Java Integer To String Examples

Feb 15, 2015 Core Java, Examples, Snippet, String 10 comments

This tutorial will show example codes on how to convert **Java Int to String**. Performing conversion from int to String is a common scenario when programming with Core Java.

Here are some quick reference example codes:

Convert using Integer.toString(int)

The Integer class has a static method that returns a String object representing the specified int parameter. Using this is an efficient solution.

Syntax

```
public static String toString(int i)
```

The argument ***i*** is converted and returned as a string instance. If the number is negative, the sign will be preserved.

Example

```
int number = -782;
String numberAsString = Integer.toString(number);
```

The code is equivalent to:

```
String numberAsString = "-782";
```

If you will try and search for solutions, this is one of the most popular ways of converting an int to String.

Convert using String.valueOf(int)

The String class has several static methods that can convert most primitive types to their String representation. This includes integers.

Example

```
int number = -782;
String numberAsString = String.valueOf(number);
```

or

```
String numberAsString = String.valueOf(-782);
```

This is also an efficient solution like the first option above. And because this is simple and efficient, it is also a very popular method for converting an int to String.

Convert using new Integer(int).toString()

Another alternative method is to create an instance of Integer class and then invoke it's toString() method. This is a little less efficient than the first two options shown above. This is because a new instance of Integer is created before conversion is performed.

Example

```
int number = -782;
Integer intInstance = new Integer(number);
String numberAsString = intInstance.toString();
```

We can shortened to:

```
int number = -782;
String numberAsString = new Integer(number).toString();
```

or just:

```
String numberAsString = new Integer(-782).toString();
```

If your variable is of primitive type (int), it is better to use Integer.toString(int) or String.valueOf(int). But if your variable is already an instance of Integer (wrapper class of the primitive type int), it is better to just invoke it's toString() method as shown above.

Convert using String.format()

String.format() is a new alternative that can be used for converting an Integer to String object. It is first introduced in Java 5 (JDK1.5) and has many cool features.

Syntax

```
public static String format(String format, Object... args)
```

There are many options on how to use this method. But for conversion purposes, here is a simple example:

Example

```
int number = -782;
String numberAsString = String.format ("%d", number);
```

And the variable *numberAsString* will have the value *"-782"*

If you are using Java 5 or higher, this is also a very simple alternative for converting an int to String object.

Convert using DecimalFormat

The class **java.text.DecimalFormat** is a class that formats a number to a String representation following a certain pattern.
Example

```
int number = 12345;
DecimalFormat decimalFormat = new DecimalFormat("#");
String numberAsString = decimalFormat.format(number);
System.out.println(numberAsString);
```

Will output

```
12345
```

This is an example that will convert to String but with commas

```
int number = 12345;
DecimalFormat decimalFormat = new DecimalFormat("#,##0");
String numberAsString = decimalFormat.format(number);
System.out.println(numberAsString);
```

Will output

```
12,345
```

This is my favorite from all the options shown in this post because of the level of control that you can do with the formatting. You can specify the number of decimal places and comma separator for readability.

Convert using StringBuffer or StringBuilder

StringBuffer is a class that is used to concatenate multiple values into a String. StringBuilder works similarly but not thread safe like StringBuffer. These two classes can be used to convert a Java Integer to String.

StringBuffer Example

```
int number = -782;
StringBuffer sb = new StringBuffer();
sb.append(number);
String numberAsString = sb.toString();
```

StringBuilder Example

```
int number = -782;
StringBuilder sb = new StringBuilder();
sb.append(number);
String numberAsString = sb.toString();
```

Shortened Examples

```
String numberAsString = new StringBuffer().append(-782).toString();
String numberAsString = new StringBuilder().append(-782).toString();
```

Quick Solution

If you want something simple that works but not elegant, you can use this code:

```
int number = -782;
String numberAsString = "" + number;
```

When you concatenate a String and an integer, the result is a String. Internally, the code is inefficient because there are intermediate objects that are created behind the scene.

Convert with special radix

All of the examples above uses the base (radix) 10. There are cases when we wish to convert a Java Integer to String but using another base. There are convenient methods to convert to binary, octal, and hexadecimal system. Arbitrary custom number system is also supported.

Examples

- Binary

```
int number = 255;
String binaryString = Integer.toBinaryString(number);
System.out.println(binaryString);
```

The output is

```
11111111
```

11111111 is the binary representation of the number 255.

- Octal

```
int number = 255;
String octalString = Integer.toOctalString(number);
System.out.println(octalString);
```

The output is

```
377
```

377 is the octal representation of the number 255.

- Hexadecimal

```
int number = 255;
String hexString = Integer.toHexString(number);
System.out.println(hexString);
```

The output is

```
ff
```

ff is the hexadecimal representation of the number 255.

- Custom Base/Radix
  We can use any other custom base/radix when converting an int to String. The sample shown below uses the base 7 number system.

```
int number = 255;
String customString = Integer.toString(number, 7);
System.out.println(customString);
```

The output is

```
513
```

513 is the representation of the number 255 when written in the base 7 system.