**TITLE**

Multiple Tickers

# **TOPICS**

Multithreading

Animation

Configuring Properties

Customizing Objects

# **DESCRIPTION**

**General**

Create an application that displays three moving stock tickers one below the other and a panel containing two buttons labeled Start All, Stop All. Initially all tickers are stopped. The Start All and Stop All buttons are used to start or stop all tickers simultaneously. (See picture below).

**Instructions**

Do this assignment by enhancing the last exercise. In the last exercise, you developed a simple Ticker object with only one property namely moving. A true value of this boolean property indicated that the ticker moving and a false value the ticker stopped. You were able to start and stop the ticker by controlling the value of this property using Start and Stop buttons. In this exercise, you will enhance the simple ticker of the last exercise and provide the following additional properties:

Ticker Name

Ticker Price

Ticker Direction of Movement

Ticker Sleep Time (i.e. Ticker Speed).

You will drop three instances of the Ticker object in a frame and will configure these instances to have property values as below.

Configure each ticker to be initially stopped.

Configure each ticker to have a different stock name and price.

Configure the first and the third ticker to be moving in the forward direction and the middle ticker in the reverse direction.

Configure a different sleep time (speed) value for each ticker.

Configure all the tickers before starting the program. Thus, when the program is running, each ticker will display its symbol and price, move in its configured direction at its configured speed.

# **Details**

Create the following classes:

**Ticker Class**

**JFrameExt Class**

Create a JFrameExt class using an extended JFrame. Do this by creating an application and its frame as below:

��������� File | new | General tab | select Application Icon

Set layout of JFrameExt�s content pane to GridLayout with 4 rows and 1 column.

Drop three Ticker (extended JPanel) objects in the content pane.

Drop one standard JPanel in the content pane.

In the standard JPanel, drop two buttons labeled Start All and Stop All.

Code event handlers for the buttons to start and stop all tickers simultaneously.

# **IMPLEMENTATION**

Create the following classes.

**Ticker class**

**Algorithm**

Create the Ticker class as an extended JPanel.

Implement in it the interface Runnable and provide the method run.

Code the method run to animate the ticker.

Code the method paintComponent to draw the ticker name and price.

Implementation

Create a class Ticker that extends JPanel and implements interface Runnable. Provide the following in the class.

## 

## Properties

## Provide the following properties along with their get/set methods:�

(Property moving is already discussed above and is included below for completion).

String stockName

Represents the name of the stock to be displayed.

double stockPrice

Represents the price of the stock.

boolean direction

Represents the direction in which the ticker should be moving.

A true value indicates left to right movement.

A false value indicates right to left movement.

long sleepTime

Represents the sleepTime in milliseconds. It represents the amount of time that the ticker thread should sleep during each pass through the loop. The sleep time controls the speed of the moving ticker. A low value of sleepTime indicates a fast moving ticker and a high value of sleepTime indicates a slow moving ticker.

boolean moving��������������

Represents moving property. It is used to start/stop the ticker.

A true value indicates that the ticker is moving.

A false value indicates that the ticker is stopped.

All of the above properties will have default initial values. The moving will have a default value of false.

�

***moving Property***

The moving property along with its get/set method is presented below as an example.

private boolean moving = false;

public void setMoving (boolean newMoving )

{

��������� moving = newMoving;

}

public boolean isMoving ( )

{

��������� return moving;

}

**Ticker Constructor**

Provide a parameterless constructor in which you do the following:

Set moving property to false so the ticker will be initially stopped.

Create a Thread object and pass it your own reference (this).

Start the Thread object. (Thread will run but will not animate because moving is set to false).

**paintComponent Method**

Override the method paintComponent so that it draws the ticker symbol and price.

The sample method below draws the text �DVC� starting at coordinates x and y.

x and y are instance variables whose values are updated by another thread.

The other thread updates the values of x, y and calls repaint. Calling of repaint results in the method paintComponent being called and the ticker symbol and date being redrawn. The sample pseudocode for the other thread is given later under the run method. The sample code for the paintComponent method is given below.

public void paintComponent (Graphics g)

{

��������� //Call the parent constructor to clear the drawing surface.

super.paintComponent (g);

//Draw the string starting at coordinates x, y.

g.drawString (x, y, �DVC�);

}

**run Method**

Provide a method run to implement the interface Runnable.

Sample pseudo code below shows a run method that animates the stock ticker by using an infinite loop.

�

(Note that in the infinite loop, sleep is outside the if statement. So the infinite loop always sleeps for a certain amount of time, say 100 ms, even when moving is false. This allows other threads to run. Otherwise, the program will hang in the infinite loop).

while(true)

{

��������� if (moving)

��������� {�������

Update stock Ticker�s draw co-ordinates.

������������������ Call Ticker�s repaint method.

��������� }

��������� Sleep for a fixed time.

}

**Application Frame**

Create an application frame and do the following in the frame:

Set the frame�s content pane�s layout to GridLayout.

Set the GridLayout object�s row property to four and column to 1.

In the frame�s content pane, drop three instances of Ticker (extended JPanel) object.

In the frame�s content pane, drop one instance of standard JPanel.

Set the JPanel�s layout to FlowLayout.

Drop two buttons labeled Start All, Stop All onto the JPanel. (These buttons will be used to start and stop all tickers simultaneously).

Code the Start button so that clicking it will set the moving property of all Ticker objects to true.

Code the Stop button so that clicking it will set the moving property of all Ticker objects to false.

**Configuring Tickers**

Configure the appropriate initial values for each of the three Ticker objects using the property inspector.

(Do this by displaying the frame in the design mode of the builder tool and by selecting the appropriate Ticker object for configuration).

Set the moving property of all Ticker objects to false.

Set the direction property to false for the middle Ticker object and true for the other two.

Set other properties appropriately.

## **TESTING**

During execution, test that all tickers can be started and stopped simultaneously using Start All and Stop All buttons. That each displays its name and price correctly, moves in the correct direction at the expected speed and wraps around correctly.

**APPLICATION FRAME PICTURE**

****

**SAMPLE CODE FOR WAIT/NOTIFY**

� public void run()

� {

��� while(true)

��� {

����� myRun();

����� try

����� {

��������� Thread.sleep(100);

����� }

����� catch (InterruptedException ex)

����� {

����� }

��� }

� }

� public synchronized void myRun ()

� {

��� if(!moving)

��� {

����� try

����� {

��������� wait();

����� }

����� catch (InterruptedException ex)

����� {

����� }

��� }

��� else

��� {

����� if (x >= this.getWidth() - stringWidth)

��������� x = 10;

����� else

��������� x += 10;

����� this.repaint();

��� }

� }

� public synchronized void setMoving(boolean moving)

� {

��� this.moving = moving;

��� notify();

� }