AWT LINE2D CLASS

Introduction

The Line2D class states a line segment in x, y coordinate space.

Class declaration

Following is the declaration for **java.awt.geom.Line2D** class:

```
public abstract class Line2D
   extends Object
      implements Shape, Cloneable
```

Class constructors

S.N.	Constructor & Description
1	**protected Line2D** This is an abstract class that cannot be instantiated directly.

Class methods

S.N.	Method & Description
1	**Object clone** Creates a new object of the same class as this object.
2	**boolean contains*doublex, doubley*** Tests if a specified coordinate is inside the boundary of this Line2D.
3	**boolean contains*doublex, doubley, doublew, doubleh*** Tests if the interior of this Line2D entirely contains the specified set of rectangular coordinates.
4	**boolean contains*Point2Dp*** Tests if a given Point2D is inside the boundary of this Line2D.
5	**boolean contains*Rectangle2Dr*** Tests if the interior of this Line2D entirely contains the specified Rectangle2D.
6	

Rectangle getBounds

Returns an integer Rectangle that completely encloses the Shape.

7

abstract Point2D getP1

Returns the start Point2D of this Line2D.

8

abstract Point2D getP2

Returns the end Point2D of this Line2D.

9

PathIterator getPathIterator*AffineTransformat*

Returns an iteration object that defines the boundary of this Line2D.

10

PathIterator getPathIterator*AffineTransformat, doubleflatness*

Returns an iteration object that defines the boundary of this flattened Line2D.

11

abstract double getX1

Returns the X coordinate of the start point in double precision.

12

abstract double getX2

Returns the X coordinate of the end point in double precision.

13

abstract double getY1

Returns the Y coordinate of the start point in double precision.

14

abstract double getY2

Returns the Y coordinate of the end point in double precision.

15

boolean intersects*doublex, doubley, doublew, doubleh*

Tests if the interior of the Shape intersects the interior of a specified rectangular area.

16

boolean intersects*Rectangle2Dr*

Tests if the interior of the Shape intersects the interior of a specified Rectangle2D.

17

boolean intersectsLine*doublex1, doubley1, doublex2, doubley2*

Tests if the line segment from $x1, y1$ to $x2, y2$ intersects this line segment.

18

boolean intersectsLine*Line2Dl*

Tests if the specified line segment intersects this line segment.

19

static boolean linesIntersect
***double*x1, *double*y1, *double*x2, *double*y2, *double*x3, *double*y3, *double*x4, *double*y4**

Tests if the line segment from $x1, y1$ to $x2, y2$ intersects the line segment from $x3, y3$ to $x4, y4$.

20

double ptLineDist*doublepx, doublepy*

Returns the distance from a point to this line.

21

static double ptLineDist*doublex*1, *doubley*1, *doublex*2, *doubley*2, *doublepx, doublepy*

Returns the distance from a point to a line.

22

double ptLineDist*Point2Dpt*

Returns the distance from a Point2D to this line.

23

double ptLineDistSq*doublepx, doublepy*

Returns the square of the distance from a point to this line.

24

static double ptLineDistSq*doublex*1, *doubley*1, *doublex*2, *doubley*2, *doublepx, doublepy*

Returns the square of the distance from a point to a line.

25

double ptLineDistSq*Point2Dpt*

Returns the square of the distance from a specified Point2D to this line.

26

double ptSegDist*doublepx, doublepy*

Returns the distance from a point to this line segment.

27

static double ptSegDist*doublex*1, *doubley*1, *doublex*2, *doubley*2, *doublepx, doublepy*

Returns the distance from a point to a line segment.

28

double ptSegDist*Point2Dpt*

Returns the distance from a Point2D to this line segment.

29

double ptSegDistSq*doublepx, doublepy*

Returns the square of the distance from a point to this line segment.

30

static double ptSegDistSq*double***x1, *double***y1, *double***x2, *double***y2, *double***px, *double***py**

Returns the square of the distance from a point to a line segment.

31

double ptSegDistSq*Point2D***pt**

Returns the square of the distance from a Point2D to this line segment.

32

int relativeCCW*double***px, *double***py**

Returns an indicator of where the specified point *px*, *py* lies with respect to this line segment.

33

static int relativeCCW*double***x1, *double***y1, *double***x2, *double***y2, *double***px, *double***py**

Returns an indicator of where the specified point *px*, *py* lies with respect to the line segment from *x*1, *y*1 to *x*2, *y*2.

34

int relativeCCW*Point2D***p**

Returns an indicator of where the specified Point2D lies with respect to this line segment.

35

abstract void setLine*double***x1, *double***y1, *double***x2, *double***y2**

Sets the location of the end points of this Line2D to the specified double coordinates.

36

void setLine*Line2D***l**

Sets the location of the end points of this Line2D to the same as those end points of the specified Line2D.

37

void setLine*Point2D***p1, *Point2D***p2**

Sets the location of the end points of this Line2D to the specified Point2D coordinates.

Methods inherited

This class inherits methods from the following classes:

- java.lang.Object

Line2D Example

Create the following java program using any editor of your choice in say **D:/ > AWT > com > tutorialspoint > gui >**

AWTGraphicsDemo.java

```
package com.tutorialspoint.gui;

import java.awt.*;
```

```
import java.awt.event.*;
import java.awt.geom.*;

public class AWTGraphicsDemo extends Frame {

   public AWTGraphicsDemo(){
      super("Java AWT Examples");
      prepareGUI();
   }

   public static void main(String[] args){
      AWTGraphicsDemo  awtGraphicsDemo = new AWTGraphicsDemo();
      awtGraphicsDemo.setVisible(true);
   }

   private void prepareGUI(){
      setSize(400,400);
      addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent windowEvent){
            System.exit(0);
         }
      });
   }

   @Override
   public void paint(Graphics g) {
      Line2D shape = new Line2D.Double();
      shape.setLine(250D,250D,150D,150D);
      Graphics2D g2 = (Graphics2D) g;
      g2.draw (shape);
      Font font = new Font("Serif", Font.PLAIN, 24);
      g2.setFont(font);
      g.drawString("Welcome to TutorialsPoint", 50, 70);
      g2.drawString("Line2D.Line", 100, 120);
   }
}
```

Compile the program using command prompt. Go to **D:/ > AWT** and type the following command.

```
D:\AWT>javac com\tutorialspoint\gui\AWTGraphicsDemo.java
```

If no error comes that means compilation is successful. Run the program using following command.

```
D:\AWT>java com.tutorialspoint.gui.AWTGraphicsDemo
```

Verify the following output