Java Chapter 14 Part 2

* More on Shapes, Mouse Events, Timers, and Audio
* CIS 255 • Shelby-Hoover Campus

Drawing in a Smaller Area

* The paint method uses the entire applet area as the canvas
* You may want to restrict the canvas to a specific component in the window, such as a panel; coordinates should then be relative to the upper-left corner of that component
* A class that extends JPanel can contain a void method named paintComponent with a Graphics object parameter that indicates what shapes should be drawn on that panel
* The paintComponent method can be triggered by invoking repaint() on the panel object

Panel Painting Example

* The applet in GraphicsWindow.java (Code Listing 14-17) allows the user to select from among seven different shapes to be drawn in a panel
  + The checkboxes, and their titles, are in arrays
  + Each checkbox has an ItemListener
  + The listener method itemStateChanged invokes the method repaint for the panel drawingPanel
* The code for the panel is in a separate class, DrawingPanel.java (Code Listing 14-18)
  + The constructor for the panel receives the checkboxes so it can check each box’s state
  + The preferred size of the panel is set using the Dimension class
  + The method paintComponent calls the superclass method (inherited from JPanel) and then checks the selection of each box to determine the colors and shapes to be drawn

Mouse Events

* There are two separate listener interfaces for mouse actions
* MouseListener requires the following methods:
  + mousePressed (pressing down on the mouse button but not releasing it)
  + mouseClicked (pressing and releasing the mouse button in one location)
  + mouseReleased (called after mousePressed when the mouse button is released)
  + mouseEntered (the pointer enters the area)
  + mouseExited (the pointer exits the area)
* MouseMotionListener requires these methods:
  + mouseDragged (called after mousePressed when the pointer is moved with the button pressed)
  + mouseMoved (called when the pointer is moved over the area)
* Each method is a public void method with a single argument: a MouseEvent object
* The MouseEvent object supports calls to the methods getX() and getY() that return the coordinates of the pointer when the event occurs

Mouse Events Example

* MouseEvents.java (Code Listing 14-19) uses an array of text fields to display information about the current mouse events
* The private inner class MyMouseListener handles singular mouse events
  + Each text field not related to the current event is reset to a gray background
  + The text field related to the current event is highlighted with a yellow background
* The private inner class MyMouseMotionListener handles events with motion
  + The text field for dragging is highlighted by itself
  + When the mouse moves, the current X and Y coordinates are shown in separate text fields

Listener Adapter Classes

* A mouse-driven program may not require action for every single type of mouse event
* When a class implements an interface, it must contain a definition for every method in that interface, even if the body is empty
* Java provides **adapter classes** for these listener interfaces that already include definitions with empty bodies for all of the required methods; the program then overrides specific methods to provide behavior
  + Instead of implements MouseListener, use extends MouseAdapter
  + Instead of implements MouseMotionListener, use extends MouseMotionAdapter

Adapter Classes Example

* In DrawBoxes2.java (Code Listing 14-21), the applet draws a rectangle based on the users click (for the upper-left corner) and drag (for the width and height)
* The only events of interest are mousePressed (for the upper-left X and Y) and mouseDragged (for the width and height)
* Because the private inner classes extend the adapter classes instead of implementing the interfaces, only the methods that require action are defined in the inner classes

Timer Objects

* Sometimes an action should occur based on the passing of an interval of time
* An object of the Java class Timer generates ActionEvent objects at a certain interval
  + The first argument to the constructor is the delay between event firings in milliseconds
  + The second argument is an object of an ActionListener type (a class that implements that interface)
* Timer objects support several methods:
  + addActionListener(listener\_object) (in the event that the original constructor argument was null)
  + getDelay() and setDelay(int\_value)
  + isRunning() (returns true or false)
  + start() and stop()
* A program must start the timer (usually in init())

Timer Example

* The applet in BouncingBall.java (Code Listing 14-22) moves a solid red oval 20 pixels up or down every 30 milliseconds
* The init method creates the timer object and starts it
* The paint method draws the solid oval based on the current coordinates
* The listener checks to see the direction the ball is moving and whether or not the ball has room to move; it then either adds or subtracts 20 pixels from the y coordinate, or it changes the direction of the ball
* The listener calls repaint to draw the ball in a new location

Audio

* The Java class Applet has a method named play that supports playing an audio file one time
  + The first argument is the URL for the location of the file (not including the name)
    - The method getDocumentBase() returns a URL object with the location of the HTML file embedding the applet
    - The method getCodeBase() returns a URL object with the location of the class file
  + The second argument is the file name (.aif / .aiff, .au, .mid / .rmi, or .wav)
* A more flexible approach from the java.applet package is to use the class AudioClip
  + Instead of constructing an AudioClip object, assign a call to getAudioClip (with the same arguments as to play above) to the AudioClip variable
  + AudioClip objects support the methods play(), loop(), and stop() (no arguments needed)

Audio Example

* The class AudioDemo2.java (Code Listing 14-23) creates an applet with three buttons to control playback of a sound clip
* A private AudioClip object is assigned the value returned by getAudioClip for an audio file in the same location as the HTML file
* Each button has a listener; the private inner listener class detects which button is clicked and invokes the appropriate method on the AudioClip object

Audio in Applications

* Since an application does not extend JApplet, the syntax is a little different
* The first step is to create a File object for the file to be played:  
    
  File file = new File("step.wav");
* Next, create a URI (Uniform Resource Identifier) object from the File object:  
    
  URI uri = file.toURI();
* Third, create a URL (Uniform Resource Locator) object from the URI object:  
    
  URL url = uri.toURL();
* Finally, invoke the static method newAudioClip on the class Applet, with the URL object as the argument, and assign this to an AudioClip object:  
    
  sound = Applet.newAudioClip(url);
* The full example (AudioFrame.java) shows that this requires the program to import the packages java.applet, java.io (for File), and java.net (for URL); methods that deal with the class URL (or call such a method) must include a throws clause for the exception type MalformedURLException

Reminders for Chapter 14

* An applet class extends JApplet instead of JFrame, and it contains the method init in place of a constructor
* HTML documents are placed within an html block and are divided into head and body blocks
  + The head section usually contains a title block
  + The body section can consist of a combination of paragraph and various header blocks
* Although the tag object is the preferred way to embed an applet in an HTML body, the tag applet is more widely supported; specify the class file name, the width, and the height in the tag
* An applet can contain any of the usual GUI components (buttons, text fields, labels)
* An applet can also serve as a canvas for drawing
* Calls to drawing methods should be nested within the method paint or paintComponent, and the first statement in these methods should be a call to the superclass version (e.g., super.paint(g);)

More Reminders for Chapter 14

* Set the color and /or the font of the Graphics object before drawing a shape or text
* Methods beginning with draw display hollow shapes, whereas methods beginning with fill display filled shapes
* Coordinates begin at the upper-left corner of the canvas
* The paint / paintComponent method automatically executes when the window first appears or an overlapping window is moved; to trigger it again (e.g., in response to an event), call repaint
* The mouse and mouse motion listener interfaces require the definition of multiple methods; their corresponding adapter classes only require a programmer to override select methods
* A Timer object generates ActionEvent objects with a certain interval between them; the timer must have an associated listener and must be started
* Audio clips may be incorporated into applets and applications, but the syntax varies