

# Java Convert String to int examples

BY CHAITANYA SINGH | FILED UNDER: [JAVA CONVERSION](#)

In this [tutorial](#) we will learn how to **convert a String to int in Java**. Even if a String is made up of digits like 1,2,3 etc, any arithmetic operation cannot be performed on it until it gets converted into an integer value. In this tutorial we will see two ways to convert String to int –

1. Java – Convert String to int using Integer.parseInt(String) method
2. Java – Convert String to int using Integer.valueOf(String) method

## 1. Java - Convert String to int using Integer.parseInt(String)

The parseInt() method of [Integer wrapper class](#) parses the string as signed integer number. This is how we do the conversion –

Here we have a String `str` with the value “1234”, the method `parseInt()` takes `str` as argument and returns the integer value after parsing.

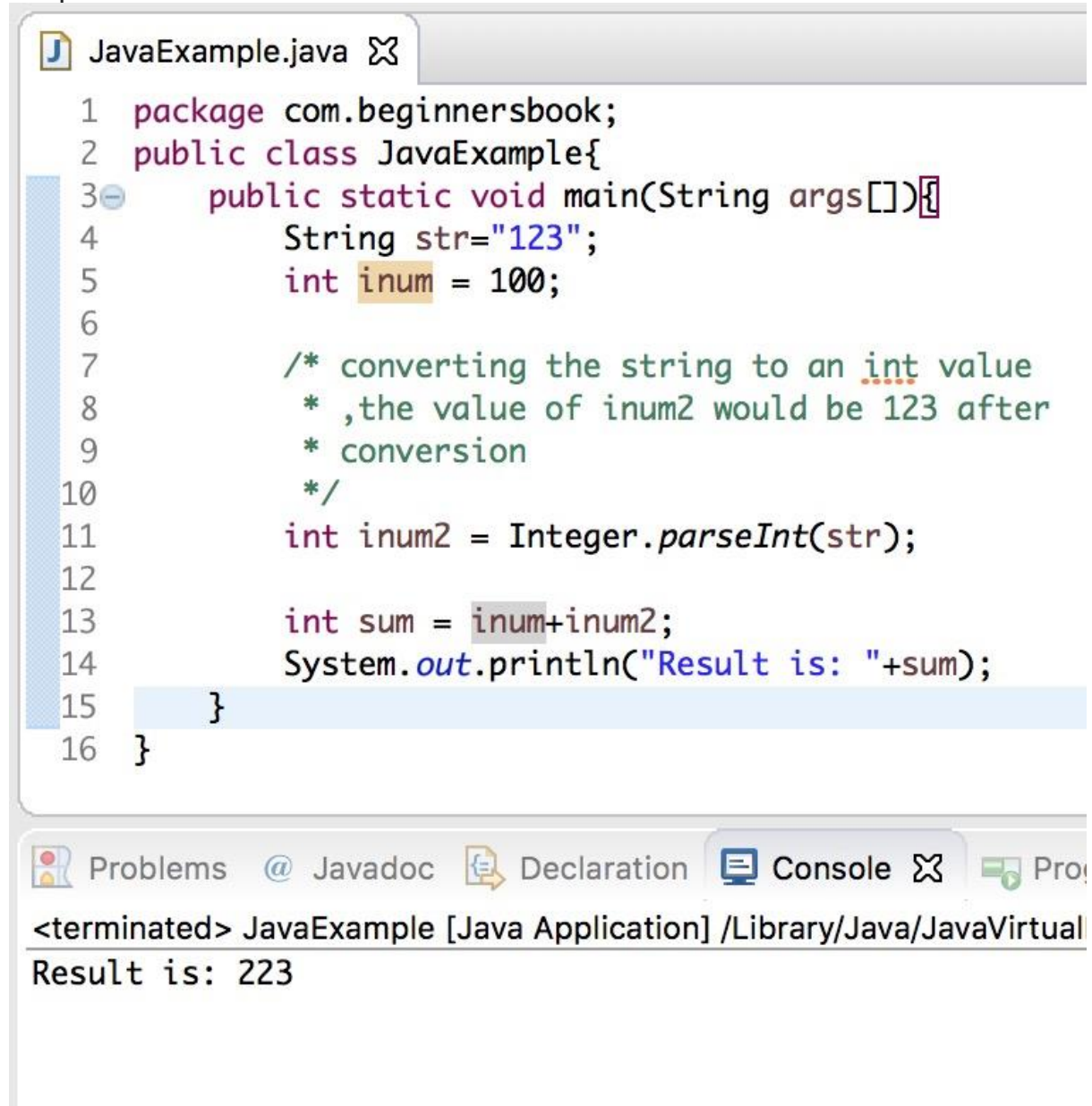
```
String str = "1234";  
int inum = Integer.parseInt(str);
```

Lets see the complete example –

## Java Convert String to int example using Integer.parseInt(String)

```
public class JavaExample{  
    public static void main(String args[]){  
        String str="123";  
        int inum = 100;  
  
        /* converting the string to an int value  
        * ,the value of inum2 would be 123 after  
        * conversion  
        */  
        int inum2 = Integer.parseInt(str);  
  
        int sum = inum+inum2;  
        System.out.println("Result is: "+sum);  
    }  
}
```

Output:



The screenshot shows an IDE window with a tab titled 'JavaExample.java'. The code is as follows:

```
1 package com.beginnersbook;
2 public class JavaExample{
3     public static void main(String args[]) {
4         String str="123";
5         int inum = 100;
6
7         /* converting the string to an int value
8          * ,the value of inum2 would be 123 after
9          * conversion
10        */
11        int inum2 = Integer.parseInt(str);
12
13        int sum = inum+inum2;
14        System.out.println("Result is: "+sum);
15    }
16 }
```

Below the code editor, there is a 'Console' tab. It shows the output of the program:

```
<terminated> JavaExample [Java Application] /Library/Java/JavaVirtual
Result is: 223
```

**Note:** All characters in the String must be digits, however the first character can be a minus '-' sign. For example:

```
String str="-1234";
int inum = Integer.parseInt(str);
The value of inum would be -1234
```

Integer.parseInt() throws NumberFormatException, if the String is not valid for conversion. For example:

```
String str="1122ab";
int num = Integer.valueOf(str);
```

This would throw `NumberFormatException`. you would see a compilation error like this:

```
Exception in thread "main" java.lang.NumberFormatException: For input string:
"1122ab"
    at java.lang.NumberFormatException.forInputString(Unknown Source)
    at java.lang.Integer.parseInt(Unknown Source)
    at java.lang.Integer.parseInt(Unknown Source)
```

Lets see the complete code for String to int conversion.

## 2. Java - Convert String to int using `Integer.valueOf(String)`

`Integer.valueOf(String)` works same as `Integer.parseInt(String)`. It also converts a String to int value. However there is a difference between `Integer.valueOf()` and `Integer.parseInt()`, the `valueOf(String)` method returns an object of Integer class whereas the `parseInt(String)` method returns a primitive int value. The output of the conversion would be same whichever method you choose. This is how it can be used:

```
String str="1122";
int inum = Integer.valueOf(str);
The value of inum would be 1122.
```

This method also allows first character of String to be a minus '-' sign.

```
String str="-1122";
int inum = Integer.valueOf(str);
Value of inum would be -1122.
```

Similar to the `parseInt(String)` method it also throws `NumberFormatException` when all the characters in the String are not digits. For example a String with value "11aa22" would throw an exception.

Lets see the complete code for conversion using this method.

### Java Convert String to int example using `Integer.valueOf(String)`

```
public class JavaExample{
    public static void main(String args[]){
        //String with negative sign
        String str="-234";

        //An int variable
        int inum = 110;
```

```

    /* Convert String to int in Java using valueOf() method
    * the value of variable inum2 would be negative after
    * conversion
    */
    int inum2 = Integer.valueOf(str);

    //Adding up inum and inum2
    int sum = inum+inum2;

    //displaying sum
    System.out.println("Result is: "+sum);
}
}

```

Output:

The screenshot shows an IDE window titled 'JavaExample.java'. The code is as follows:

```

1 package com.beginnersbook;
2 public class JavaExample{
3     public static void main(String args[]){
4         //String with negative sign
5         String str="-234";
6
7         //An int variable
8         int inum = 110;
9
10        /* Convert String to int in Java using valueOf() method
11        * the value of variable inum2 would be negative after
12        * conversion
13        */
14        int inum2 = Integer.valueOf(str);
15
16        //Adding up inum and inum2
17        int sum = inum+inum2;
18
19        //displaying sum
20        System.out.println("Result is: "+sum);
21    }
22 }

```

Below the code editor, the 'Console' tab is active, showing the output: `<terminated> JavaExample [Java Application] /Library/Java/JavaVirtualMachines/jdk-9.0.4/Contents/Home/bin/java -Djava.library.path= /Library/Java/JavaVirtualMachines/jdk-9.0.4/Contents/Home/bin/java -Djava.library.path= Result is: -124`

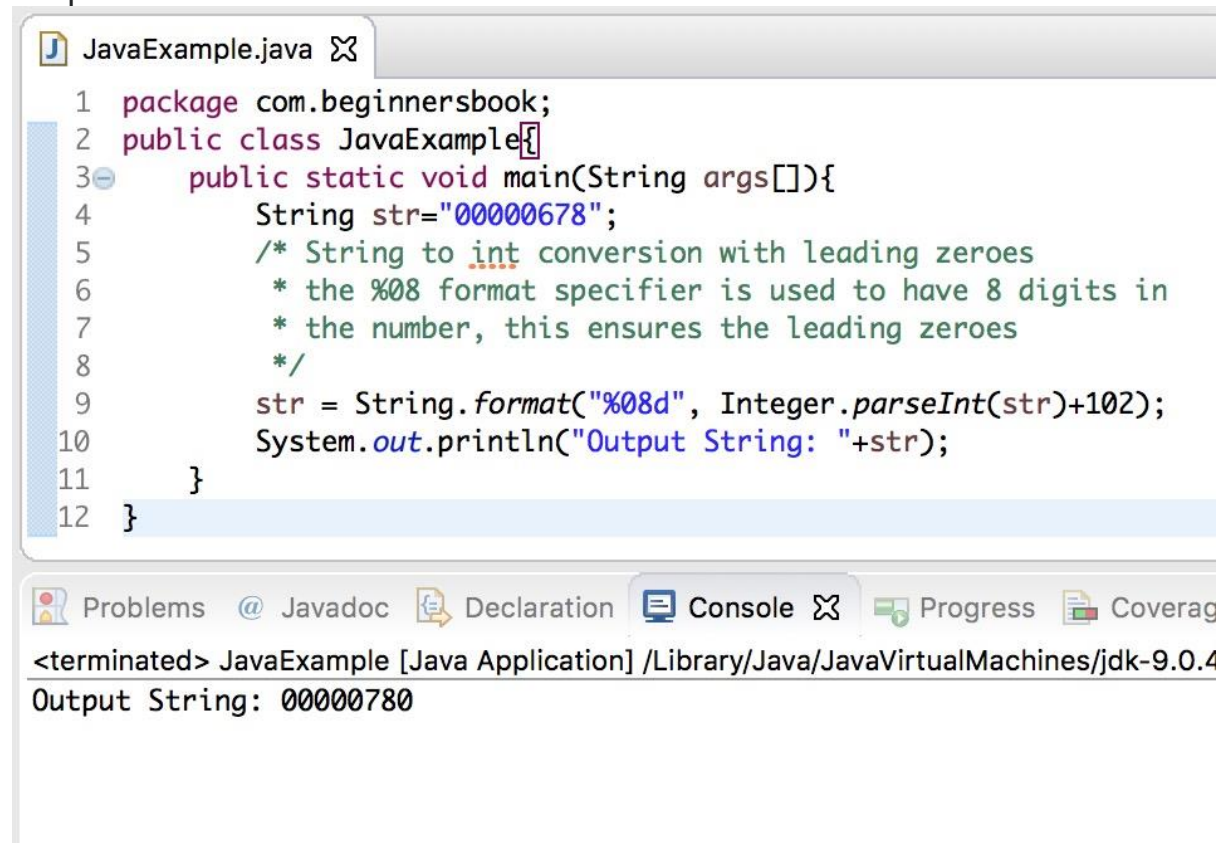
Lets see another interesting example of String to int conversion.

## Convert a String to int with leading zeroes

In this example, we have a string made up of digits with leading zeroes, we want to perform an arithmetic operation on that string retaining the leading zeroes. To do this we are converting the string to int and performing the arithmetic operation, later we are converting the output value to string using `format()` method.

```
public class JavaExample{
    public static void main(String args[]){
        String str="00000678";
        /* String to int conversion with leading zeroes
         * the %08 format specifier is used to have 8 digits in
         * the number, this ensures the leading zeroes
         */
        str = String.format("%08d", Integer.parseInt(str)+102);
        System.out.println("Output String: "+str);
    }
}
```

Output:



The screenshot shows an IDE window titled "JavaExample.java". The code is as follows:

```
1 package com.beginnersbook;
2 public class JavaExample{
3     public static void main(String args[]){
4         String str="00000678";
5         /* String to int conversion with leading zeroes
6          * the %08 format specifier is used to have 8 digits in
7          * the number, this ensures the leading zeroes
8          */
9         str = String.format("%08d", Integer.parseInt(str)+102);
10        System.out.println("Output String: "+str);
11    }
12 }
```

Below the code editor, the "Console" tab is active, showing the output:

```
<terminated> JavaExample [Java Application] /Library/Java/JavaVirtualMachines/jdk-9.0.4
Output String: 00000780
```