Contemplate
THINKING ABOUT SOFTWARE

Introduction to ThreadSafe

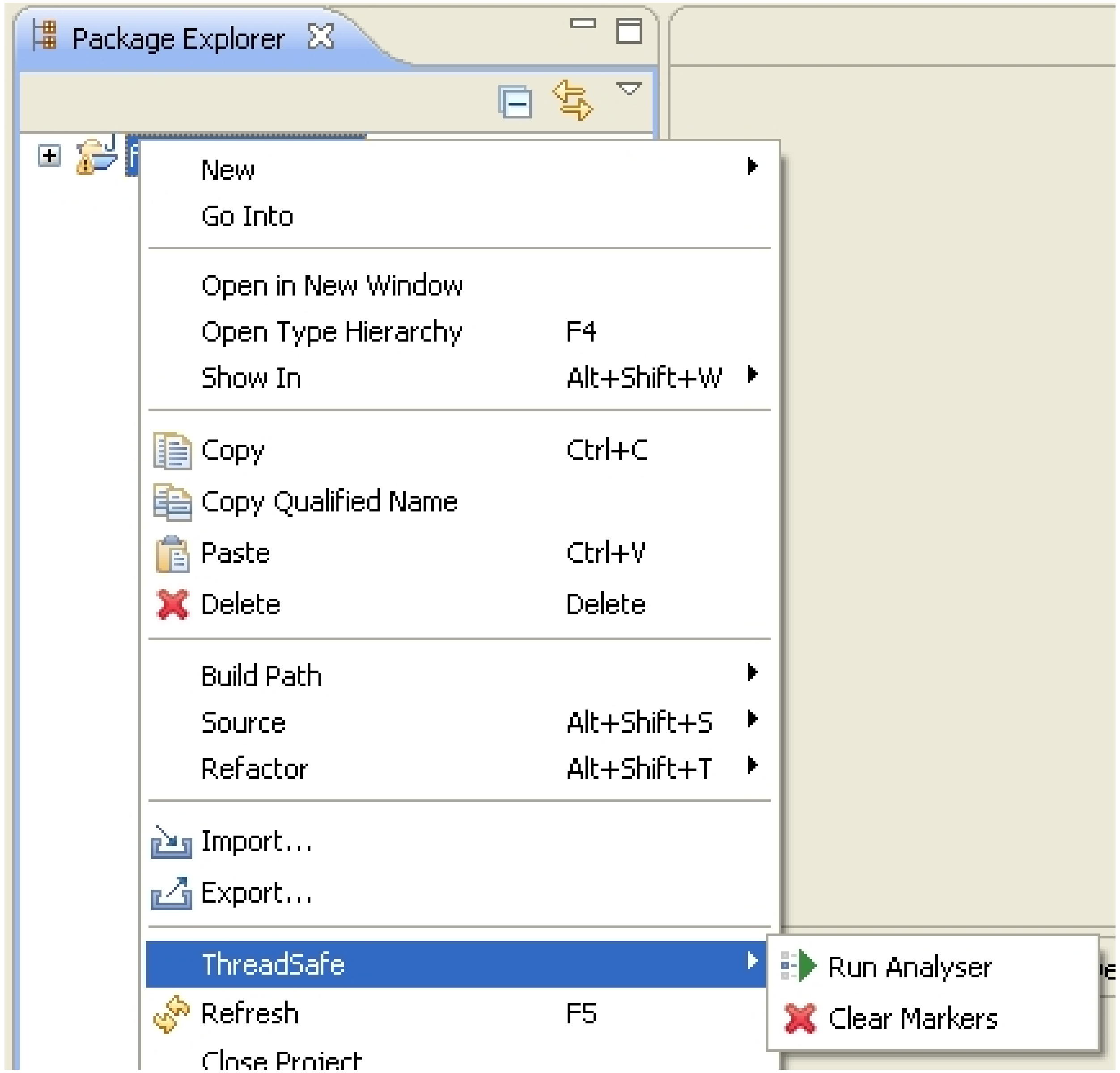
Martin Ellis

JBUG:Scotland

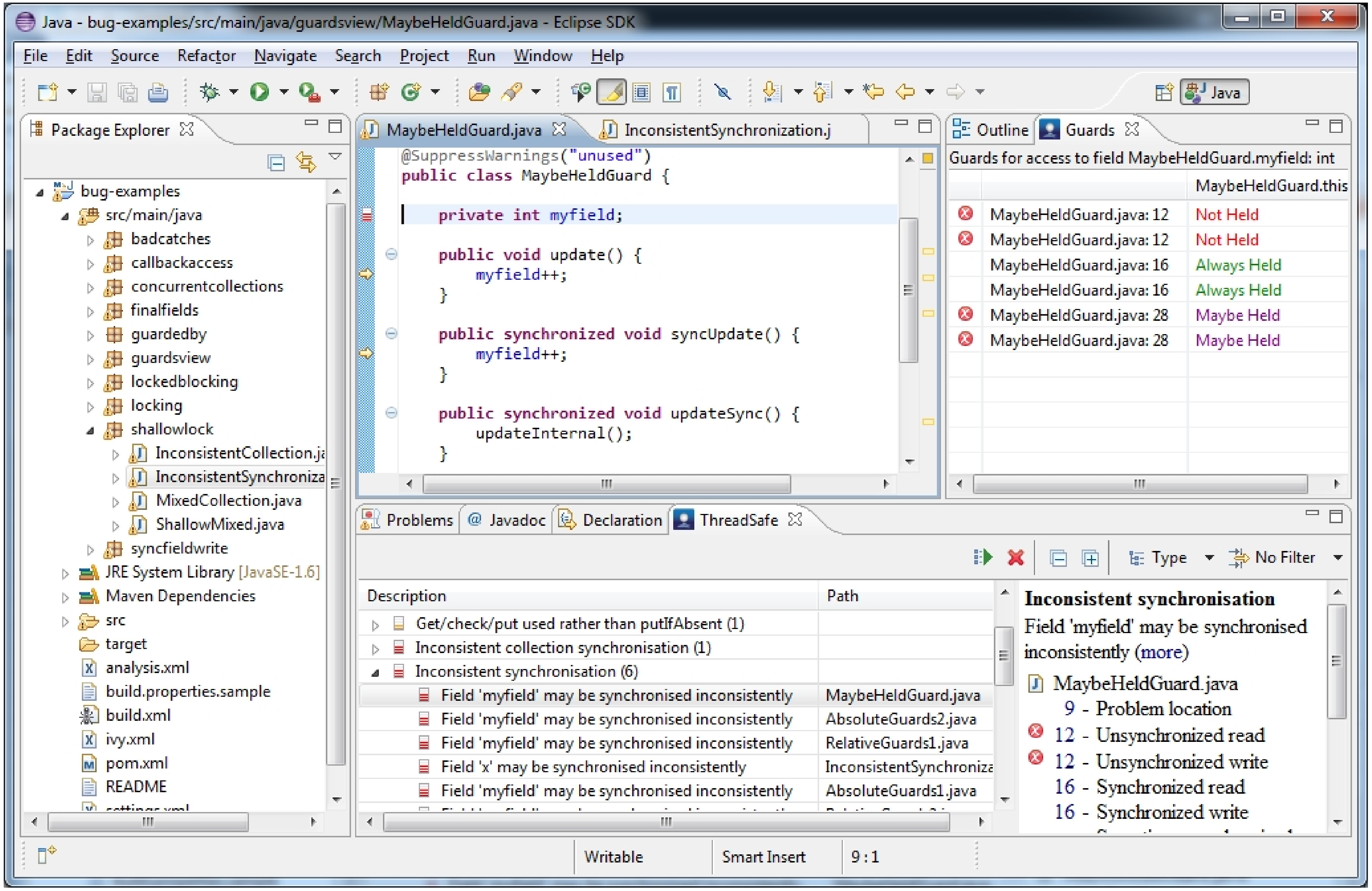
What is ThreadSafe?

- Finds Java concurrency bugs
  - with a focus on commonly used concurrency: such as synchronization, java.util.concurrent…
- Uses static analysis of JVM bytecode
  - no need to run the program
- Integrates into Eclipse and Sonar
  - Eclipse is the best place to triage bugs
  - Sonar improves visibility of bugs in a team

Analysing projects in Eclipse



ThreadSafe in Eclipse



Java - bug-examples/src/main/java/guardsview/MaybeHeldGuard.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package Explorer
MaybeHeldGuard.java
InconsistentSynchronization.j
Outline
Guards
Problems
Javadoc
Declaration
ThreadSafe
Type
No Filter
Writable
Smart Insert
9 : 1

Package Explorer:

- bug-examples
  - src/main/java
    - badcatches
    - callbackaccess
    - concurrentcollections
    - finalfields
    - guardedby
    - guardsview
    - lockedblocking
    - locking
    - shallowlock
      - InconsistentCollection.ja
      - InconsistentSynchroniza
      - MixedCollection.java
      - ShallowMixed.java
    - syncfieldwrite
  - JRE System Library [JavaSE-1.6]
  - Maven Dependencies
  - src
  - target
  - analysis.xml
  - build.properties.sample
  - build.xml
  - ivy.xml
  - pom.xml
  - README

```
@SuppressWarnings("unused")
public class MaybeHeldGuard {

    private int myfield;

    public void update() {
        myfield++;
    }

    public synchronized void syncUpdate() {
        myfield++;
    }

    public synchronized void updateSync() {
        updateInternal();
    }
```

Guards for access to field MaybeHeldGuard.myfield: int

		MaybeHeldGuard.this
❌	MaybeHeldGuard.java: 12	Not Held
❌	MaybeHeldGuard.java: 12	Not Held
	MaybeHeldGuard.java: 16	Always Held
	MaybeHeldGuard.java: 16	Always Held
❌	MaybeHeldGuard.java: 28	Maybe Held
❌	MaybeHeldGuard.java: 28	Maybe Held

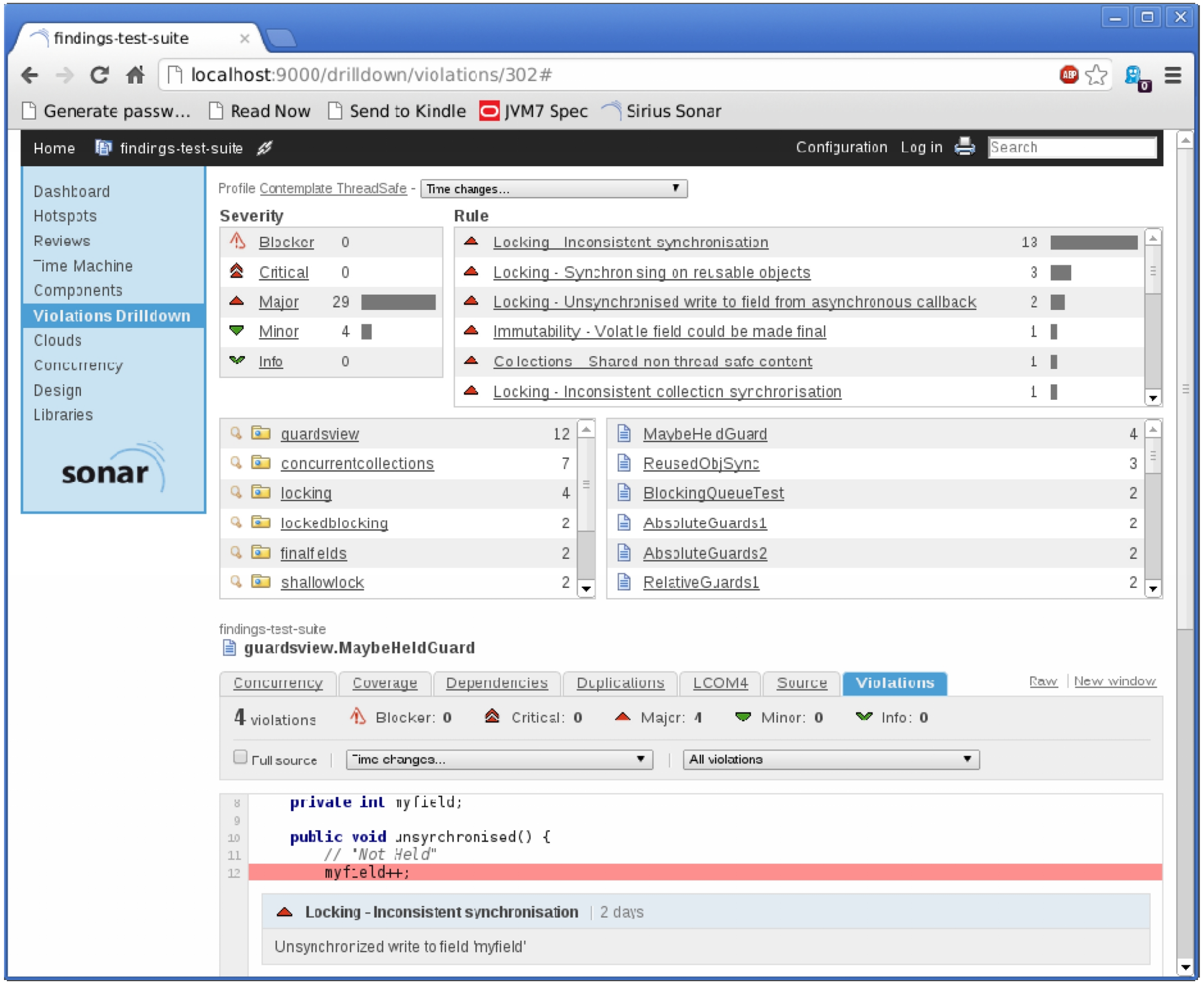
Description	Path
Get/check/put used rather than putIfAbsent (1)	
Inconsistent collection synchronisation (1)	
Inconsistent synchronisation (6)	
Field 'myfield' may be synchronised inconsistently	MaybeHeldGuard.java
Field 'myfield' may be synchronised inconsistently	AbsoluteGuards2.java
Field 'myfield' may be synchronised inconsistently	RelativeGuards1.java
Field 'x' may be synchronised inconsistently	InconsistentSynchroniza
Field 'myfield' may be synchronised inconsistently	AbsoluteGuards1.java

Inconsistent synchronisation

Field 'myfield' may be synchronised inconsistently (more)

MaybeHeldGuard.java
- 9 - Problem location
- ❌ 12 - Unsynchronized read
- ❌ 12 - Unsynchronized write
- 16 - Synchronized read
- 16 - Synchronized write

ThreadSafe in Sonar



findings-test-suite

localhost:9000/drilldown/violations/302#

Generate passw… | Read Now | Send to Kindle | JVM7 Spec | Sirius Sonar

Home | findings-test-suite | Configuration | Log in | Search

Dashboard
Hotspots
Reviews
Time Machine
Components
Violations Drilldown
Clouds
Concurrency
Design
Libraries

Profile Contemplate ThreadSafe - Time changes...

Severity		
Blocker	0	
Critical	0	
Major	29	▇▇▇▇▇
Minor	4	▇
Info	0	

Rule		
Locking - Inconsistent synchronisation	13	▇▇▇▇▇
Locking - Synchronising on reusable objects	3	▇
Locking - Unsynchronised write to field from asynchronous callback	2	▇
Immutability - Volatile field could be made final	1	▇
Collections - Shared non thread safe content	1	▇
Locking - Inconsistent collection synchronisation	1	▇

guardsview	12
concurrentcollections	7
locking	4
lockedblocking	2
finalfields	2
shallowlock	2

MaybeHeldGuard	4
ReusedObjSync	3
BlockingQueueTest	2
AbsoluteGuards1	2
AbsoluteGuards2	2
RelativeGuards1	2

findings-test-suite
guardsview.MaybeHeldGuard

Concurrency | Coverage | Dependencies | Duplications | LCOM4 | Source | **Violations** | Raw | New window

4 violations | Blocker: 0 | Critical: 0 | Major: 4 | Minor: 0 | Info: 0

☐ Full source | Time changes... | All violations

```
    private int myfield;

    public void unsynchronised() {
        // "Not Held"
        myfield++;
```

Locking - Inconsistent synchronisation | 2 days

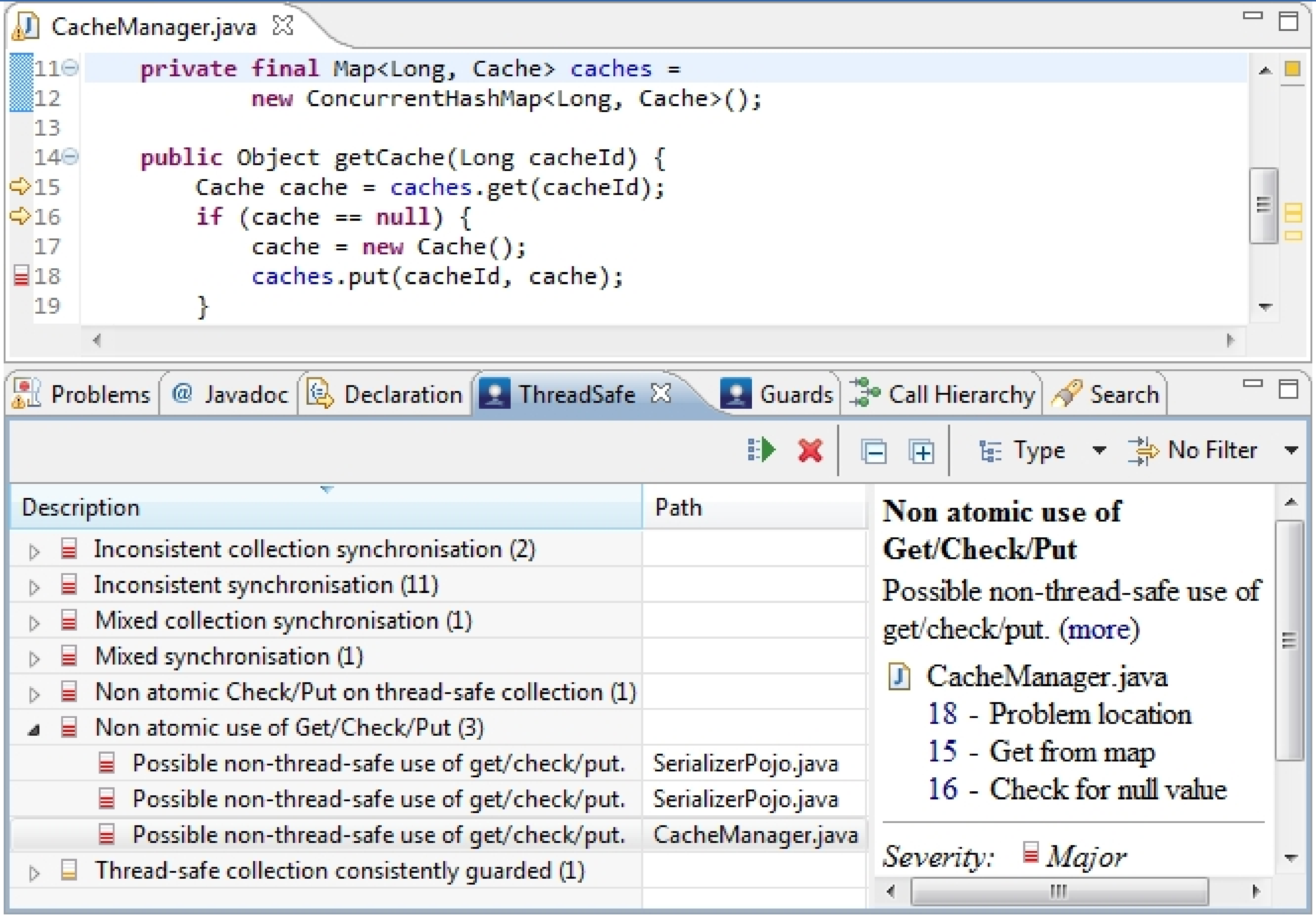
Unsynchronized write to field 'myfield'

Bug example: Get/Check/Put

```
    private final Map<Long, Cache> caches =
    new ConcurrentHashMap<Long, Cache>();

public Object getCache(Long cacheId) {
    Cache cache = caches.get(cacheId); // get
    if (cache == null) {               // check
        cache = new Cache();
        caches.put(cacheId, cache);    // put
             }
    return cache;
  }
```

ThreadSafe: Get/Check/Put



CacheManager.java

```
11      private final Map<Long, Cache> caches =
12              new ConcurrentHashMap<Long, Cache>();
13
14      public Object getCache(Long cacheId) {
15          Cache cache = caches.get(cacheId);
16          if (cache == null) {
17              cache = new Cache();
18              caches.put(cacheId, cache);
19          }
```

Description	Path
Inconsistent collection synchronisation (2)	
Inconsistent synchronisation (11)	
Mixed collection synchronisation (1)	
Mixed synchronisation (1)	
Non atomic Check/Put on thread-safe collection (1)	
Non atomic use of Get/Check/Put (3)	
Possible non-thread-safe use of get/check/put.	SerializerPojo.java
Possible non-thread-safe use of get/check/put.	SerializerPojo.java
Possible non-thread-safe use of get/check/put.	CacheManager.java
Thread-safe collection consistently guarded (1)	

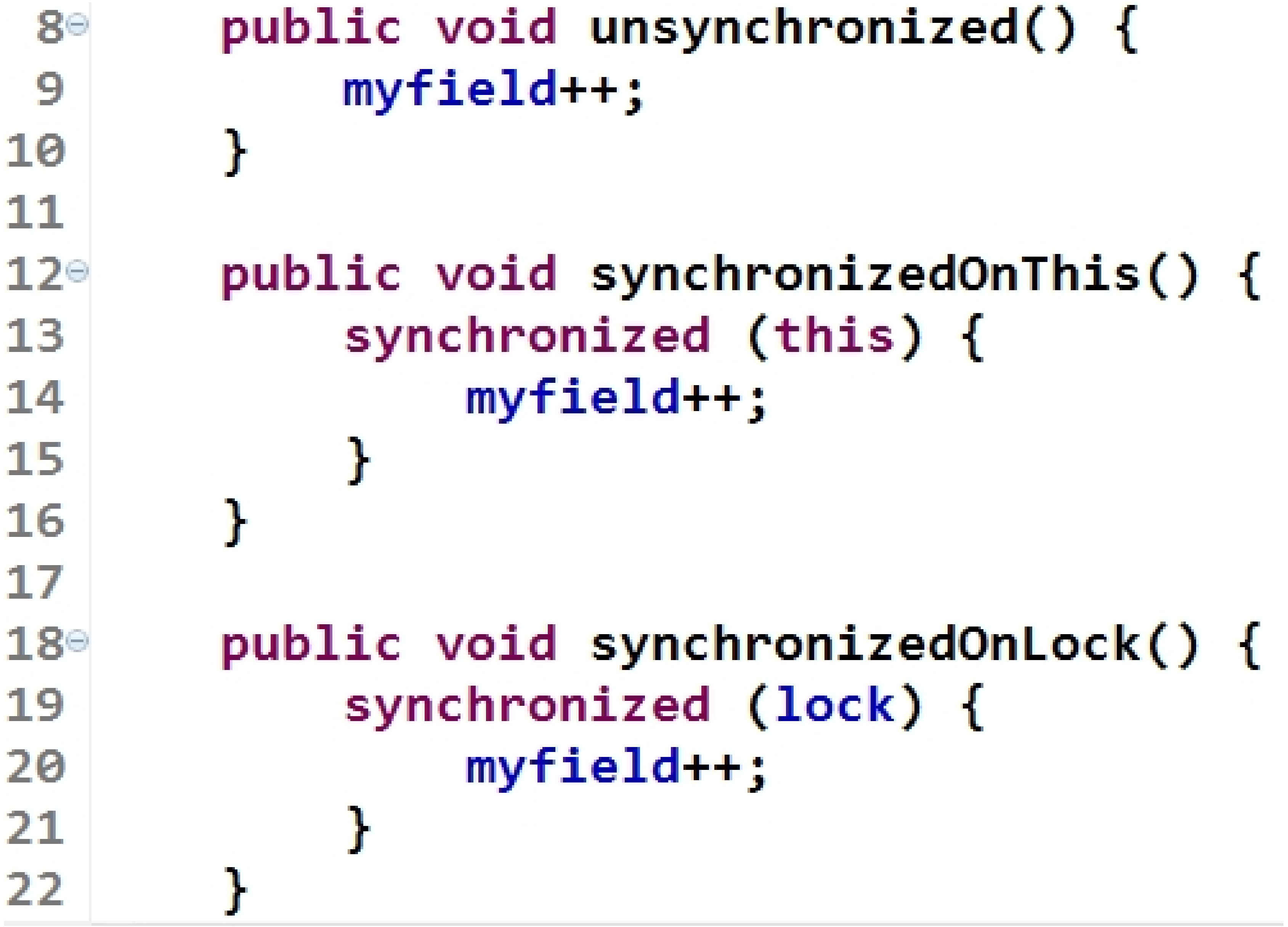
Non atomic use of Get/Check/Put

Possible non-thread-safe use of get/check/put. (more)

CacheManager.java
18 - Problem location
15 - Get from map
16 - Check for null value

Severity: *Major*

Bug: Field not guarded by one lock

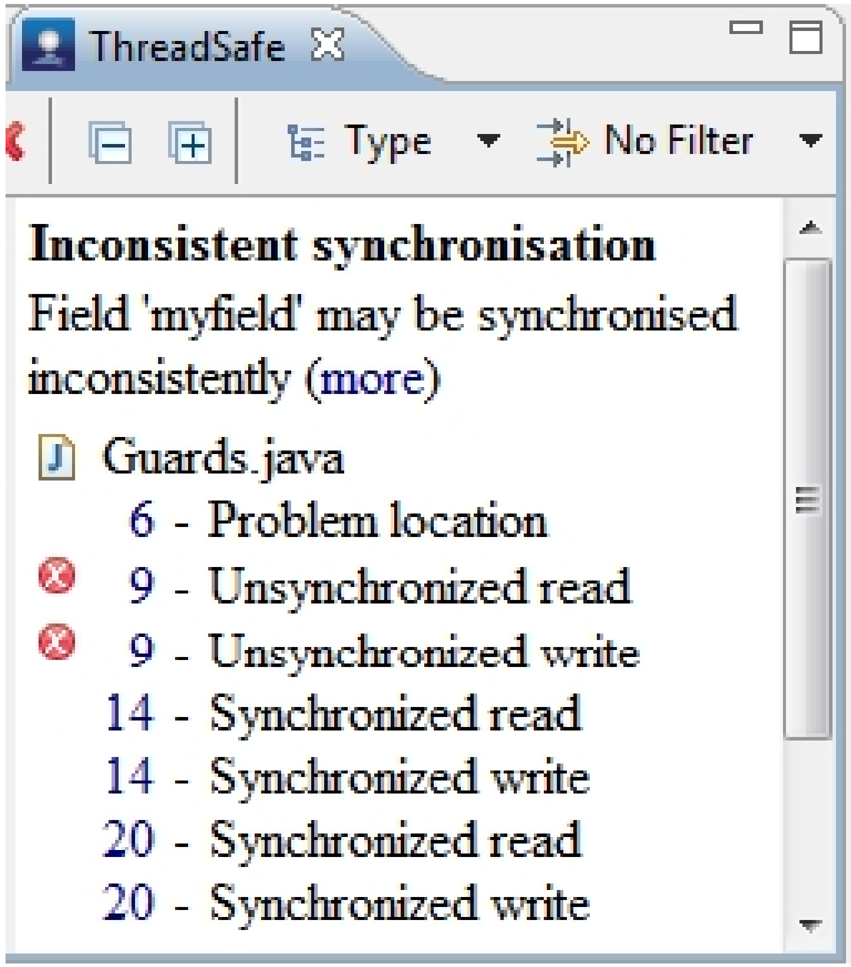


```
    public void unsynchronized() {
        myfield++;
    }

    public void synchronizedOnThis() {
        synchronized (this) {
            myfield++;
        }
    }

    public void synchronizedOnLock() {
        synchronized (lock) {
            myfield++;
        }
    }
```

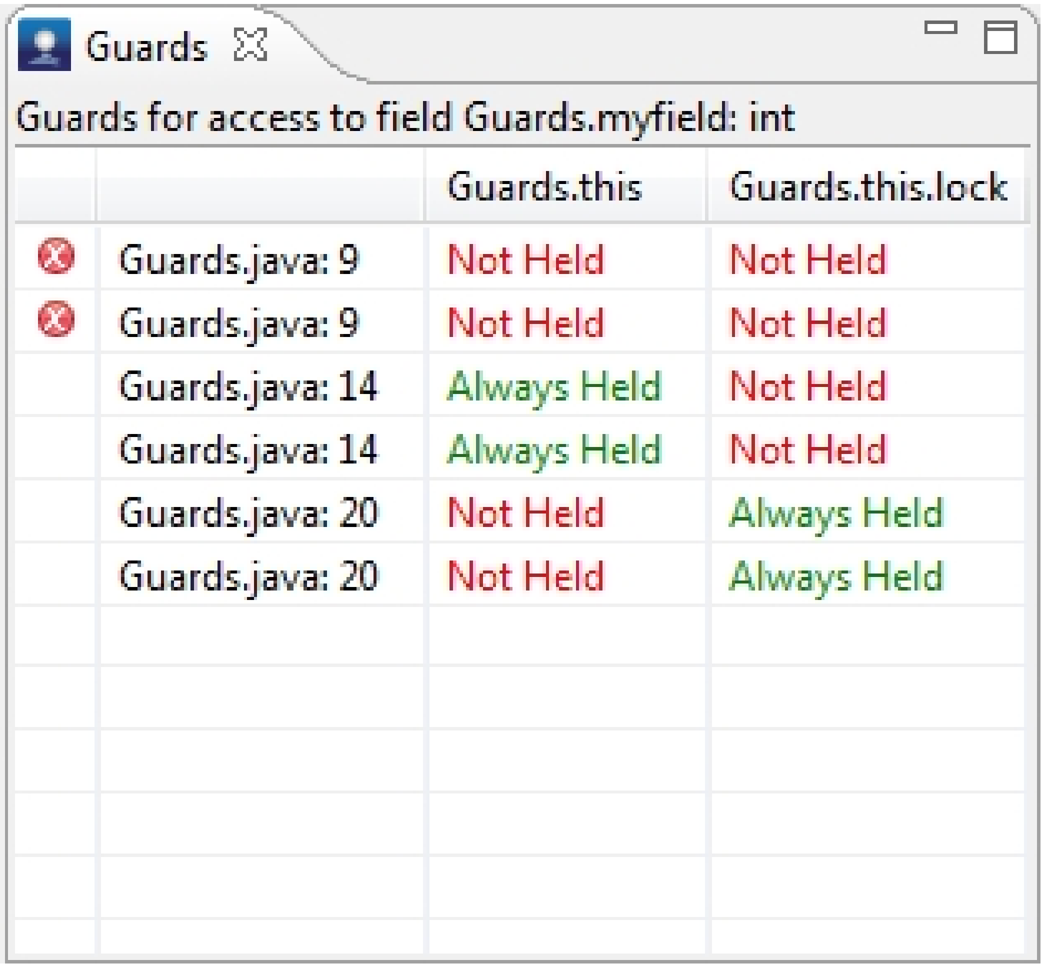
ThreadSafe: Field Guards



Inconsistent synchronisation

Field 'myfield' may be synchronised inconsistently (more)

Guards.java
- 6 - Problem location
- 9 - Unsynchronized read
- 9 - Unsynchronized write
- 14 - Synchronized read
- 14 - Synchronized write
- 20 - Synchronized read
- 20 - Synchronized write



Guards for access to field Guards.myfield: int

		Guards.this	Guards.this.lock
❌	Guards.java: 9	Not Held	Not Held
❌	Guards.java: 9	Not Held	Not Held
	Guards.java: 14	Always Held	Not Held
	Guards.java: 14	Always Held	Not Held
	Guards.java: 20	Not Held	Always Held
	Guards.java: 20	Not Held	Always Held

Concurrency Bugs

ThreadSafe finds bugs related to:

- Locking
  - Consistency: every access, common lock
- Atomicity
  - get/check/put; isLocked()/lock()
- Collections
  - legacy, guarded, synchronized*, j.u.c
- Deadlock
- Visibility problems

More information

- Contemplate
  - http://contemplateltd.com
- Martin Ellis
  - martin@contemplateltd.com