JAVA.MATH.BIGDECIMAL.TOSTRING METHOD

Description

The **java.math.BigDecimal.toString** returns the string representation of this BigDecimal, using scientific notation if an exponent is needed.

A standard canonical string form of the BigDecimal is created as though by the following steps: first, the absolute value of the unscaled value of the BigDecimal is converted to a string in base ten using the characters '0' through '9' with no leading zeros $exceptifitsvalueiszero, inwhichcaseasingle'0'characterisused.$

Next, an adjusted exponent is calculated; this is the negated scale, plus the number of characters in the converted unscaled value, less one. That is, -scale+$ulength - 1$, where ulength is the length of the absolute value of the unscaled value in decimal digits $itsprecision$.

If the scale is greater than or equal to zero and the adjusted exponent is greater than or equal to -6, the number will be converted to a character form without using exponential notation.

In this case, if the scale is zero then no decimal point is added and if the scale is positive a decimal point will be inserted with the scale specifying the number of characters to the right of the decimal point. '0' characters are added to the left of the converted unscaled value as necessary. If no character precedes the decimal point after this insertion then a conventional '0' character is prefixed.

Otherwise $thatis, ifthescaleisnegative, ortheadjustedexponentislessthan - 6$, the number will be converted to a character form using exponential notation. In this case, if the converted BigInteger has more than one digit a decimal point is inserted after the first digit.

An exponent in character form is then suffixed to the converted unscaled value $perhapswithinserteddecimalpoint$; this comprises the letter 'E' followed immediately by the adjusted exponent converted to a character form.

The latter is in base ten, using the characters '0' through '9' with no leading zeros, and is always prefixed by a sign character '-' $'\u002D'$ if the adjusted exponent is negative, '+' $'\u002B'$ otherwise.

Finally, the entire string is prefixed by a minus sign character '-' $'\u002D'$ if the unscaled value is less than zero. No sign character is prefixed if the unscaled value is zero or positive.

Declaration

Following is the declaration for **java.math.BigDecimal.toString** method

```
public String toString()
```

Overrides

- toString in class **Object**

Parameters

- **NA**

Return Value

This method returns string representation of this BigDecimal

Exception

- **NA**

Example

The following example shows the usage of math.BigDecimal.toString method

```
package com.tutorialspoint;

import java.math.*;

public class BigDecimalDemo {

     public static void main(String[] args) {

        // create a BigDecimal object
        BigDecimal bg;

        // create a String object
        String s;

 MathContext mc = new MathContext(3); // 3 precision

        bg = new BigDecimal("1234E4", mc);

        // assign the string value of bg to s
        s = bg.toString();

 String str = "String value of " + bg + " is " + s;

        // print s value
        System.out.println( str );
    }
}
```

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

```
String value of 1.23E+7 is 1.23E+7
```