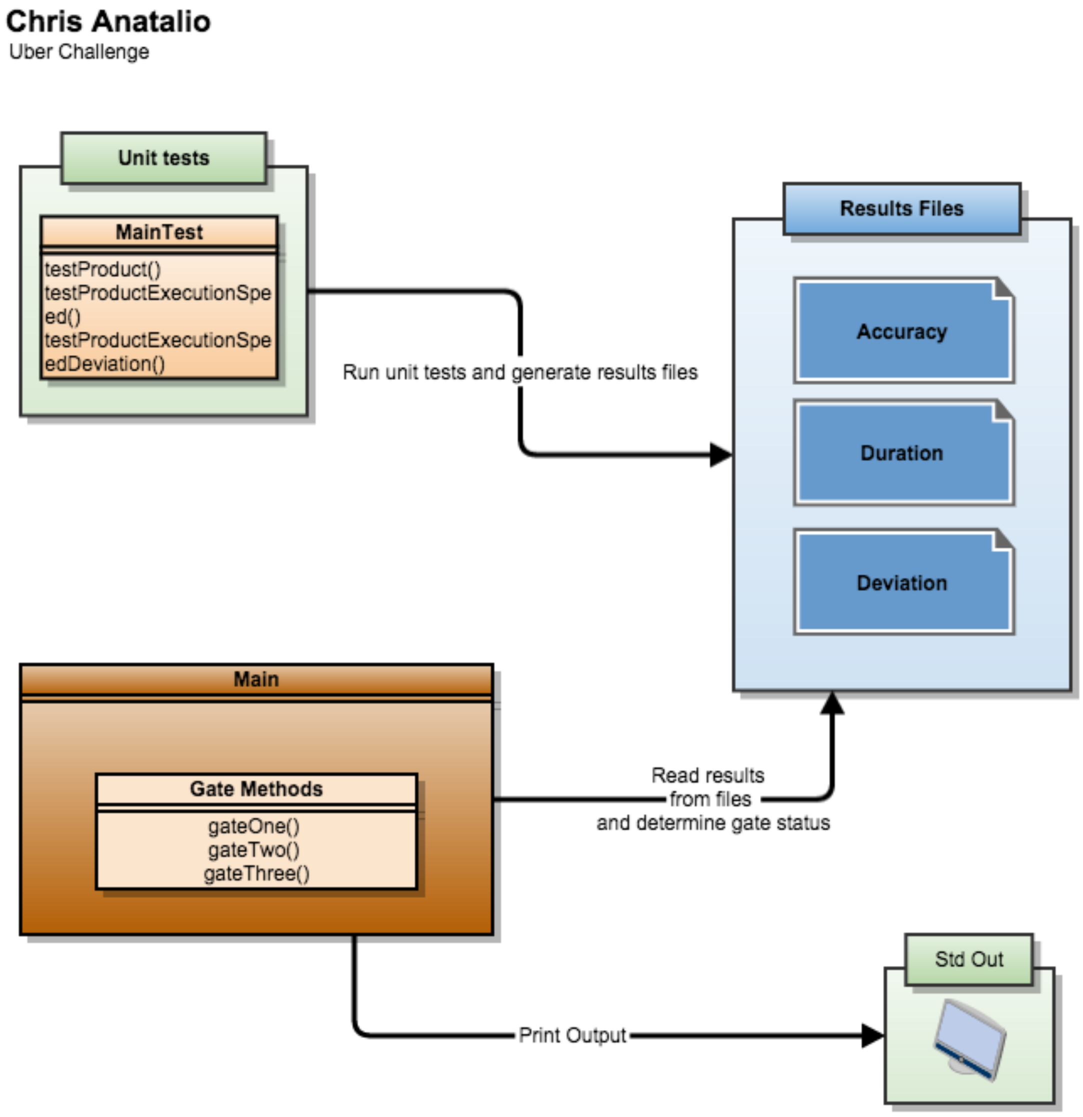


Uber Code Challenge

Prepared by: Chris Anatalio

January 21, 2016

SUMMARY

Implementation

Created Product class to act as service being tested.

Unit tests execution adds results to HashMap which is streamed through and the Files lib is used to create and update the file. Existing files are cleaned out before writing.

System nano time is used for measuring method execution time.

Parallel streams are used to simulate async method invocations.

Gates are implemented as methods in Main class. Gates use Files lib to read out from text files created from unit testing. Gates are then call in the main method

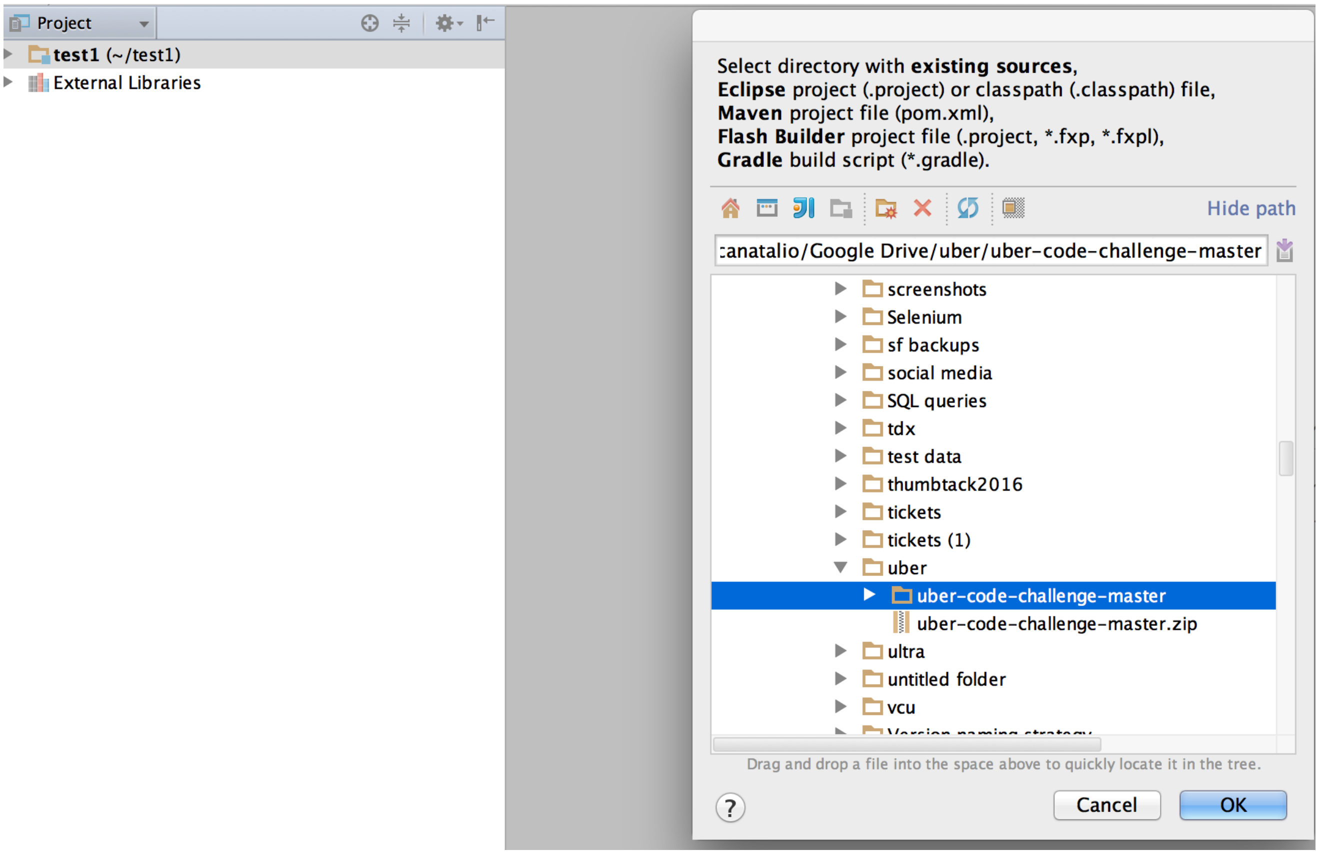
Run instructions for Command Line

Was not able to get project running on command line.

Install instructions for IDE

Unzip project

Import project and select Java language level 8

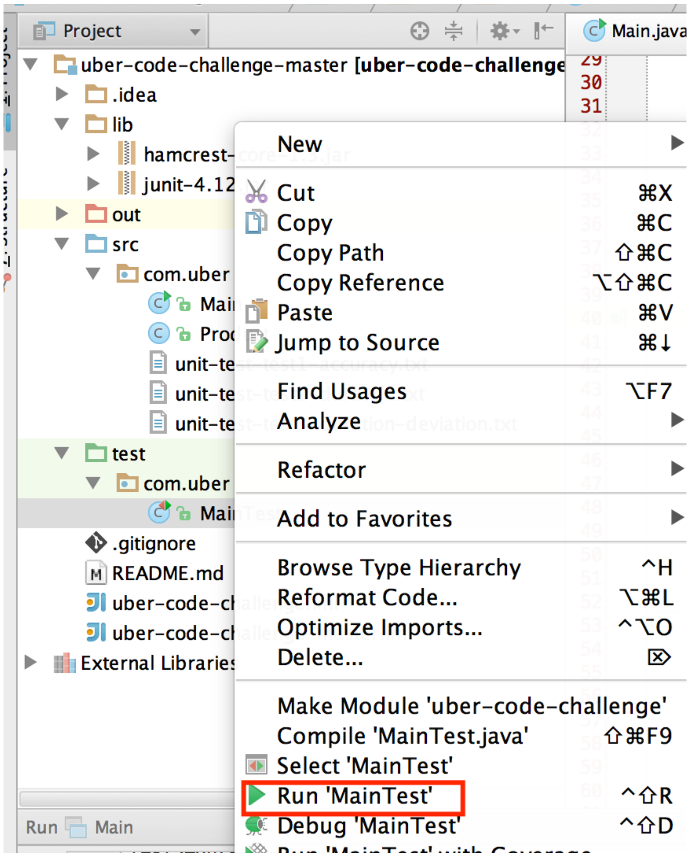


Select directory with **existing sources**,
Eclipse project (.project) or classpath (.classpath) file,
Maven project file (pom.xml),
Flash Builder project file (.project, *.fxp, *.fxpl),
Gradle build script (*.gradle).

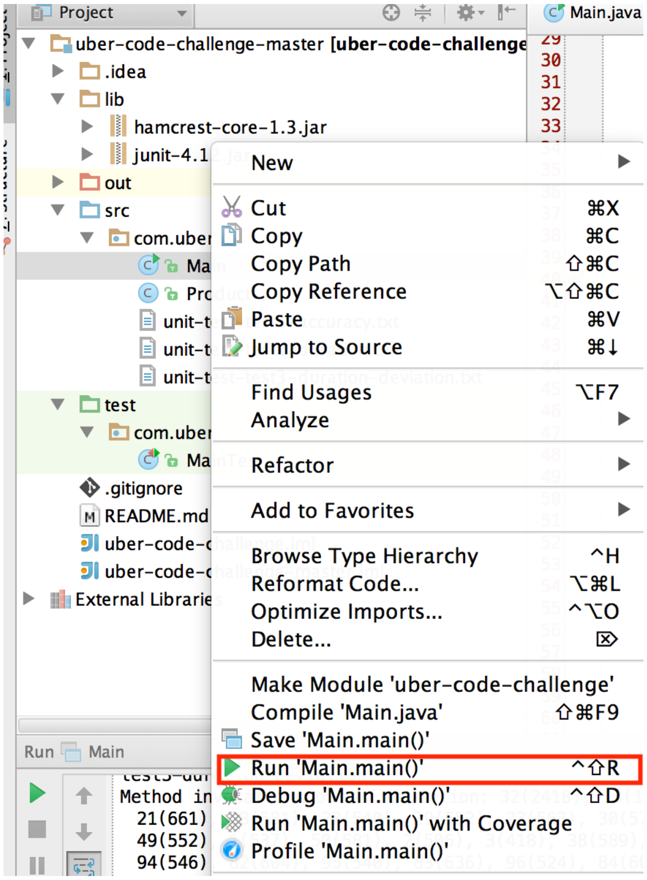
:anatalio/Google Drive/uber/uber-code-challenge-master

Drag and drop a file into the space above to quickly locate it in the tree.

Run tests: test -> com -> uber -> Run MainTest


Project
Main.java
uber-code-challenge-master [uber-code-challenge
.idea
lib
hamcrest-
junit-4.12
out
src
com.uber
Mai
Proc
unit-te
unit-te
unit-te
test
com.uber
Mai
.gitignore
README.md
uber-code-ch
uber-code-ch
External Libraries
Run
Main
New
Cut ⌘X
Copy ⌘C
Copy Path ⇧⌘C
Copy Reference ⌥⇧⌘C
Paste ⌘V
Jump to Source ⌘↓
Find Usages ⌥F7
Analyze
Refactor
Add to Favorites
Browse Type Hierarchy ^H
Reformat Code... ⌥⌘L
Optimize Imports... ^⌥O
Delete... ⌦
Make Module 'uber-code-challenge'
Compile 'MainTest.java' ⇧⌘F9
Select 'MainTest'
Run 'MainTest' ^⇧R
Debug 'MainTest' ^⇧D
Run 'MainTest' with Coverage

Run application: src -> com -> uber -> Run Main



OUTPUT

Example Output from Running Application

```
Actual Line Count = 2
Gate One Results = true

Method ExecutionTime = 2235
Max Method ExecutionTime = 10000
Gate Two Results = true
```

```
test3-duration-deviation
Method invocations out of deviation: 32(2418), 13(1262), 33(911), 14(735), 15(569), 34(679), 16(662), 35(629),
17(557), 36(526), 18(573), 37(553), 19(632), 26(760), 20(558), 27(609), 21(661), 28(581), 22(540), 29(552),
23(563), 30(575), 24(537), 31(562), 25(580), 7(486), 44(572), 8(563), 45(567), 9(549), 46(587), 10(523), 47(579),
11(546), 48(530), 12(537), 49(552), 1(537), 50(581), 2(506), 3(418), 38(589), 4(527), 5(518), 6(444), 39(15359),
40(543), 41(593), 42(474), 88(568), 43(469), 89(563), 90(451), 91(437), 92(416), 93(428), 94(546), 82(604),
95(540), 83(636), 96(524), 84(601), 97(534), 85(580), 98(521), 86(569), 99(547), 87(537), 100(541), 76(524),
57(577), 77(1275), 58(487), 78(567), 59(520), 79(561), 60(535), 80(543), 61(532), 81(579), 62(553), 51(513),
52(428), 69(580), 53(436), 54(542), 70(581), 55(428), 71(569), 56(517), 72(556), 73(477), 74(510), 75(471),
63(1812), 64(466), 65(438), 66(422), 67(448), 68(452),
warning method invocation deviation > 2%
Gate Three Results = true


====

Ready to deploy!
```

Example Unit Test run Output

```
Method execution duration: 2084287 nano seconds

results.keySet() = [test1-accuracy]

test1-accuracy: path = src/com/uber/unit-test-test1-accuracy.txt - result: success

Method execution duration: 1859 nano seconds


upperBoundExecutionSpeed = 1897

lowerBoundExecutionSpeed = 1821

results.keySet() = [test2-duration]

test2-duration: path = src/com/uber/unit-test-test2-duration.txt - result: 1859

Method execution duration: 1859 nano seconds
```

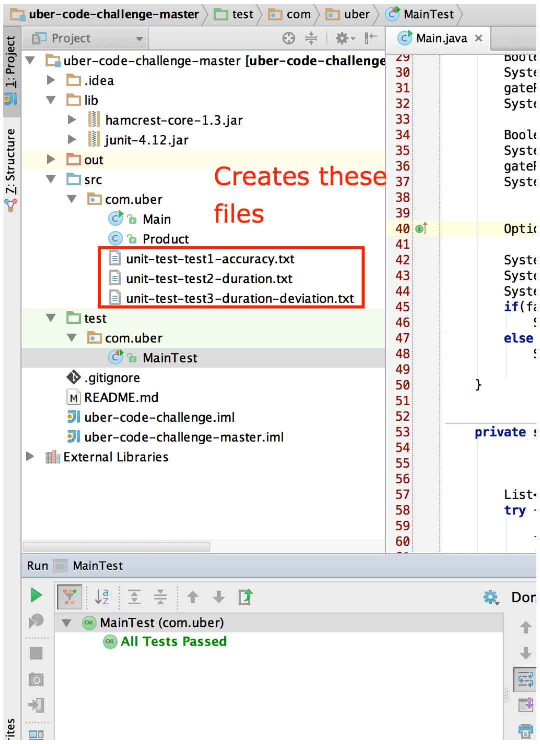
```
upperBoundExecutionSpeed = 1897

lowerBoundExecutionSpeed = 1821

results.keySet() = [test3-duration-deviation]

test3-duration-deviation: path = src/com/uber/unit-test-test3-duration-deviation.txt - result: Method invocations
out of deviation: 32(2248), 33(769), 34(583), 35(489), 36(443), 37(450), 26(646), 27(614), 28(454), 29(411),
30(435), 31(427), 44(606), 45(566), 13(1243), 46(417), 47(632), 14(719), 48(448), 15(610), 49(534), 16(559),
50(569), 17(552), 18(441), 38(566), 39(454), 19(583), 40(578), 20(560), 41(539), 21(553), 42(642), 22(573),
43(619), 23(512), 7(767), 24(528), 8(599), 25(562), 9(534), 10(554), 11(423), 12(405), 88(598), 89(499), 1(15161),
90(523), 2(675), 82(686), 91(588), 3(759), 83(720), 92(562), 4(586), 84(549), 93(543), 5(591), 85(617), 86(640),
6(602), 94(575), 87(571), 95(556), 96(451), 97(443), 76(663), 98(514), 99(577), 57(1402), 77(636), 100(469),
58(729), 78(746), 59(523), 79(510), 60(492), 80(545), 51(577), 61(567), 81(558), 62(565), 52(539), 69(485),
53(634), 70(571), 54(549), 71(544), 55(575), 72(561), 56(531), 73(558), 74(450), 75(482), 63(693), 64(794),
65(488), 66(439), 67(472), 68(446),


Process finished with exit code 0
```



Creates these files

MainTest (com.uber)
All Tests Passed