



Band XI International, LLC
150 Beacon Hill Drive
West Hartford, CT 06117
USA

phone: +1 860.233.1526
fax: +1 860.233.1529
web: www.bandxi.com

3 October 2007

Symposium: Test Driven Development

Brett Hackleman, Band XI International

Interests and Discussion Topics

As a member of a 3-person software development company that both practices Test Driven Development (TDD) on internal projects and teaches it during customer workshops, I am a strong supporter of TDD. Reflecting on TDD conjures up both positive and negative thoughts:

- My code is always of a higher quality and with increased functionality for the length of the entire project, when I write tests before and during development.
- I find it easy to get bogged down with “what ifs” and feature creep unless I start with a test. If a line of new code doesn’t help the test pass, then I tweak the current test, add a new test stub or comment to the testcase, or delete the code entirely.
- Writing tests does take longer than just banging out code that works the first time! However, when I start with TDD, the return on my investment pays off by the 2nd or 3rd new feature or refactoring, almost without fail. This usually occurs within the first half-day of development, but at time=0, it still takes a conscious effort to avoid the “easy” route.
- I love green bars.
- I have yet to find, or develop myself, a nightly build system that makes testing as easy as it should be. We have tried them all, but with only a few developers on a project, setting up a comprehensive build system takes a lot of effort.
- Knowing what to test, and at what granularity, is difficult for both new and old developers. Sometimes we’re testing a single method, a class, or an entire plugin/bundle. Splitting these into different categories/packages/projects often feels like too much overhead.
- Visibility of methods/fields can sometimes make it difficult to reach the code you want to test, and there is no one approach that works in all cases (subclassing, mock objects, static fields/methods, AOP, accessor methods, etc).
- Code rot affects “application” and “test” code equally
- When tests become tedious to maintain, they feel like an albatross. This is especially true for tests that contain non-code resources and therefore aren’t touched by the Eclipse refactoring tools. Strings, database tables, etc.
- Test teams love products that are thrown over the wall with testcases. On past projects we have been told the overall quality was orders of magnitude better than what they normally saw.

Goals for Participation

I am interested in discussing TDD with other Eclipse developers, and learning what tools/techniques might be out there and worth examining for small development shops. I would also appreciate any feedback on improvements that could be made to the first revision of the “FitNesse Plugin For Eclipse” (<http://bandxi.com/fitnesse/index.html>) that we created before we really knew how to develop plugins.

About the Author

Brett Hackleman is one of the founders of Band XI International, a small software and services company started in 2005 that builds everything using Eclipse tooling and OSGi service-oriented bundle architectures. He hates writing about himself in the 3rd person, but while he’s at it, he would also tell you that in his past life he was a member of the Embedded Java Enablement Team (eJET) in IBM’s Pervasive Computing Group, where he worked for 6 years in the Telematics and RFID domains. Before that, Brett was happily employed by Object Technology International, Inc.