(base) mimi@mithilas-air ~ % cd test

(base) mimi@mithilas-air test % java -jar target/benchmarks.jar -wi 15 -i 15 -f 5 -tu ns -w 500ms -r 500ms -bm avgt

# JMH version: 1.37

# VM version: JDK 21.0.4, OpenJDK 64-Bit Server VM, 21.0.4+7-LTS

# VM invoker: /Library/Java/JavaVirtualMachines/temurin-21.jdk/Contents/Home/bin/java

# VM options: <none>

# Blackhole mode: compiler (auto-detected, use -Djmh.blackhole.autoDetect=false to disable)

# Warmup: 15 iterations, 500 ms each

# Measurement: 15 iterations, 500 ms each

# Timeout: 10 min per iteration

# Threads: 1 thread, will synchronize iterations

# Benchmark mode: Average time, time/op

# Benchmark: org.sample.MyBenchmark.testMethod

# Run progress: 0.00% complete, ETA 00:01:15

# Fork: 1 of 5

# Warmup Iteration 1: 0.561 ns/op

# Warmup Iteration 2: 0.736 ns/op

# Warmup Iteration 3: 0.559 ns/op

# Warmup Iteration 4: 0.559 ns/op

# Warmup Iteration 5: 0.559 ns/op

# Warmup Iteration 6: 0.559 ns/op

# Warmup Iteration 7: 0.559 ns/op

# Warmup Iteration 8: 0.559 ns/op

# Warmup Iteration 9: 0.559 ns/op

# Warmup Iteration 10: 0.559 ns/op

# Warmup Iteration 11: 0.560 ns/op

# Warmup Iteration 12: 0.559 ns/op

# Warmup Iteration 13: 0.559 ns/op

# Warmup Iteration 14: 0.559 ns/op

# Warmup Iteration 15: 0.560 ns/op

Iteration 1: 0.559 ns/op

Iteration 2: 0.559 ns/op

Iteration 3: 0.559 ns/op

Iteration 4: 0.559 ns/op

Iteration 5: 0.559 ns/op

Iteration 6: 0.559 ns/op

Iteration 7: 0.562 ns/op

Iteration 8: 0.562 ns/op

Iteration 9: 0.559 ns/op

Iteration 10: 0.559 ns/op

Iteration 11: 0.559 ns/op

Iteration 12: 0.559 ns/op

Iteration 13: 0.559 ns/op

Iteration 14: 0.657 ns/op

Iteration 15: 0.559 ns/op

# Run progress: 20.00% complete, ETA 00:01:01

# Fork: 2 of 5

# Warmup Iteration 1: 0.561 ns/op

# Warmup Iteration 2: 0.729 ns/op

# Warmup Iteration 3: 0.559 ns/op

# Warmup Iteration 4: 0.559 ns/op

# Warmup Iteration 5: 0.559 ns/op

# Warmup Iteration 6: 0.559 ns/op

# Warmup Iteration 7: 0.559 ns/op

# Warmup Iteration 8: 0.559 ns/op

# Warmup Iteration 9: 0.559 ns/op

# Warmup Iteration 10: 0.559 ns/op

# Warmup Iteration 11: 0.561 ns/op

# Warmup Iteration 12: 0.564 ns/op

# Warmup Iteration 13: 0.559 ns/op

# Warmup Iteration 14: 0.559 ns/op

# Warmup Iteration 15: 0.559 ns/op

Iteration 1: 0.559 ns/op

Iteration 2: 0.559 ns/op

Iteration 3: 0.559 ns/op

Iteration 4: 0.559 ns/op

Iteration 5: 0.559 ns/op

Iteration 6: 0.559 ns/op

Iteration 7: 0.559 ns/op

Iteration 8: 0.559 ns/op

Iteration 9: 0.559 ns/op

Iteration 10: 0.657 ns/op

Iteration 11: 0.559 ns/op

Iteration 12: 0.559 ns/op

Iteration 13: 0.559 ns/op

Iteration 14: 0.559 ns/op

Iteration 15: 0.559 ns/op

# Run progress: 40.00% complete, ETA 00:00:46

# Fork: 3 of 5

# Warmup Iteration 1: 0.561 ns/op

# Warmup Iteration 2: 0.564 ns/op

# Warmup Iteration 3: 0.559 ns/op

# Warmup Iteration 4: 0.559 ns/op

# Warmup Iteration 5: 0.559 ns/op

# Warmup Iteration 6: 0.559 ns/op

# Warmup Iteration 7: 0.559 ns/op

# Warmup Iteration 8: 0.559 ns/op

# Warmup Iteration 9: 0.559 ns/op

# Warmup Iteration 10: 0.559 ns/op

# Warmup Iteration 11: 0.724 ns/op

# Warmup Iteration 12: 0.559 ns/op

# Warmup Iteration 13: 0.559 ns/op

# Warmup Iteration 14: 0.560 ns/op

# Warmup Iteration 15: 0.562 ns/op

Iteration 1: 0.559 ns/op

Iteration 2: 0.559 ns/op

Iteration 3: 0.559 ns/op

Iteration 4: 0.559 ns/op

Iteration 5: 0.559 ns/op

Iteration 6: 0.657 ns/op

Iteration 7: 0.559 ns/op

Iteration 8: 0.559 ns/op

Iteration 9: 0.559 ns/op

Iteration 10: 0.559 ns/op

Iteration 11: 0.559 ns/op

Iteration 12: 0.559 ns/op

Iteration 13: 0.559 ns/op

Iteration 14: 0.559 ns/op

Iteration 15: 0.559 ns/op

# Run progress: 60.00% complete, ETA 00:00:30

# Fork: 4 of 5

# Warmup Iteration 1: 0.561 ns/op

# Warmup Iteration 2: 0.564 ns/op

# Warmup Iteration 3: 0.559 ns/op

# Warmup Iteration 4: 0.559 ns/op

# Warmup Iteration 5: 0.724 ns/op

# Warmup Iteration 6: 0.559 ns/op

# Warmup Iteration 7: 0.559 ns/op

# Warmup Iteration 8: 0.559 ns/op

# Warmup Iteration 9: 0.724 ns/op

# Warmup Iteration 10: 0.559 ns/op

# Warmup Iteration 11: 0.559 ns/op

# Warmup Iteration 12: 0.559 ns/op

# Warmup Iteration 13: 0.559 ns/op

# Warmup Iteration 14: 0.559 ns/op

# Warmup Iteration 15: 0.559 ns/op

Iteration 1: 0.559 ns/op

Iteration 2: 0.559 ns/op

Iteration 3: 0.562 ns/op

Iteration 4: 0.559 ns/op

Iteration 5: 0.559 ns/op

Iteration 6: 0.559 ns/op

Iteration 7: 0.559 ns/op

Iteration 8: 0.559 ns/op

Iteration 9: 0.559 ns/op

Iteration 10: 0.559 ns/op

Iteration 11: 0.559 ns/op

Iteration 12: 0.559 ns/op

Iteration 13: 0.559 ns/op

Iteration 14: 0.559 ns/op

Iteration 15: 0.559 ns/op

# Run progress: 80.00% complete, ETA 00:00:15

# Fork: 5 of 5

# Warmup Iteration 1: 0.561 ns/op

# Warmup Iteration 2: 0.564 ns/op

# Warmup Iteration 3: 0.559 ns/op

# Warmup Iteration 4: 0.559 ns/op

# Warmup Iteration 5: 0.559 ns/op

# Warmup Iteration 6: 0.559 ns/op

# Warmup Iteration 7: 0.559 ns/op

# Warmup Iteration 8: 0.559 ns/op

# Warmup Iteration 9: 0.559 ns/op

# Warmup Iteration 10: 0.559 ns/op

# Warmup Iteration 11: 0.559 ns/op

# Warmup Iteration 12: 0.559 ns/op

# Warmup Iteration 13: 0.560 ns/op

# Warmup Iteration 14: 0.559 ns/op

# Warmup Iteration 15: 0.725 ns/op

Iteration 1: 0.559 ns/op

Iteration 2: 0.559 ns/op

Iteration 3: 0.559 ns/op

Iteration 4: 0.561 ns/op

Iteration 5: 0.562 ns/op

Iteration 6: 0.560 ns/op

Iteration 7: 0.559 ns/op

Iteration 8: 0.559 ns/op

Iteration 9: 0.559 ns/op

Iteration 10: 0.559 ns/op

Iteration 11: 0.559 ns/op

Iteration 12: 0.559 ns/op

Iteration 13: 0.559 ns/op

Iteration 14: 0.559 ns/op

Iteration 15: 0.560 ns/op

Result "org.sample.MyBenchmark.testMethod":

0.563 ±(99.9%) 0.008 ns/op [Average]

(min, avg, max) = (0.559, 0.563, 0.657), stdev = 0.019

CI (99.9%): [0.556, 0.571] (assumes normal distribution)

# Run complete. Total time: 00:01:16

REMEMBER: The numbers below are just data. To gain reusable insights, you need to follow up on

why the numbers are the way they are. Use profilers (see -prof, -lprof), design factorial

experiments, perform baseline and negative tests that provide experimental control, make sure

the benchmarking environment is safe on JVM/OS/HW level, ask for reviews from the domain experts.

Do not assume the numbers tell you what you want them to tell.

NOTE: Current JVM experimentally supports Compiler Blackholes, and they are in use. Please exercise

extra caution when trusting the results, look into the generated code to check the benchmark still

works, and factor in a small probability of new VM bugs. Additionally, while comparisons between

different JVMs are already problematic, the performance difference caused by different Blackhole

modes can be very significant. Please make sure you use the consistent Blackhole mode for comparisons.

Benchmark Mode Cnt Score Error Units

MyBenchmark.testMethod avgt 75 0.563 ± 0.008 ns/op

(base) mimi@mithilas-air test %