    page.drawOval (75, 65, 20, 20);    // circle
    page.drawLine (35, 60, 100, 120);  // line

    page.drawString ("Out of clutter, find simplicity.", 110, 70);
    page.drawString ("-- Albert Einstein", 130, 100);
  }
}
import javax.swing.JApplet;
import java.awt.*;

public class Einstein extends JApplet {
    public void paint (Graphics page) {
        page.drawRect (50, 50, 40, 40); // square
        page.drawRect (60, 80, 225, 30); // rectangle
        page.drawOval (75, 65, 20, 20); // circle
        page.drawLine (35, 60, 100, 120); // line

        page.drawString ("Out of clutter, find simplicity.", 110, 70);
        page.drawString ("-- Albert Einstein", 130, 100);
    }
}
import javax.swing.JApplet;
import java.awt.*;

public class Einstein extends JApplet {
    //=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
    // Draws a quotation by Albert Einstein among some shapes.
    //=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
    public void paint (Graphics page) {
        page.drawRect (50, 50, 40, 40); // square
        page.drawRect (60, 80, 225, 30); // rectangle
        page.drawOval (75, 65, 20, 20); // circle
        page.drawLine (35, 60, 100, 120); // line

        page.drawString ("Out of clutter, find simplicity.", 110, 70);
        page.drawString ("-- Albert Einstein", 130, 100);
    }
}
Outline

Character Strings
Variables and Assignment
Primitive Data Types
Expressions
Data Conversion
Interactive Programs
Graphics
Applets
Drawing Shapes
Drawing Shapes

• Shapes can be filled or unfilled, depending on which method is invoked

• The method *parameters* specify coordinates and sizes

• For curves and ovals we specify the shape’s *bounding rectangle*
Drawing a Line

```java
page.drawLine (10, 20, 150, 45);
page.drawLine (150, 45, 10, 20);
```

CSC 1051 M.A. Papalaskari, Villanova University
```java
page.drawRect (50, 20, 100, 40);
```
Drawing an Oval

```
page.drawOval (175, 20, 50, 80);
```
Drawing an Arc

- An arc is defined by an oval, a start angle, and an arc angle:
Drawing Shapes

• Every drawing surface has a *background color*

• Every graphics context has a current *foreground color*

• Both can be set explicitly

• **See** *Snowman.java*

• **See also Snowman applet on a webpage*
import javax.swing.JApplet;
import java.awt.*;

public class Snowman extends JApplet {

    final int MID = 150;
    final int TOP = 50;

    setBackground (Color.cyan);

    page.setColor (Color.blue);
    page.fillRect (0, 175, 300, 50); // ground

    page.setColor (Color.yellow);
    page.fillOval (-40, -40, 80, 80); // sun

    continued
page.setColor(Color.white);
page.fillOval(MID-20, TOP, 40, 40); // head
page.fillOval(MID-35, TOP+35, 70, 50); // upper torso
page.fillOval(MID-50, TOP+80, 100, 60); // lower torso

page.setColor(Color.black);
page.fillOval(MID-10, TOP+10, 5, 5); // left eye
page.fillOval(MID+5, TOP+10, 5, 5); // right eye

page.drawArc(MID-10, TOP+20, 20, 10, 190, 160); // smile

page.drawLine(MID-25, TOP+60, MID-50, TOP+40); // left arm
page.drawLine(MID+25, TOP+60, MID+55, TOP+60); // right arm

page.drawLine(MID-20, TOP+5, MID+20, TOP+5); // brim of hat
page.fillRect(MID-15, TOP-20, 30, 25); // top of hat

}
continued

```java
    page.setColor(Color.white);
    page.fillOval(MID-20, TOP, 40, 40);  // head
    page.fillOval(MID-35, TOP+35, 70, 50);  // upper torso
    page.fillOval(MID-50, TOP+80, 100, 60);  // lower torso
    page.setColor(Color.black);
    page.fillOval(MID-10, TOP+10, 5, 5);  // left eye
    page.fillOval(MID+5, TOP+10, 5, 5);  // right eye
    page.drawArc(MID-10, TOP+20, 20, 10, 190, 160);  // smile
    page.drawLine(MID-25, TOP+60, MID-50, TOP+40);  // left arm
    page.drawLine(MID+25, TOP+60, MID+55, TOP+60);  // right arm
    page.drawLine(MID-20, TOP+5, MID+20, TOP+5);  // brim of hat
    page.fillRect(MID-15, TOP-20, 30, 25);  // top of hat

    page.fillRect (MID-15, TOP-20, 30, 25);  // top of hat
```

Applet started.

```
The HTML applet Tag

• An applet is embedded into an HTML file using a tag that references the bytecode file of the applet

• The bytecode version of the program is transported across the web and executed by a Java interpreter that is part of the browser.

```html
<html>
  <head>
    <title>The Einstein Applet</title>
  </head>
  <body>
    <applet code="Einstein.class" width=350 height=175>
    </applet>
  </body>
</html>
```
Homework

• Read Sections 2.7-2.9

  – *Always* do all self-review exercises when you review material

• Do Exercises EX 2.13 -2.18