

Wikiprint Book

Title: @GuardedBy and LockChecker

Subject: Java Path Finder - projects/jpf-aprop/GuardedBy

Version: 1

Date: 03/12/2013 05:30:05 PM

Table of Contents

TracNav	3
Introduction	3
Installing JPF	3
User Guide	3
Developer Guide	3
MJI	3
Projects	3
About	4
@GuardedBy and LockChecker	4
Properties	5

TracNav

- [JPFWiki - Welcome Page](#)

Introduction

- [What is JPF](#)
- [Testing vs model checking](#)
- [Random Example](#)
- [Race Example](#)
- [JPF classification](#)

Installing JPF

- [System requirements](#)
- [Download snapshots](#)
- [Download repositories](#)
- [Create site.properties](#)
- [Install NetBeans IDE plugin](#)
- [Install Eclipse IDE plugin](#)
- [Building and testing](#)

User Guide

- [Application Types](#)
- [JPF Components](#)
- [Configuring JPF](#)
- [Running JPF](#)
- [JPF Output](#)
- [The JPF API](#)

Developer Guide

- [Design](#)
- [Choice Generator](#)
- [Partial Order Reduction](#)
- [Attributes](#)
- [Listener](#)

MJI

- [Mangling for MJI](#)
- [Bytecode Factory](#)
- [Logging](#)
- [Report](#)
- [Embedded](#)
- [JPF tests](#)
- [JPF project layout](#)
- [Create a JPF project](#)
- [Coding Conventions](#)
- [Hosting update site](#)

Projects

- [jpf-core](#)
- [jpf-actor](#)
- [jpf-awt](#)
- [jpf-awt-shell](#)

- [jpf-concurrent](#)
- [jpf-cv](#)
- [jpf-delayed](#)
- [jpf-guided-test](#)
- [jpf-mango](#)
- [jpf-racefinder](#)
- [jpf-rtembed](#)
- [jpf-statechart](#)
- [net-iocache](#)
- [jpf-aprop](#)
- [jpf-numeric](#)
- [jpf-symbc](#)
- [jpf-concolic](#)
- [jpf-symbc-load?](#)
- [jpf-extended-test-gen](#)
- [jpf-parallel-spf?](#)
- [eclipse-jpf](#)
- [netbeans-jpf](#)
- [jpf-inspector](#)
- [jpf-shell](#)
- [jpf-template](#)
- [jpf-trace-server](#)
- [standard NB example](#)
- [Summer Projects](#)
- [External Projects](#)
- [Change\(B\)log](#)

About

- [About this Wiki](#)
- [About the Mailing Lists](#)
- [About the Development Process?](#)
- [About the Repository?](#)
- [How to Contribute](#)
- [JPF contributor account](#)
- [Events](#)
- [Presentations](#)
- [Papers](#)
- [FAQ](#)
- [History?](#)
- [Support](#)
- [People?](#)
- [Playground](#)
- [Table of Context](#)

@GuardedBy and LockChecker

The `javax.annotation.concurrent.GuardedBy` annotation is part of the [JSR-305 proposal](#). It specifies by which object access to a certain field is assumed to be lock protected:

```
import javax.annotation.concurrent.GuardedBy;

class GuardViolation {
    @GuardedBy("getLock()")
}
```

```

int myPrecious;

private Object lock = ...
private Object getLock() {
    return lock;
}

public void doSomethingPrecious () {
    myPrecious++; // error - should be protected by synchronized(getLock()){..}
}

...
}

```

The annotation takes a String parameter that specifies the lock, which can be:

- **this** - synchronized on owner object
- **<fieldname>** - synchronized on peer field of owner object
- **<classname>.class** - synchronized on class object
- **<classname>.<fieldname>** - synchronized on static field
- **<methodname>()** - synchronized on return value of method
- **<classname>.<methodname>()** - synchronized on return value of static method

Properties

To let JPF check for `@GuardedBy` violations, the associated `gov.nasa.jpf.aprop.listener.LockChecker` has to be configured:

```
listener=.aprop.listener.LockChecker
```

Alternatively, the `LockChecker` can be included in the JPF autoload `listener` set

```

listener.autoload=javax.annotation.concurrent.GuardedBy, ...

listener.javaannotation.concurrent.GuardedBy=.aprop.listener.LockChecker
...

```