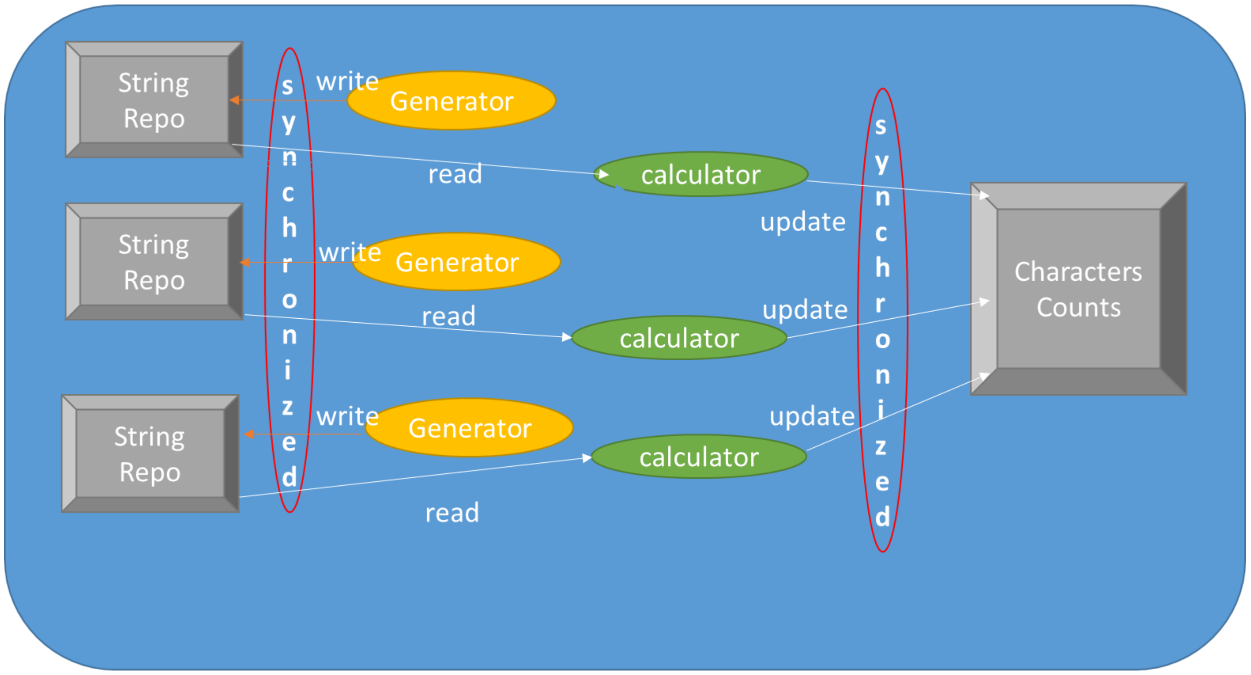
Table of Contents

Solution overview



Classes Description

Character Statistics processor

- Start to build corresponding number of String generator threads using the 2 dimension array List to overcome the Integer size of Array List then if the length is great than Integer Max number .
- Start all created Threads to build the Strings repositories.
- Program will wait till all threads done their work.
- Start to create the clustered calculators threads.
- Keep building the statistics per character using the shared Hashmap between threads.
- Once the Calculators threads done then print the final statistics and exit;

String Generation:

- Program will start to build random Strings to fill the 2 dimension Array List.
- Start to generate Random Number between 1000 and 10000 as per requirements document.
- This Random number will be the String length.
- Start to generate random number between 97 and 122 to be mapped to character and fill the String by character till the length.

Statistic Calculator

This class responsible for character statistics calculation.

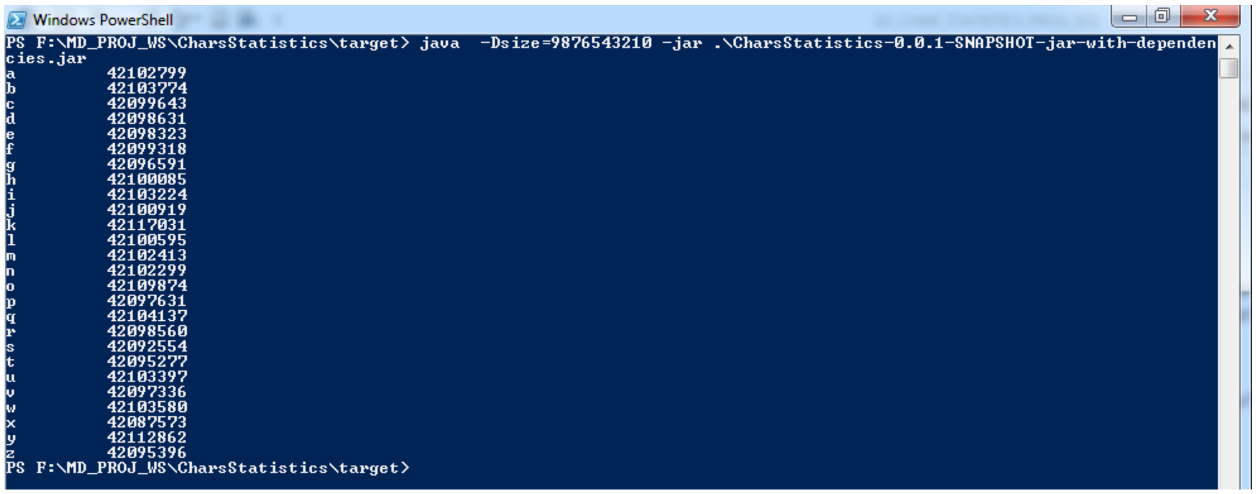
- Go through the shared repositories to retrieve the Strings.
- Go through each string to update the shared Statistics Hashmap for each character.

How to run and Test the solution:

`You can start the Application by doing the following`

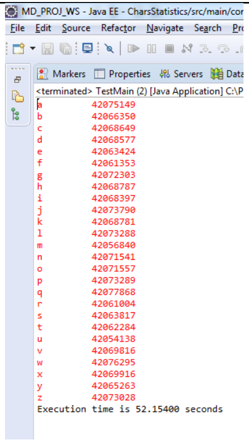
Copy the log4j.properties the target folder and modify the log4j.appender.DAILY.File property then run the following command sample:

```
Java -Dsize=1234 -jar  CharsStatistics-0.0.1-SNAPSHOT-jar-with-dependencies.jar
```

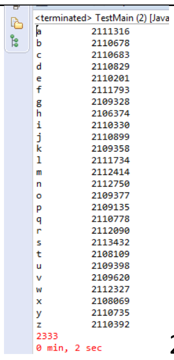
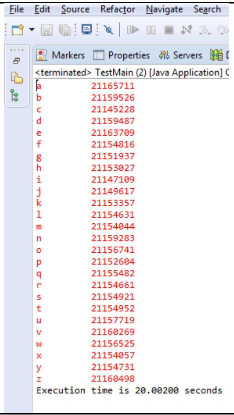


```
PS F:\MD_PROJ_WS\CharsStatistics\target> java  -Dsize=9876543210 -jar .\CharsStatistics-0.0.1-SNAPSHOT-jar-with-dependen
cies.jar
a             42102799
b             42103774
c             42099643
d             42098631
e             42098323
f             42099318
g             42096591
h             42100085
i             42103224
j             42100919
k             42117031
l             42100595
m             42102413
n             42102299
o             42109874
p             42097631
q             42104137
r             42098560
s             42092554
t             42095277
u             42103397
v             42097336
w             42103580
x             42087573
y             42112862
z             42095396
PS F:\MD_PROJ_WS\CharsStatistics\target>
```

Time table

Input Amount	Time consumption
9876543210l	 53 seconds

```
<terminated> TestMain (2) [Java Application] C:\P
a          42075149
b          42066350
c          42068649
d          42068577
e          42063424
f          42061353
g          42072303
h          42068787
i          42068397
j          42073790
k          42068781
l          42073288
m          42056840
n          42071541
o          42071557
p          42073289
q          42077868
r          42061004
s          42063817
t          42062284
u          42054138
v          42069816
w          42076295
x          42069916
y          42065263
z          42073028
Execution time is 52.15400 seconds
```

10000	 2 seconds
100000	

```
<terminated> TestMain (2) [Java
a         2111316
b         2110678
c         2110683
d         2110829
e         2110201
f         2111793
g         2109328
h         2106374
i         2110330
j         2110899
k         2109358
l         2111734
m         2112414
n         2112750
o         2109377
p         2109135
q         2110778
r         2112090
s         2113432
t         2108109
u         2109398
v         2109620
w         2112327
x         2108069
y         2110735
z         2110392
2333
0 min, 2 sec
```

```
<terminated> TestMain (2) [Java Application]
a         21165711
b         21159526
c         21145228
d         21159487
e         21163709
f         21154816
g         21151937
h         21153027
i         21147109
j         21149617
k         21153357
l         21154631
m         21154044
n         21159283
o         21156741
p         21152604
q         21155482
r         21154661
s         21154921
t         21154952
u         21157719
v         21160269
w         21156525
x         21154057
y         21154731
z         21160498
Execution time is 20.00200 seconds
```

Implementation Idea

The solution could be used to produce random String we may using it for network discovery to build random IP or , in fetching the big data on the Social media to check the trend of special words (ex: we can use it to count number of “stc” word in twitter data.

Future enhancement

- Performance Testing.
- Check optimal number of threads.
- Configure the length of word limits.
- Make the number of threads more dynamic based on the lunching options.