**TITLE**

## Simple Ticker

# **TOPICS**

## Multithreading

Animation

# 

# **DESCRIPTION**

# **General**

## Create a java application that displays a moving stock ticker which can be started and stopped using Start and Stop buttons. See picture below. Initially, the ticker is stopped. �It is started by pushing the Start button.

The ticker should display the stock symbol along with its price. The ticker should start at the left edge of the panel showing both the name and the price. It should move to the right. When the last character of ticker�s price reaches the right edge, the ticker should disappear from the right and reappear at the left and continue moving in a cycle.� At all times, the user should see the full ticker including the symbol and the price. At no time, the user should see only a part of the ticker�s symbol or price.

# **DISCUSSION**

**Multithreading**

Java language provides for creating multiple threads.

One way of creating a thread in a class say JPanelExt that extends JPanel by doing the following in the class JPanelExt:

�••••••• Implement the interface Runnable in the class. This will require that you provide the method run.

�••••••• Create a Thread object (say in the constructor) and pass it the self reference (this). (The Thread constructor accepts only a reference of type Runnable. It will accept the above self reference because you have implemented the interface Runnable).

�••••••• Start a new thread (say in the constructor) by calling the method start of the Thread object. (The Thread object will start a new thread and will launch on it the method run of the object passed t the constructor of the Thread object during the creation of the Thread object. The run method will execute in a separate thread but will be able to access the instance variables and methods of the class in which it resides).

**Animation**

Animation implies creating a moving text/picture programmatically. This is accomplished by redrawing the picture over and over again in an infinite loop. This is usually done in a separate thread. In each pass through the loop, the coordinates of the text are changed before redrawing so the text appears moving to the user. Also sleep is called in each pass through the loop to allow other threads to run. Otherwise, the infinite loop will monopolize the CPU and the program will hang.

�

# **ALGORITHM**

For this assignment, create the following classes.

**Ticker class**

Create the Ticker class as an extended JPanel.

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

Code the run method to animate the ticker.

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

**JFrameExt class**

Create the class JFrameExt as an extended JFrame.

Set the layout of its content pane to be BorderLayout.

Drop a ticker (extended JPanel) object in the center of the content pane.

Drop a JPanel towards the south of the content pane.

In the JPanel, drop two buttons labeled Start, Stop.

Code the button event handlers appropriately for starting and stopping the ticker.

# **IMPLEMENTATION**

For this assignment, implement the following classes.

**Ticker class**

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

**moving property**

Provide a boolean property moving with its access methods as below. A true value of the property will imply a moving ticker and a false value a stopped ticker.

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 in the method run. The other thread updates the values of x, y and calls repaint.

Calling of repaint results in this method being called and text redrawn. The pseudo code for the other thread is given later under the run method).

//instance variables

private String symbol = �DVC 32.0�;

private int symbolWidth;

private int x = 30;

private int y = 100;

public void paintComponent (Graphics g)

{

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

super.paintComponent (g);

//Determine the width of String symbol

//You will need this width value in the run method

//to determine when to set the x value back to 0

FontMetrics fm = g.getFontMetrics();

symbolWidth = fm.stringWidth(symbol);

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

g.drawString (x, y, symbol);

}

**run Method**

Implement the interface Runnable and provide a method run.

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.

}

**JFrameExt class**

Create a class JFrameExt that extends JFrame. In the constructor of the class, do the following:

Set the layout of JFrameExt�s content pane to BorderLayout.

Drop a Ticker (extended JPanel) object on the CENTER part of the content pane.

Drop a standard JPanel object onto the SOUTH part of the content pane.

Set the JPanel�s layout to FlowLayout.

Drop two buttons labeled Start, Stop onto the JPanel. (The buttons will be used to start and stop the ticker).

Code the Start button event handler so that it will set the moving property to true.

Code the Stop button event handler so that it will set the moving property to false.

### APPLICATION FRAME PICTURE

****