Splitter Class

We saw [Joiner](http://data-structure-learning.blogspot.com/2015/05/joiner-class-part-1.html) and [MapJoiner](http://data-structure-learning.blogspot.com/2015/05/joiner-class-part-2-mapjoiner.html) class in previous posts. Joiner class is used to concatenate strings together with a given delimiter. MapJoiner is a used to join Key-Value using a key value separator.

Splitter is opposite of Joiner class. It is used to split the string on the given delimiter.

Let us first see how to split the string normal using java String API split(..) method.

**public** **class** JavaStringSplit {

**public** **static** **void** main(String[] args) {

String testString = "Monday,Tuesday,,Wednesday,,Thursday,,";

String[] output = testString.split(",");

**for** (**int** i = 0; i < output.length; i++) {

System.***out***.print(output[i]);

}

}

}

The above example uses split(regex) method of String class which returns the String[].

Splitter class provides several other methods to make our task easier. We will see them after we see working of Splitter class.

First, Splitter class is immutable. So you must store and use the new splitter instance. Following is invalid.

/\*\*

\* on method returns new Splitter instance.

\* So this statement is good to do its job.

\* \*/

Splitter splitter= Splitter.*on*("|");

/\*\*

\* Now, Splitter class is immutable.

\* trimResults() method returns the

\* \*/

splitter.trimResults();

/\*\*

\* Results will contain empty elements.

\* \*/

Iterator<String> part=splitter.split("a|b|c|").iterator();

Now let us consider a string with delimiter as comma “,”

String testString = "Monday, Tuesday ,Wednesday ,Thursday ";

Simple Splitter class Demo

/\*\*

\* This method is used to Split the test String on

\* , as delimiter.

\* Remember the output might contains empty strings as

\* we are not trimming the results.

\* \*/

**public** **static** **void** splitterOn(String testString) {

Iterator<String> str = Splitter.*on*(",").split(testString).iterator();

System.***out***.println("Input String:: "+testString);

**while** (str.hasNext()) {

String temp=str.next();

System.***out***.println(temp);

}

}

Output

Input String:: Monday, Tuesday ,Wednesday ,Thursday

Monday

Tuesday

Wednesday

Thursday

Remember the results are not trimmed. For trimming the results you need to explicity call the trimResults() method. Remember trimResults() method returns the new instance of Splitter class.

/\*\*

\* This method is used to Split the test String on

\* , as delimiter.

\* Remember the output will still contain empty strings

\* even we are using trimResults() method

\* \*/

**public** **static** **void** splitterOnTrimResults(String testString) {

System.***out***.println("Input String:: "+testString);

Splitter splitter = Splitter.*on*(",").trimResults();

Iterator<String> iter = splitter.split(testString).iterator();

**while** (iter.hasNext()) {

System.***out***.println(iter.next());

}

}

Output:

Input String:: Monday, ,Tuesday ,Wednesday ,Thursday

Monday

Tuesday

Wednesday

Thursday

Now as we saw we can trim the result after the strings are split. But still we are not able to remove the empty strings. For that we can use the omitEmptyStrings(). Remember as trimResults() method omitEmptyStrings() method also returns new instance of Splitter class.

/\*\*

\* This method is used to Split the test String on

\* , as delimiter.

\* Remember the output will still contain empty strings

\* even we are using trimResults() method.

\*

\* To remove the empty strings we are using the omitEmptyStrings()

\* method.

\* \*/

**public** **static** **void** splitterOnTrimResults(String testString) {

System.***out***.println("Input String:: "+testString);

Splitter splitter = Splitter.*on*(",").trimResults().omitEmptyStrings();

Iterator<String> iter = splitter.split(testString).iterator();

**while** (iter.hasNext()) {

System.***out***.println(iter.next());

}

}

Output:

Input String:: Monday, ,Tuesday ,Wednesday ,Thursday

Monday

Tuesday

Wednesday

Thursday

Now let’s say that you want to split the string for some number of times. That is it splits the string till the limit is not reached. Limit also specified number of items returned by Iterator.

Assume the below code:

Splitter.*on*(',').limit(3).split("a,b,c,d");

It returns the Iterable containing ["a", "b", "c,d"]

Code:

/\*\*

\* This method is used to Split the test String on

\* , as delimiter.

\* Remember the output will still contain empty strings

\* even we are using trimResults() method.

\*

\* We are also using limit method that will split the string

\* the number of time specified in parameter.

\* \*/

**public** **static** **void** splitterOnLimitDemo(String testString) {

System.***out***.println("Input String:: "+testString);

Splitter splitter = Splitter.*on*(",").trimResults().limit(3);

Iterator<String> iter = splitter.split(testString).iterator();

**while** (iter.hasNext()) {

System.***out***.println(iter.next());

}

}

Output:

Input String:: Monday, Tuesday ,Wednesday ,Thursday

Monday

Tuesday

Wednesday ,Thursday

Let us try another input.

Output

Input String:: Monday, ,Tuesday ,Wednesday ,Thursday

Monday

Tuesday ,Wednesday ,Thursday

The output contains empty string. So now we can use omitEmptyStrings() method to remove empty strings from output.

/\*\*

\* This method is used to Split the test String on

\* , as delimiter.

\* Remember the output will still contain empty strings

\* even we are using trimResults() method.

\*

\* To remove the empty strings we are using the omitEmptyStrings()

\* method.

\*

\* We are also using limit method that will split the string

\* the number of time specified in parameter.

\* \*/

**public** **static** **void** splitterOnLimitDemo(String testString) {

System.***out***.println("Input String:: "+testString);

Splitter splitter = Splitter.*on*(",").trimResults().limit(3).omitEmptyStrings();

Iterator<String> iter = splitter.split(testString).iterator();

**while** (iter.hasNext()) {

System.***out***.println(iter.next());

}

}

Output

Input String:: Monday, ,Tuesday ,Wednesday ,Thursday

Monday

Tuesday

Wednesday ,Thursday

In next post we will discuss the MapSplitter static class of Splitter class.

For additional posts on [Google Guava](http://data-structure-learning.blogspot.com/p/google-guava-1.html) [click here](http://data-structure-learning.blogspot.com/p/google-guava-1.html).

I have also written posts on [Java Collections Framework](http://data-structure-learning.blogspot.com/p/java-collections.html), [Java Collections Interview Questions and Answers](http://data-structure-learning.blogspot.com/p/java-collections_26.html), [Java 8](http://data-structure-learning.blogspot.com/p/functional-programming-in-java.html), [StringUtils class of SpringFramework](http://data-structure-learning.blogspot.com/p/stringutils-classs-methods-description.html) and [others](http://data-structure-learning.blogspot.com/p/main-page.html). Please read it. You feedback is welcomed.