Exchange Reports Summary

Amit Kumar Sharma

22July 2018

Contents

[Technology Stack 3](#_Toc520013990)

[Why Univocity parser 3](#_Toc520013991)

[How it works 4](#_Toc520013992)

[How to use it for other reports 4](#_Toc520013993)

[How to compile and run it 4](#_Toc520013994)

[Location of input and output file 5](#_Toc520013995)

[Logs 5](#_Toc520013996)

[Output.csv 5](#_Toc520013997)

[Troubleshooting 5](#_Toc520013998)

[Classes 5](#_Toc520013999)

[Resources 7](#_Toc520014000)

[Unit Tests 7](#_Toc520014001)

[Run Unit Tests 7](#_Toc520014002)

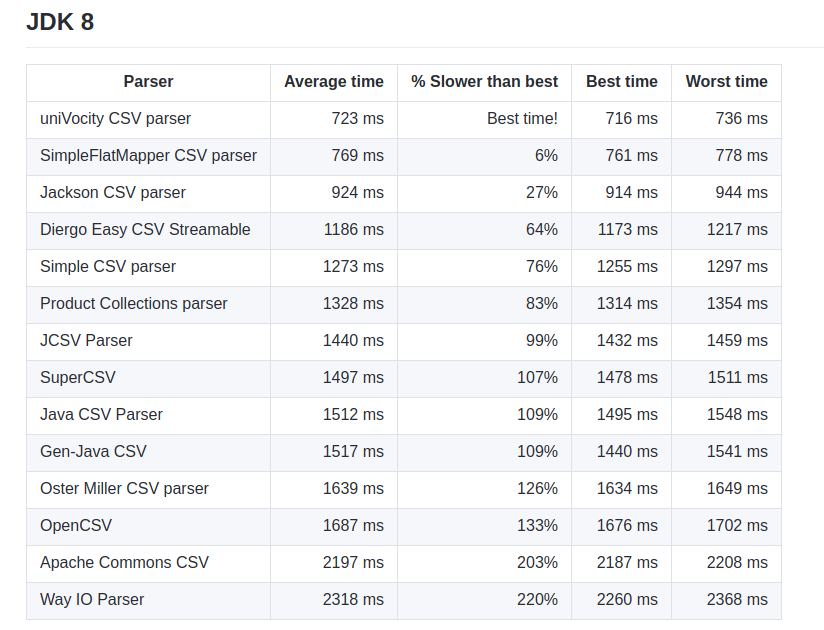
# Technology Stack

We have used following tools to provide the solution

|  |  |  |
| --- | --- | --- |
| Software | Version | Comments, if any |
| Spring boot | 2.0.3 Release |  |
| Java | 1.8 |  |
| Univocity | 2.72 | Parser used for file input/output |
| Junit | 4 |  |
| Maven | 3.5.2 |  |
| Operating System | Ubuntu 18.04 |  |

# Why Univocity parser

This parser is the best performance-based parser, comparison [here](https://github.com/uniVocity/csv-parsers-comparison).



# How it works

Solution is profile based as same code can be used with minimum changes to support multiple files parsing and reporting.

Right now, the active profile is singapore, so this flow takes a different flow and produces desired report for singapore.

# How to use it for other reports

A sample profile has been set for the hongkong also, just for the ease same files are used but it can show how same code can be used to different report generation.

# How to compile and run it

Follow these steps

* unzip summary.zip
* cd ${HOME}/summary
* mvn clean install
* cp {from actual source}/Input.txt
* java -jar target/summary1-0.0.1-SNAPSHOT.jar ,or
  + java -jar target/summary1-0.0.1-SNAPSHOT.jar --input.filename=/home/${USER}/Input.txt --output.filename=/home/${USER}/Output.csv

After successful run output file will be created in ${HOME}/summary

# Location of input and output file

For the existing settings both input file and output file are kept at the project root folder. Location of both input file and output file can be managed from

* application-profile.properties or application.properties
  + input.filename=Input.txt
  + output.filename=Output.csv
* or from commandline as
  + java -jar target/summary1-0.0.1-SNAPSHOT.jar --input.filename=/home/${USER}/Input.txt --output.filename=/home/${USER}/Output.csv

# Logs

In the existing flow logs are being created at cd ${HOME}/summary/logs directory and the log name is report.log

This can be overridden by the application.properties

* logging.file=logs/report.log

# Output.csv

For the existing Input.txt the contents of Output.csv are

* Client\_Information,Product\_Information,Total\_Transaction\_Amount
* CL-4321-3-1,CME-FU-N1-20100910, -79.00
* CL-1234-3-1,CME-FU-N1-20100910, 285.00
* CL-4321-2-1,SGX-FU-NK-20100910, 46.00
* CL-1234-3-1,CME-FU-NK.-20100910,-215.00
* CL-1234-2-1,SGX-FU-NK-20100910, -52.00

# Troubleshooting

Right now, the code is running in debug mode.

Modes can be handled from application.properties

* logging.level.com.exchange.summary.report=DEBUG

# Classes

* Model
  + TradeImpl – Model class for the parsed file
  + ClientSummary – Model class for the summarized data
* Parser
  + FileParser – Parameterised parser which takes input file and gives model class list as output
* Configuration
  + TradeConfiguration – Acts as a manager

SpringMainApplication



# Resources

It has following files

* application.properties
* application-singapore.properties
* application-hongkong.properties

# Unit Tests

Unit are there for all the classes and modules.

Limited unit tests for Models as most of them are getters and setters

* Model
  + TradeImplTests
  + ClientSummaryTest
* Parser
  + FileParserTests
* Configuration
  + TradeConfigurationTests

SpringMainApplicationTests

# Run Unit Tests

mvn tests