User-defined Counters

While processing information/data using MapReduce job, it is a challenge to monitor the progress of parallel threads running across nodes of distributed clusters. Moreover, it is also complicated to distinguish between the data that has been processed and the data which is yet to be processed. The MapReduce Framework offers a provision of user-defined Counters, which can be effectively utilized to monitor the progress of data across nodes of distributed clusters.

Now, let's use Hadoop Counters to identify the number of complaints pertaining to debt collection, mortgage and other categories in the consumer complaints dataset. The following job code, would help us to accomplish this.

```
?
```

```
1  package com.evoke.bigdata.mr.complaint;
2  import org.apache.hadoop.conf.Configured;
3  import org.apache.hadoop.fs.Path;
4  import org.apache.hadoop.io.IntWritable;
5  import org.apache.hadoop.io.Text;
6  import org.apache.hadoop.mapred.FileInputFormat;
7  import org.apache.hadoop.mapred.FileOutputFormat;
8  import org.apache.hadoop.mapred.JobClient;
9  import org.apache.hadoop.mapred.JobConf;
10 import org.apache.hadoop.mapred.RunningJob;
11 import org.apache.hadoop.util.Tool;
12 import org.apache.hadoop.util.ToolRunner;
13 public class ComplaintCounter extends Configured implements Tool {
14 @Override
15 public int run(String[] args) throws Exception {
16 String input, output;
17 if(args.length == 2) {
18  input = args[0];
19  output = args[1];
20 } else {
   input = &amp;quot;your-input-dir&amp;quot;;
   output = &amp;quot;your-output-dir&amp;quot;;
 }
```

```
21  JobConf conf = new JobConf(getConf(), ComplaintCounter.class);
22  conf.setJobName(this.getClass().getName());
23  FileInputFormat.setInputPaths(conf, new Path(input));
24  FileOutputFormat.setOutputPath(conf, new Path(output));
25  conf.setMapperClass(ComplaintCounterMapper.class);
26  conf.setMapOutputKeyClass(Text.class);
27  conf.setMapOutputValueClass(IntWritable.class);
28  conf.setOutputKeyClass(Text.class);
29  conf.setOutputValueClass(IntWritable.class);
30  conf.setNumReduceTasks(0);
31  RunningJob job = JobClient.runJob(conf);
32  long debt = job.getCounters().findCounter(&quot;Debt-Counter&quot;, &quot;debt&quot
33  long mortage = job.getCounters().findCounter(&quot;Mortgage-Counter&quot;,
    &quot;mortgage&quot;).getValue();
34  long other = job.getCounters().findCounter(&quot;Other-Counter&quot;, &quot;other&q
35  System.out.println(&quot;Debt = &quot; + debt);
36  System.out.println(&quot;Mortgage = &quot; + mortage);
37  System.out.println(&quot;OTHER = &quot; + other);
38  return 0;
39  }
40  public static void main(String[] args) throws Exception {
41   int exitCode = ToolRunner.run(new ComplaintCounter(), args);
42   System.exit(exitCode);
    }
    }
```

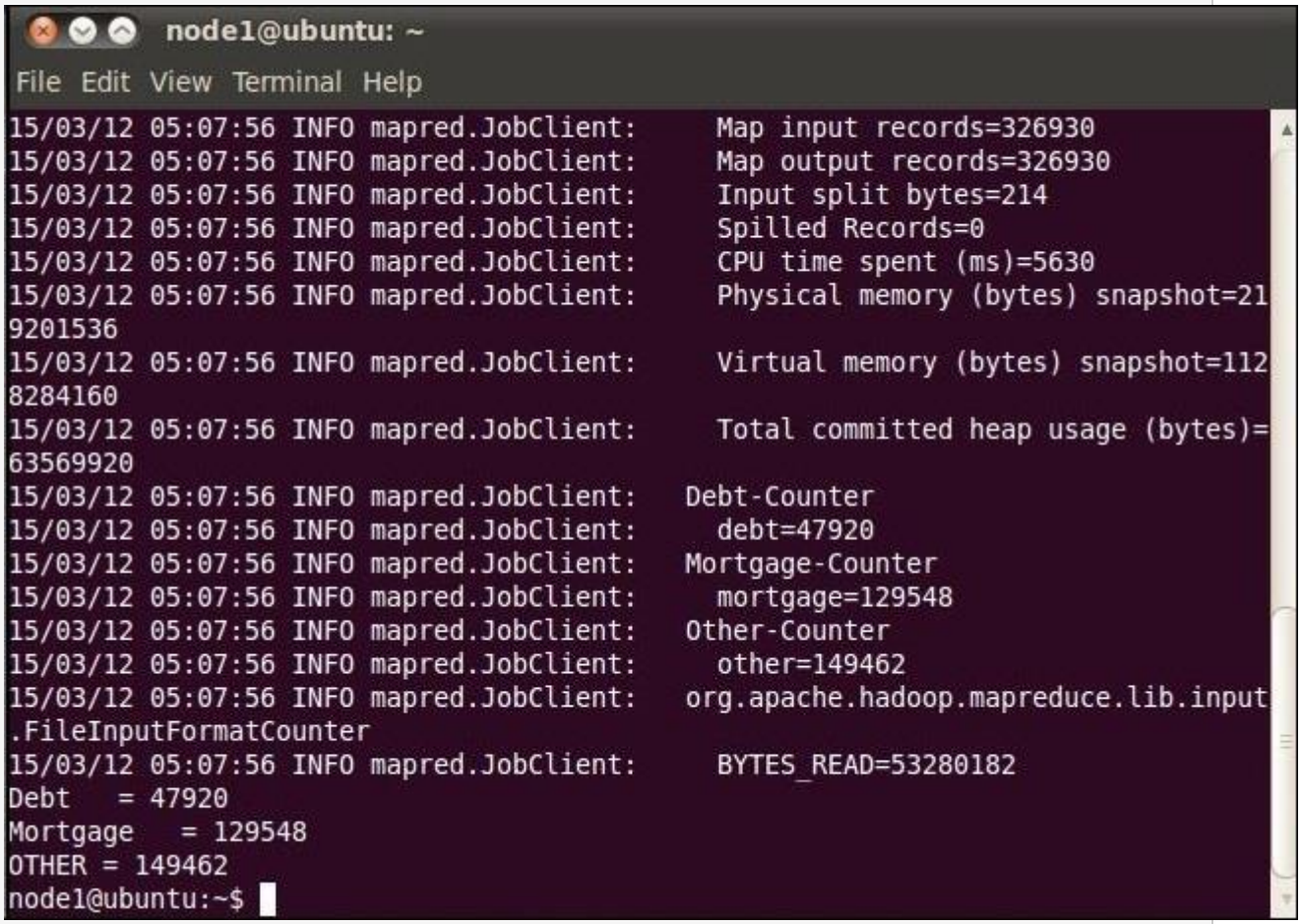
The above Java code contains Map-only job and it displays the values of debt collection, mortgage and other occurrences.

Below is the map code, which is used in the above job code. The belo w map code scans through each line of the dataset and increments deb t collection, mortgage and other occurrences.

?

```
1
2  package com.evoke.bigdata.mr.complaint;
3  import java.io.IOException;
4  import org.apache.hadoop.io.IntWritable;
5  import org.apache.hadoop.io.LongWritable;
6  import org.apache.hadoop.io.Text;
7  import org.apache.hadoop.mapred.MapReduceBase;
8  import org.apache.hadoop.mapred.Mapper;
9  import org.apache.hadoop.mapred.OutputCollector;
   import org.apache.hadoop.mapred.Reporter;
10
11 public class ComplaintCounterMapper extends MapReduceBase implements
12  Mapper&amp;amp;lt;LongWritable, Text, Text, IntWritable&amp;amp;gt; {
13 public void map(LongWritable key, Text value,
14 OutputCollector&amp;amp;lt;Text, IntWritable&amp;amp;gt; output, Reporter reporter)
15 throws IOException {
16 String[] fields = value.toString().split(&quot;,&quot;);
17 if (fields.length &amp;amp;gt; 1) {
18  String fileName = fields[1].toLowerCase();
19  if (fileName.equals(&quot;debt collection&quot;)) {
20  reporter.getCounter(&quot;Debt-Counter&quot;, &quot;debt&quot;).increment(1);
21  } else if (fileName.equals(&quot;mortgage&quot;)) {
22  reporter.getCounter(&quot;Mortgage-Counter&quot;, &quot;mortgage&quot;).increment(
23  } else {
24  reporter.getCounter(&quot;Other-Counter&quot;, &quot;other&quot;).increment(1);
25  }
26  output.collect(new Text(fileName), new IntWritable(1));
27  }
   }
28
29
```

Here’s the output, when the above map code is executed:



Hadoop offers Job Tracker, an UI tool to determine the status and st
atistics of all jobs. Using the job tracker UI, developers can view
the Counters that have been created (refer the below screenshot for
more clarity).

	Counter
File System Counters	FILE: Number of bytes read
	FILE: Number of bytes written
	FILE: Number of read operations
	FILE: Number of large read operations
	FILE: Number of write operations
	HDFS: Number of bytes read
	HDFS: Number of bytes written
	HDFS: Number of read operations
	HDFS: Number of large read operations
	HDFS: Number of write operations
Job Counters	Launched map tasks
	Data-local map tasks
	Total time spent by all maps in occupied slots (ms)
	Total time spent by all reduces in occupied slots (ms)
	Total time spent by all maps waiting after reserving slots (ms)
	Total time spent by all reduces waiting after reserving slots (m
Map-Reduce Framework	Map input records
	Map output records
	Input split bytes
	Spilled Records
	CPU time spent (ms)
	Physical memory (bytes) snapshot
	Virtual memory (bytes) snapshot
	Total committed heap usage (bytes)
Debt-Counter	debt
Mortgage-Counter	mortgage
Other-Counter	other
org.apache.hadoop.mapreduce.lib.input.FileInputFormatCounter	BYTES_READ

Conclusion

Hadoop is considered as one of the best parallel processing architectures around, it offers some really amazing features. Counter is one such feature in Hadoop that allows developers to track the status of data that has been processed. Additionally, it also helps developers

```
identify junk data and mark it for identification. Hope this post he
lped developers gain some knowledge on Hadoop Counters, in my next p
ost, I would be talking about Hadoop Combiners.
```