THE *TIMER* CLASS

A **Timer** is an object that allows users to time certain events that occur in a program. If, for example, you are creating a memory game, a timer could come in handy if you want to time how long it takes the user to clear the board. The **Timer** class implements a basic timer that generates events at regular intervals and lets you set up a listener object to handle these events. Before we start looking at how a timer works, however, let's take a look at some of the common constructors and methods.

CONSTRUCTORS

The following are a list of constructors included in the **JTextArea** class:

CONSTRUCTOR	DESCRIPTION
Timer(int delay, ActionListener listener)	Creates a timer and initializes both the initial delay and between-event delay in milliseconds.

METHODS

The following table outlines some of the methods that are used when working with a text area:

METHOD	DESCRIPTION
boolean isRunning()	Returns **true** if the timer is running.
void restart()	Restarts the timer, canceling any pending firings and causing it to fire with its initial delay
void setDelay(int delay)	Sets the timer's between-event delay, the number of milliseconds between successive action events.
void setInitialDelay(int initialDelay)	Sets the timer's initial delay, the time in milliseconds to wait after the timer is started before firing the first event.
void start()	Starts the timer, causing it to start sending action events to its listeners.
void stop()	Stops the timer, causing it to stop sending action events to its listeners.

In the following example, we are going to use a simple **Timer** object for a program that outputs to **JLabel** the time in seconds each time a second transpires.

The first thing we'll need to do is import **javax.swing** since the **Timer** class and the **ActionEvent** class are included in this package:

```
import javax.swing.Timer;
```

```
import java.awt.event.ActionEvent;
```

Next we will need to implement **ActionListener** in our class declaration and declare a **Timer** object globally:

```
public class TimerExample extends JFrame implements ActionListener {

   private Timer tmrTimer;
   ...

   public static void main(String[] args) {

      new TimerExample();
   }
}
```

We will then need to initialize our **Timer** object in our constructor. The first parameter is an **int** value that represents how often the timer events occur. The second parameter is an object that implements the **ActionListener** interface (in this case, it is our class which is extending **JFrame**). Once we have initialized the timer, we are going to start it right away. We do this by calling the **start()** method.

```
public TimerExample() {

   tmrTimer = new Timer(1000, this);
   tmrTimer.start();
   ...
}
```

Finally, we will need to create a method called **actionPerformed()** which receives a parameter of type **ActionEvent** (the same method that we used in Unit 1 for our **JButton** objects). In this method, we need to write whatever it is we want to happen at each interval (in this case, every 1000 milliseconds or 1 second). In our example, we just want the number of seconds to go up by 1 (which I have stored in an **int** variable called **seconds**) and output the value in a label called **lblTimer**.

```
public void actionPerformed(ActionEvent e) {

   seconds++;
   lblTimer.setText(Integer.toString(seconds));
}
```

There's nothing more to a **Timer** object than that. Again, a **Timer** is a very useful object whenever you want to create a program that requires something to happen at regular intervals.