Chapter 4

Introduction to
Applets and Graphics

Topics

- Applet Structure
- Executing an Applet
- Drawing Shapes with *Graphics* Methods
- Using Colors

Applets

- Executed by a browser or applet viewer
- Opens a window, as well as the Java console
- Applet viewer comes with Java Software Development Kit (SDK)

Applet Structure

- Do not use *main* method
- Two methods called automatically:
  1. *init* method
     - Browser calls *init* when applet starts
     - Use to initialize variables and objects
  2. *paint* method
     - Browser calls after *init* and whenever window needs to be redrawn
     - Use to draw to screen
- *See Example 4.01 ShellApplet.java*

Executing an Applet

- A Web page tells the browser to run the applet
- HTML tags come in pairs, data goes between start and end tags

`<HTML> </HTML>`	start and end of HTML code
`<HEAD> </HEAD>`	start and end of header
`<TITLE> </TITLE>`	text to display in title bar
`<BODY> </BODY>`	start and end of page content

<Applet> Tag

```
<APPLET>
 CODE = Classname.class
 CODEBASE = '.'     directory of class file
 WIDTH = nnn        width of window in pixels
 HEIGHT = nnn       height of window in pixels
</APPLET>
```

Minimal HTML File

```
<HTML>
<HEAD>
  <TITLE>TitleName</TITLE>
</HEAD>
<BODY>
  <APPLET CODE="ClassName.class"
          CODEBASE=.
          WIDTH=nnn
          HEIGHT=nnn>
  </APPLET>
</BODY>
</HTML>
```

HTML File for FirstApplet

```
<HTML>
<HEAD>
  <TITLE>My First Applet</TITLE>
</HEAD>
<BODY>
  <APPLET CODE="FirstApplet.class"
          CODEBASE=.
          WIDTH=400
          HEIGHT=300>
  </APPLET>
</BODY>
</HTML>
```

Executing an Applet

- If HTML file is named *FirstApplet.html*, you can execute the applet using this command:

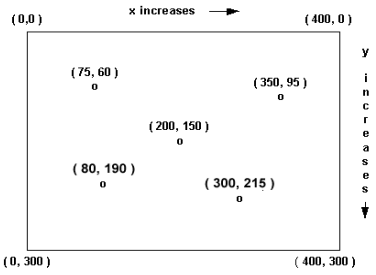
```
appletviewer FirstApplet.html
```

- Many IDEs automatically create and launch the HTML file for running an applet

The *Graphics* Class

- Browser or appletviewer sends a *Graphics* object to the *paint* method
- The *Graphics* object represents the applet window, current font, and current color
- Provides methods to draw shapes and text on the window

The Graphics Coordinate System



Graphics Class Methods

- Methods are available for drawing lines, rectangles, ovals, and other shapes, and for setting the current color
- All methods have a *void* return type, so method calls are standalone statements
- *draw…* methods draw an outlined shape
- *fill…* methods draw a solid shape

Displaying Text

Return type	Method name and argument list
void	`drawString( String s, int x, int y )` displays the String s. The (*x, y*) coordinate is lower-left corner of first letter.

- Example:

```
g.drawString( "Hello", x, y );
```

- *See Example 4.4 DrawingTextApplet.java*

Drawing a Line

Return type	Method name and argument list
void	`drawLine( int xStart, int yStart, int xEnd, int yEnd )` draws a line starting at (*xStart, yStart*) and ending at (*xEnd, yEnd*)

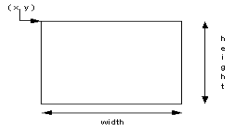
```
g.drawLine( xStart, yStart, xEnd, yEnd );
```

- *See Example 4.5 LineDrawingApplet.java*

Drawing A Rectangle

Return type	Method name and argument list
void	`drawRect( int x, int y, int width, int height )` draws an outlined rectangle with (*x,y*) as the upper-left corner and the width and height specified

```
g.drawRect( x, y, width, height );
```



Drawing A Solid Rectangle

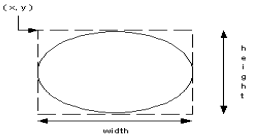
Return type	Method name and argument list
void	`fillRect( int x, int y, int width, int height )` draws a solid rectangle in the current color with (*x,y*) as the upper-left corner and the width and height specified

```
g.fillRect( x, y, width, height );
```

Drawing An Oval

Return type	Method name and argument list
void	`drawOval( int x, int y, int width, int height )` draws an outlined oval within an invisible bounding rectangle.

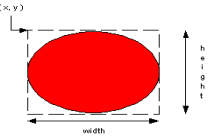
```
g.drawOval( x, y, width, height );
```



Drawing A Solid Oval

Return type	Method name and argument list
void	`fillOval( int x, int y, int width, int height )` draws a solid oval in the current color inside an invisible bounding rectangle

```
g.fillOval( x, y, width, height );
```



Drawing Squares and Circles

- To draw a square, use *drawRect* or *fillRect* with equal values for *width* and *height*.
- To draw a circle, use *drawOval* or *fillOval* with equal values for *width* and *height*

- *See Example 4.4 ShapeDrawingApplet.java*

- When drawing a figure using *Graphics* methods, specify coordinate values as offsets from a starting (x,y) coordinate.
- This will make your figure easier to move or resize.

- *See Example 4.7 Astronaut.java*

Using Color

- The *Graphics* context has a **current foreground color**
- All drawing is done in current color; the current color is in effect until changed
- The default color is black.
- To use color, import the *Color* class from the *java.awt* package

Setting the Current Color

Return value	Method name and argument list
`void`	`setColor( Color c)` sets the current color to the *Color c*

Example:

```
g.setColor( Color.RED );
```

static Color Constants

Color.BLACK
Color.WHITE
Color.RED
Color.GREEN
Color.BLUE
Color.CYAN
Color.LIGHT_GRAY
Color.DARK_GRAY
Color.GRAY
Color.ORANGE
Color.YELLOW
Color.PINK
Color.MAGENTA

Custom Colors

- Colors consist of **red**, **green**, and **blue** components (RGB).
- *Color* constructor:

`Color( int rr, int gg, int bb )` creates a color consisting of the red (*rr*), green (*gg*), and blue (*bb*) values specified. *rr*, *gg*, and *bb* must be between 0 and 255

- Example:

```
Color green = new Color( 0, 255, 0 );
```

- *See Example 4.8 AstronautWithColor.java*

Homework1

- Solve the following Problems:
- Number 31 Page 31
- Number 35 Page 203