JAVA.LANG.MATH.NEXTAFTER METHOD

 Copyright © tutorialspoint.com

Description

The **java.lang.Math.nextAfter***doublestart, doubledirection* returns the floating-point number adjacent to the first argument in the direction of the second argument. If both arguments compare as equal the second argument is returned. Special cases:

- If either argument is a NaN, then NaN is returned.
- If both arguments are signed zeros, direction is returned unchanged *asimpliedbytherequirementofreturningthesecondargumentiftheargumentscompareasequal*.
- If *start* is -Double.MIN_VALUE and direction has a value such that the result should have a smaller magnitude, then a zero with the same sign as start is returned.
- If *start* is infinite and direction has a value such that the result should have a smaller magnitude, Double.MAX_VALUE with the same sign as start is returned.
- If *start*is equal to - Double.MAX_VALUE and direction has a value such that the result should have a larger magnitude, an infinity with same sign as start is returned.

Declaration

Following is the declaration for **java.lang.Math.nextAfter** method

```
public static double nextAfter(double start, double direction)
```

Parameters

- **start** -- starting floating-point value
- **direction** -- value indicating which of start's neighbors or start should be returned

Return Value

This method returns the floating-point number adjacent to start in the direction of direction.

Exception

- **NA**

Example

The following example shows the usage of lang.Math.nextAfter method.

```
package com.tutorialspoint;

import java.lang.*;

public class MathDemo {

   public static void main(String[] args) {

   // get two double numbers
   double x = 98759.765;
   double y = 154.28764;

   // print the next number for x towards y
   System.out.println("Math.nextAfter(" + x + "," + y + ")="
   + Math.nextAfter(x, y));

   // print the next number for y towards x
```

```
   System.out.println("Math.nextAfter(" + y + "," + x + ")="
   + Math.nextAfter(y, x));
   }
}
```

Let us compile and run the above program, this will produce the following result:

```
Math.nextAfter(98759.765,154.28764)=98759.76499999998
Math.nextAfter(154.28764,98759.765)=154.28764000000004
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js