Java DecimalFormat Example

Jerry Zhao November 10, 2017 0

We often need to format numbers, such as taking two decimal places for a number, show only integer part of a number etc. Java provide

java.text.DecimalFormat class, which can helps you to format numbers use your specified pattern as quickly as possible.

1. java.text.DecimalFormat Creation.

1. **DecimalFormat Constructor** : The DecimalFormat constructor's parameter is the format pattern which the numbers should be formatted with.

```
String formatPattern = "###,###.###";

DecimalFormat df = new DecimalFormat(formatPattern);
```

2. **applyPattern(String newPattern)** : This method is used to change the number format pattern to a new pattern for the DecimalFormat instance. For example: decimalFormat.applyPattern("#0.##");
3. **applyLocalizedPattern(String newPattern)** : This method is similar to applyPattern(String newPattern), it also change the number format pattern. The difference is that it will use localized character in the pattern string instead of standard characters. For example, in Danish comma is used as the integer and fraction separator. But in English, dot is used as that separator. So **decimalFormat.applyLocalizedPattern("#0,##");** is used to format a number in Danish language.

2. Number Formatting Pattern Syntax.

Following characters can be used in number formatting pattern.

1. 0 : If the digit is insufficient then fill in 0. The digits will always be shown if it is exist, otherwise a 0 character will be shown.
2. # : Show the digits only when it is not 0, otherwise omit.
3. . : Decimal separator which is used to separate integer part and decimal fraction part.

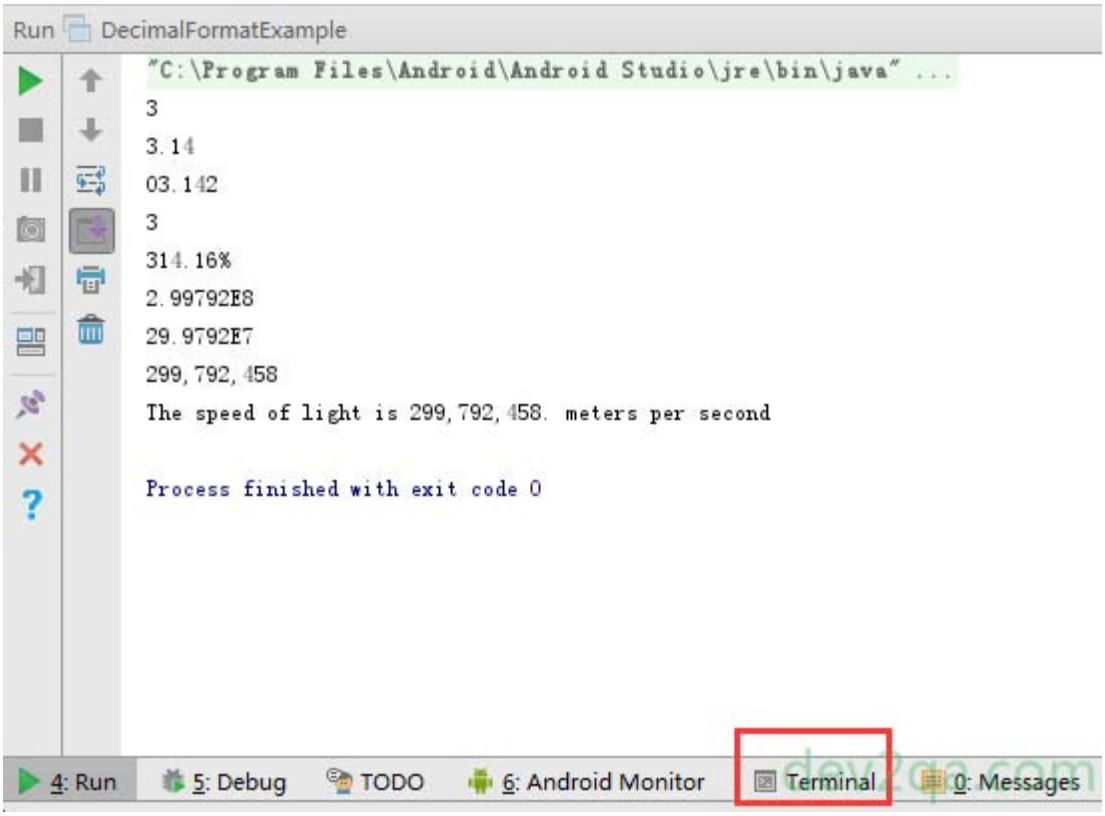
4. , : This is a grouping separator. For example thousand separator.
5. E : Formatting the number use scientific notation. E5 means ： The five power of 10.
6. % : Multiplies the format number by 100, shows the formatted number with percentage.
7. – : Format the number with negative prefix.
8. ; : Format pattern separator.
9. ? : Multiplies the format number by 1000, shows the formatted number with per mille.
10. X : Use the character as number prefix or suffix.
11. ¤ : This is the currency sign, which will be replaced by the locale currency sign.
12. ¤¤ : Format number use international monetary symbols.
13. ‘ : The quote character around format number prefix or suffix.

READ : Java TimeZone Example

3. java.text.DecimalFormat Example

```
package com.dev2qa.example.java;

import java.text.DecimalFormat;

public class DecimalFormatExample {
    public static void main(String args[])
    {
        // PI
        double pi=3.1415927;

        // Take one integer.
        DecimalFormat df = new DecimalFormat("0");
        System.out.println(df.format(pi)); // 3

        //Take one integer and two decimal
        df = new DecimalFormat("0.00");
        System.out.println(df.format(pi)); // 3.14

        // Take two integers and three decimals, and the inadequacies for integer are filled
with 0.
```

```
        df = new DecimalFormat("00.000");
        System.out.println(df.format(pi));// 03.142

        // Take all the integers.
        df = new DecimalFormat("#");
        System.out.println(df.format(pi)); // 3

        // Count as a percentage and take two decimal places.
        df = new DecimalFormat("#.##%");
        System.out.println(df.format(pi)); //314.16%

        // light speed.
        long lightSpeed = 299792458;

        // Display use scientific counting method and take five decimal places.
        df = new DecimalFormat("#.#####E0");
        System.out.println(df.format(lightSpeed)); //2.99792E8

        // Use scientific counting method to show two integers and four decimal places.
        df = new DecimalFormat("00.####E0");
        System.out.println(df.format(lightSpeed)); //29.9792E7

        //Each three integer are separated by commas.
        df = new DecimalFormat(",###");
        System.out.println(df.format(lightSpeed)); //299,792,458

        //Embed formatting in text.
        df = new DecimalFormat("The speed of light is ,### meters per second.");
        System.out.println(df.format(lightSpeed));
    }
}
```



4. Special Locale DecimalFormat Instance Creation.

If you want to create a DecimalFormat object for a special locale not for JVM default locale. You can use below code.

```
// Create special Locale DecimalFormat instance.
Locale specialLocale  = new Locale("fr", "FR");
String formatPattern = "###.##";

DecimalFormat df1 = (DecimalFormat) NumberFormat.getNumberInstance(specialLocale);
df1.applyPattern(formatPattern);

String format = df1.format(123456789.123);
System.out.println(format);
```

The output should be : 123456789,12.

5. java.text.DecimalFormatSymbols

DecimalFormatSymbols class can be used to change the decimal separator, grouping separator character.

```
// Use DecimalFormatSymbols to change decimal separators.
Locale specialLocale1 = new Locale("fr", "FR");
DecimalFormatSymbols decimalFormatSymbols = new DecimalFormatSymbols(specialLocale1);
decimalFormatSymbols.setDecimalSeparator(';');
decimalFormatSymbols.setGroupingSeparator(':');

String formatPattern1 = "#,##0.###";
DecimalFormat df2 = new DecimalFormat(formatPattern1, decimalFormatSymbols);

String number = df2.format(123456789.1234567);
System.out.println(number);
```

The output should be : 123:456:789;123. You can refer java.text.DecimalFormatSymbols to see full methods introduction.

READ : Java I/O(Input/Output) Overview

6. Grouping Digits

You can use DecimalFormat's **setGroupingSize()** method to change the default group digits number as you need. The default group digits number is 3. Below is an example.

```
String formatPattern2 = "#,###.###";
DecimalFormat df3 = new DecimalFormat(formatPattern2);
df3.setGroupingSize(4);

String number1 = df3.format(123456789.123);
System.out.println(number1);
```

The output should be : 1,2345,6789.123.

But you can also use below number format pattern to achieve same effect.

String formatPattern = "####,####.###";