Java: Log processing (kasutame hetkel seda)

Read and analyse server access logs and present some statistics.

Introduction

Here is a short snippet of a log file. The goal of the assignment is to create a program that parses the log file and extracts, aggregates and presents the data from it. The log file contains request durations. The file format is best explained with the following examples:

```
2015-08-19 00:00:01,049 (http--0.0.0.0-28080-405) [] /checkSession.do in 187
2015-08-19 00:00:01,963 (http--0.0.0.0-28080-245) [] /checkSession.do in 113
2015-08-19 00:00:02,814 (http--0.0.0.0-28080-245) [CUST:CUS5T27233] /substypechange.do?msisdn=300501633574 in 17
2015-08-19 00:00:03,260 (http--0.0.0.0-28080-245) [CUST:CUS5T27233] /mainContent.do?action=TOOLS&contentId=main_tools in 5
```

It contains:

- date
- timestamp
- thread-id (in brackets)
- optional user context (in square brackets)
- URI + query string
- string "in"
- request duration in milliseconds

…and…

```
2015-08-19 00:04:45,212 (http--0.0.0.0-28080-405) [] updateSubscriptionFromBackend 300445599231 in 203
2015-08-19 00:04:45,259 (http--0.0.0.0-28080-405) [ASP CUST:CUS5T27233] getPermission  300445599231 in 32
```

which contains:

- date
- timestamp
- thread-id (in brackets)
- optional user context (in square brackets)
- requested resource name (one string)
- data payload elements for resource (0..n elements)
- string "in"
- request duration in milliseconds

What to do?

Required interface:

1. The program needs to run from command line and must take log filename as an argument.
2. The program must display help information documenting how to use it when running with **-h** argument

Sample of expected command line:

```
aks-mbp:dist ak$ java -Xmx512m -jar assignment.jar timing.log 10
```

Functional requirements:

1. Print out top **n** (exact value of n is passed as program argument) resources with the highest average request duration.
2. Draw histogram of hourly number of requests.
3. Print out number of (milli)seconds your program ran.

Compilation and test environment

Do not send compiled code. We will compile the code ourselves. Prepare an Ant build file which default target creates **jar packed** executable to **dist** subfolder. We will compile and run the program in following environment:

1. Java 8 SE
2. Ant 1.9

Ask (and justify) if you want to use any other specific environment. The solution must be self containing - We prefer that package managers do not download "internet" to our test environment. If you use external libraries in your solution ship them with your code.

Notes

- When calculating the average request duration. URI + query string != resource. Decide if any and which part of the query string might define unique resource and group the requests based on justified logic. Be prepared to explain why did you use this particular grouping.
- Do not expect that the sample file provided contains samples of **all** possible resources.
- Assume that test files might be much bigger than provided sample
- Assume we run the code with `-Xmx512m` memory limit

Additional Material

timing.log