# Raport – Reflection Benchmark

Do pomiarów użyłem Google Caliper. Jest to framework, który umożliwia pisanie i uruchamianie microbenchmarków w Javie. Jego zaletą jest między innymi to, że sam zarządza rozgrzewką oraz liczbą powtórzeń głównej pętli w danym benchmarku.

## Opis benchmarków

runMyMethod – mierzy czas dostępu do metody z parametrem
runMyMethodReflection – to samo za pomocą refleksji

setIntField – mierzy czas zapisu do pola publicznego prymitywnego typu (int)
setIntFieldReflection – to samo za pomocą refleksji
setStringField – mierzy czas zapisu do pola publicznego używając referencji (typ String)
setStringFieldReflection – to samo za pomocą refleksji

getIntField, getIntFieldReflection, getStringField, getStringFieldReflection – analogicznie jak wyżej

## Uruchamianie

mvn clean install exec:java

## Info o systemie

java version "1.8.0_74"
Java(TM) SE Runtime Environment (build 1.8.0_74-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.74-b02, mixed mode)
OS name: "windows 10", version: "10.0", arch: "amd64", family: "dos"
CPU: Intel(R) Core(TM) i7-3612QM CPU @ 2.10GHz

## Wyniki

| SCENARIO.BENCHMARKSPEC.METHODNAME | RUNTIME (NS) |
|---|---|
| getIntField | 0.328 |
| getIntFieldReflection | 307.800 |
| getStringField | 0.330 |
| getStringFieldReflection | 126.814 |
| runMyMethod | 0.328 |
| runMyMethodReflection | 166.843 |
| setIntField | 0.659 |
| setIntFieldReflection | 309.514 |
| setStringField | 0.986 |
| setStringFieldReflection | 129.404 |
| HIDDEN DIMENSIONS (1) | |
| INVARIANTS (742) | |