JCStress: Eliminating the Nightmare of Debugging Concurrency Problems

jfeatures.com

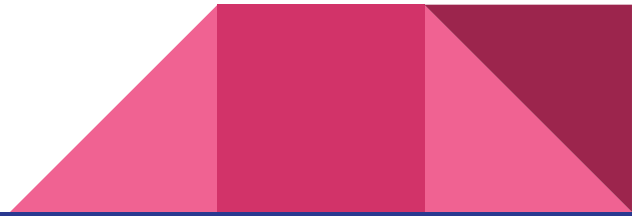
About me

Vipin Sharma

Senior Software Developer, 12+ year exp with Java

I help professional Java developers to learn language features so they can work on the best Java projects.

Blog: jfeatures.com

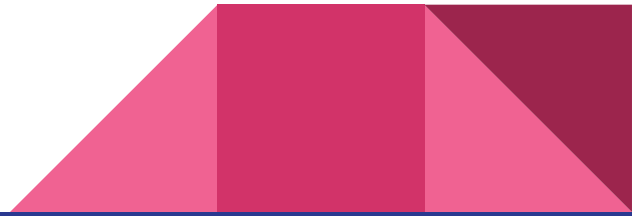


Agenda

- Introduction to JCStress
- Writing you first JCStress test
- Interpreting test results
- JCStress API to test common use cases

Bug in Java concurrency code

- Is your concurrent code correct?
- Have you came across debugging concurrency bug?
- Is concurrent code stable because it has been working fine for last couple of years?



Java Concurrency Stress (jcstress)

- Avoid risk.
- Developed and used by JVM developers to test JVM Itself.

How to use JCStress

First JCStress Test

```
@JCStressTest

// These are the test outcomes.
@Outcome(id = "1, 1", expect = Expect.ACCEPTABLE_INTERESTING, desc = "Both actors came up with the same value: atomicity failure.")
@Outcome(id = "1, 2", expect = Expect.ACCEPTABLE, desc = "actor1 incremented, then actor2.")
@Outcome(id = "2, 1", expect = Expect.ACCEPTABLE, desc = "actor2 incremented, then actor1.")

// This is a state object
@State
public class APISample_01_Simple {

    int v;

    @Actor
    public void actor1(II_Result r) {
        r.r1 = ++v; // record result from actor1 to field r1
    }

    @Actor
    public void actor2(II_Result r) {
        r.r2 = ++v; // record result from actor2 to field r2
    }

}
```

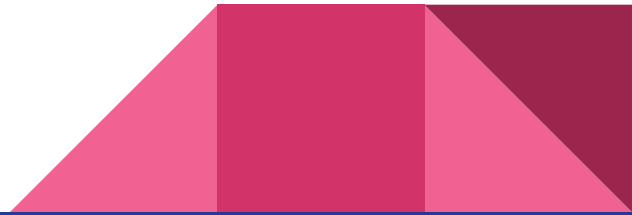
Test with AtomicInteger

```
public class ConcurrencySample_01_OperationAtomicity {

    @JCStressTest
    @Outcome(id = "1", expect = FORBIDDEN,  desc = "One update lost.")
    @Outcome(id = "2", expect = ACCEPTABLE, desc = "Both updates.")
    @State
    public static class AtomicIncrement {
        AtomicInteger ai = new AtomicInteger();

        @Actor
        public void actor1() {
            ai.incrementAndGet();
        }

        @Actor
        public void actor2() {
            ai.incrementAndGet();
        }

        @Arbiter
        public void arbiter(I_Result r) {
            r.r1 = ai.get();
        }
    }
}

}
```

Maven project to run JCStress tests

Creating standalone JCStress project using maven archetype.

```
mvn archetype:generate \
 -DinteractiveMode=false \
 -DarchetypeGroupId=org.openjdk.jcstress \
 -DarchetypeArtifactId=jcstress-java-test-archetype \
 -DgroupId=com.jfeatures \
 -DartifactId=jcstresstest \
 -Dversion=1.0
```

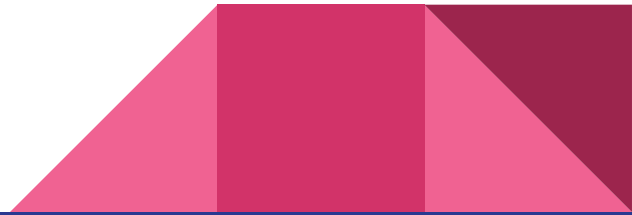


Running tests

cd jcstresstest
mvn clean verify
java -jar target/jcstress.jar

java -jar target/jcstress.jar -t ConcurrencyTest

Test report:
results/index.html



Demo Time!

Interpreting test results

Test configurations

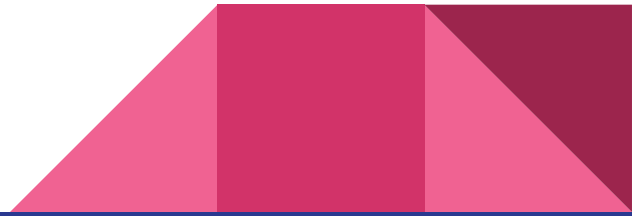
TC 1 JVM options: [-XX:-TieredCompilation] Iterations: 5 Time: 1000 Stride: [10, 10000] (capped by NONE)
TC 2 JVM options: [-XX:-TieredCompilation, -XX:+StressLCM, -XX:+StressGCM] Iterations: 5 Time: 1000 Stride: [10, 10000] (capped by NONE)
TC 3 JVM options: [-XX:TieredStopAtLevel=1] Iterations: 5 Time: 1000 Stride: [10, 10000] (capped by NONE)
TC 4 JVM options: [-Xint] Iterations: 5 Time: 1000 Stride: [10, 10000] (capped by NONE)

Observed states

Observed state	TC 1	TC 2	TC 3	TC 4	Expectation	Interpretation
1, 1	3042845	2598097	1477106	54716	ACCEPTABLE_INTERESTING	Both actors came up with the same value: atomicity failure.
1, 2	5837513	7422002	3769969	971612	ACCEPTABLE	actor1 incremented, then actor2.
2, 1	3276123	3803562	2396666	578393	ACCEPTABLE	actor2 incremented, then actor1.
	OK	OK	OK	OK		

More useful APIs

- Signal
- Shared Metadata
- Adding test descriptions and references



Test Report with Description and Reference

org.openjdk.jcstress.samples.APISample_06_Descriptions

Description and references

Sample Hello World test

http://openjdk.java.net/projects/code-tools/jcstress/

Environment

java.specification.name	Java Platform API Specification
java.specification.vendor	Oracle Corporation
java.specification.version	1.8
java.vendor	Oracle Corporation
java.version	1.8.0_201
java.vm.name	Java HotSpot(TM) 64-Bit Server VM
java.vm.vendor	Oracle Corporation
java.vm.version	25.201-b09
os.arch	amd64
os.name	Windows 10
os.version	10.0

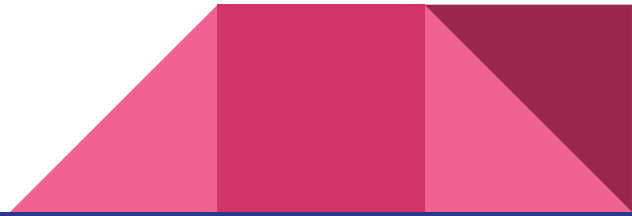
Helpful command line options

Modes (-m)
Test name(-t)
verbose result (-v)
Number of CPUs used (-c)
...

Result classes

Naming in Result classes.

I	:	int
Z	:	boolean
F	:	float
J	:	long
S	:	short
B	:	byte
C	:	char
D	:	double
L	:	object



Helpful links

https://github.com/openjdk/jcstress

https://github.com/eclipse/eclipse-collections

Q & A

jfeatures.com
twitter @vipinbit
vipin@jfeatures.com

Get eBook 5 STEPS TO BEST JAVA JOBS for Free!