Questions

Bookmark this page

Testing concurrency

1/1 point (graded)

You're running a JUnit test suite (for code written by somebody else), and some of the tests are failing. You add System.out.println statements to the one method called by all the failing test cases, in order to display some of its local variables, and the test cases suddenly start passing. Which of the following are likely reasons for this?

- [ ] The method is calling a random number generator (i.e., Math.random()), so sometimes its tests will pass by random chance.
- [x] The method has code running concurrently.
- [x] The method has a race condition.
- [ ] The method's preconditions are not being met by the test cases.

Explanation

Randomness is not a likely reason for this behavior. Although calling a random number generator does make a method nondeterministic, which is one reason why tests might sometimes pass and sometimes fail, it's less likely that the original developer would have written tests that didn't take this into account, and still more unlikely that multiple tests would start to pass coincidentally with inserting a print statement.
Concurrent code, which contains a race condition affecting its correctness, is likely to be affected by the timing changes caused by a print statement.
If the method's preconditions are not being met by the test cases, this is more likely to cause deterministic failures (test cases that always fail), rather than nondeterministic behavior.

Submit Show Answer

Answers are displayed within the problem

Some Rights Reserved

Captured by FireShot Pro: 07 January 2025, 07:47:34
https://getfireshot.com

Privacy Policy Terms of Service

© Massachusetts Institute of Technology, 2025