AgitarOne JUnit Generator

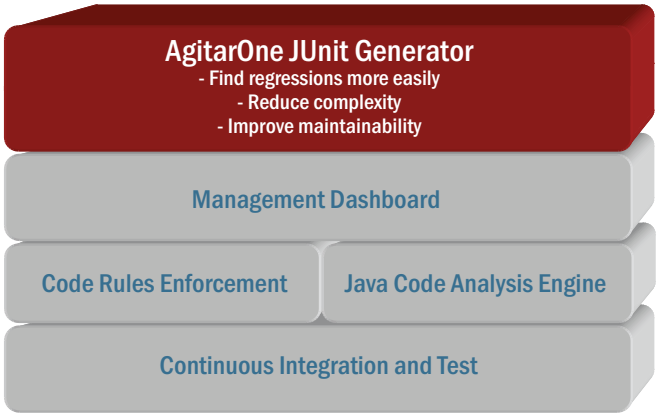
Prevent Regressions and Cut the Maintenance Cost of Your Java Applications

Let's face it — working on a "greenfield" software project is more fun than maintaining and enhancing a portfolio of existing applications. The reality, however, is that development teams must devote most of their resources to managing their portfolio of legacy applications. You'd like to extend their useful lives while reducing the lifetime cost of ownership. But their code is often buggy, ugly and frequently patched, and it's common to have tens of thousands of old Java classes that you cannot change with confidence because doing so is a major source of regressions.

With AgitarOne™ JUnit Generator, you can be agile even if your applications are fragile. It enables you to change your Java applications more quickly and easily to meet changing business needs, helps you prevent regressions, and reduces your overall maintenance costs. With AgitarOne JUnit Generator, you can achieve the following benefits:

- **Reduce your maintenance burden by 50% or more.** You can get a thorough suite of unit-level regression tests with 80% or better code coverage — at the push of a button. You can detect regressions soon after they are introduced by running the tests with every build, and dramatically reduce maintenance costs by fixing regressions when it's easiest to do so.
- **Fast feedback made easy.** AgitarOne JUnit Generator helps you get the fastest possible start with continuous integration and testing. Fast feedback brings significant improvements in team productivity and the quality of code under development.
- **Reduce uncertainty around outsourced projects.** You can ensure that your code is built for change and reduce uncertainty around the quality and maintainability of outsourced applications, by ensuring that they have a suite of unit-level regression tests.

FOR LEGACY CODE CHALLENGES



Product Overview

Automated JUnit Generation — 80% Code Coverage, or Better

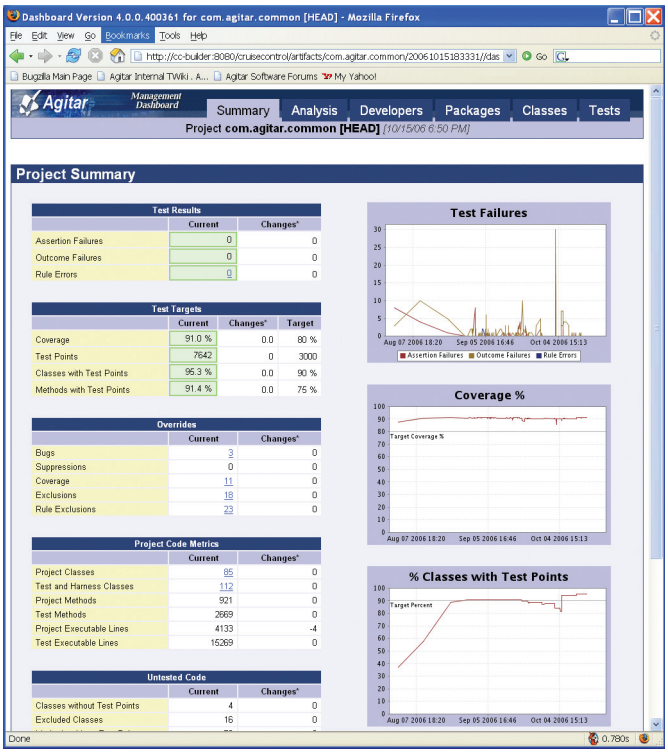
AgitarOne JUnit Generator generates thorough unit-level regression tests on your existing code base: on legacy Java applications as well as new code. The JUnit Generator analyzes your entire Java project and then generates tests that capture, and help you preserve, the code's behavior. AgitarOne JUnit Generator is so powerful that it routinely achieves 80% or better code coverage and, depending on server configuration, can generate 250,000 lines of JUnit per hour or more. The speed of test generation will scale linearly as more hardware is added. This lets the JUnit Generator deliver tests of unmatched thoroughness on even the largest projects — faster than any other approach.

The JUnit tests from AgitarOne JUnit Generator will be very thorough and will help you detect behavior changes. They are not intended to be read or enjoyed by humans; they are not intended to do what hand-crafted JUnit tests do — to improve the design of code as you write the code. These generated tests help you find regressions when you change the code, and provide the foundation for safely improving the code to cut the cost of its maintenance and enhancement.

The most thorough and effective unit tests result from a combination of automation, human insight, and domain expertise. AgitarOne JUnit Generator automatically creates high-coverage tests at the push of a button. These tests leverage Agitar's powerful mocking technology, an approach that resolves dependencies on components such as databases, and generates tests for code that would otherwise be untestable. In cases where the AgitarOne engine does not have enough information to get full coverage — such as when some domain-specific information or set-up is needed — developers can help by writing test helper methods. The typical test helper method is just a few lines of Java code but, once it's amplified by AgitarOne's powerful combinatorial and test generation engine, it will result in hundreds of lines of high-quality JUnit tests.

This is incredible for legacy code, because this means that you can get a suite of unit tests that characterize the current behavior of the code in a matter of hours. And you can immediately put this suite to work with continuous integration and testing.

AgitarOne JUnit Generator Product Overview



Management Dashboard

AgitarOne JUnit Generator includes a flexible management dashboard, producing reports that provide comprehensive feedback to empower developers and to help manage, track, and report on your project.

The management dashboard collects and shows key code and testing metrics for your project, providing continuous visibility into the testing efforts for both managers and developers.

Testing goals can be defined and tracked for the project and for individual classes, and then summarized for each developer and for the overall project.

Summary status information is provided at the individual project level or rolled up over several projects. Key metrics such as the number of test failures, overall code coverage, total number of tests, and percentage of classes and methods with tests are plotted on trend charts for easy analysis. The dashboard's e-mail notification system delivers managers and developers all the key metrics, right to their inbox.

◁ *The AgitarOne management dashboard provides comprehensive feedback on key project metrics and helps you track the unit-level quality and status of your Java project.*

Code Rules

AgitarOne JUnit Generator includes automated enforcement of Java code against a customizable set of rules, standards, and guidelines. Code rules are easy to configure to support corporate or team standards, across one or many projects.

Code rules ensure compliance and quickly detect many common errors that can then be fixed with minimal effort and risk. When you want to improve existing applications, the code rules will flag complex code so that you can focus your effort on refactoring the code that needs it the most.

Continuous Integration and Test (CIT)

AgitarOne JUnit Generator includes built-in support for continuous integration and testing. This is a proven best practice of running regression tests and validating the continued correct behavior of the code after any revisions. AgitarOne's CIT is based on CruiseControl, the most popular open-source solution for continuous integration. If you can build a Java project in Eclipse, you can start doing CIT on it in less than an hour with AgitarOne JUnit Generator.

Machine-generated JUnit regression tests plus hand-written JUnit tests give you a far more thorough set of regression tests than would be practical with hand-written tests alone. But it is still important to run these as often as possible to get the maximum benefit. Following code check-ins, AgitarOne automatically rebuilds the application, runs the regression suite, and reports back on the status.

For both new and legacy projects, continuous integration and test provides rapid feedback to developers, so they can detect and fix potential errors and unintended changes in behavior minutes after they are introduced, instead of passing them on to QA or end users.

AgitarOne Makes Unit Tests a Sustained Competitive Advantage

AgitarOne means less time tracking down defects in code you wrote weeks or months ago, and more peace of mind that the code you check in today is solid.

For distributed teams, whether offshore or outsourced, AgitarOne JUnit Generator means gaining visibility and control over the quality of software being created remotely.

From new projects to legacy systems, from local to distributed teams, AgitarOne JUnit Generator is the best thing you can do to overcome your Java development challenges.

AgitarOne JUnit Generator Deployment Approaches

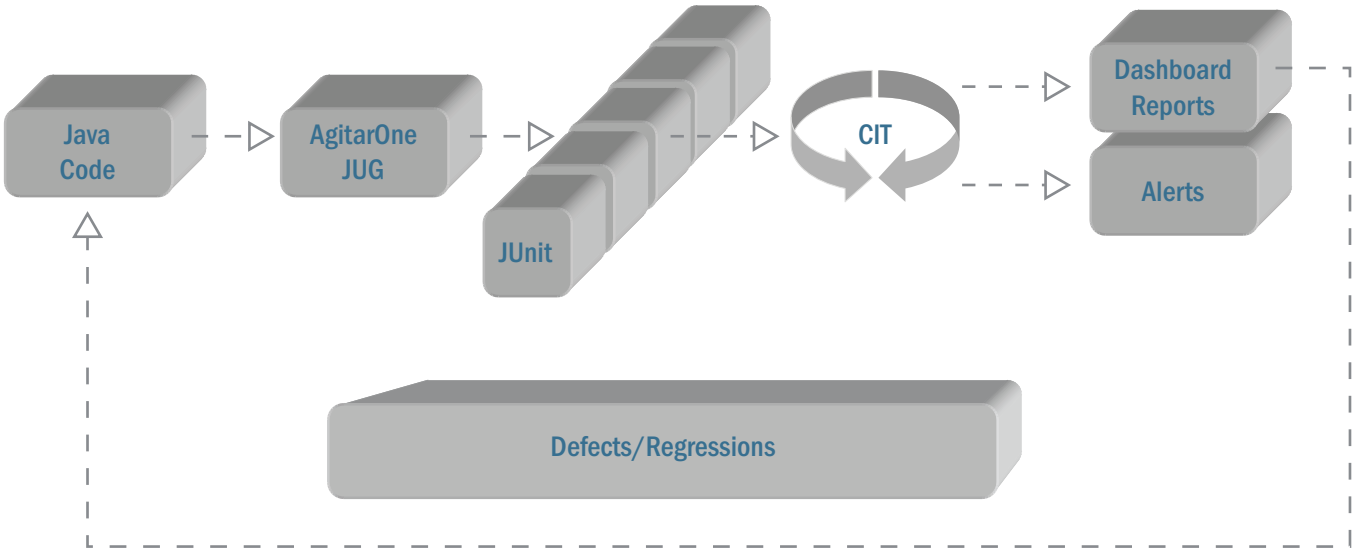
JUnit tests that are not used do not add value to any software development process. It is only when you adopt a consistent process that you get the most value from your tests. Agitar recommends first creating a safety net of unit tests for existing Java code, followed by practicing CIT (continuous integration and testing) periodically (nightly, weekly, etc.). Then leverage AgitarOne's dashboards and alert mechanism to be notified when regressions are introduced, and immediately fix the relevant defects and re-run the tests. Following this entire process on an ongoing basis has brought meaningful cost reductions and quality improvements to hundreds of Java development teams, in all industries.

Here are two recommended deployment approaches to help you get the maximum benefit from AgitarOne JUnit Generator.

Work Safer — Generate a Safety Net for Your Code
Use AgitarOne JUnit Generator to generate automatically a safety net of thorough JUnit tests that document the current behavior of the application. This safety net makes it easier to see the impact of code changes, which makes changing the code much less risky. This safety net can be generated with little or no involvement from the development staff. The generated tests are checked in and run at desired intervals using AgitarOne JUnit Generator's continuous integration and testing support. Test failures are analyzed and developers are notified when they have broken code that used to work and introduced a regression. This approach brings immediate benefits with minimal effort and no disruption to the daily work of most of the team.

Work Better — Actively Improve Your Code
Start with the thorough test suite produced by AgitarOne JUnit Generator. It enables a small "SWAT Team" of an architect, a few lead developers, or QA engineers to greatly improve the code, improving the business agility of the entire development organization. The high out-of-the-box coverage provides the necessary foundation for safe code changes. You can limit these changes to the identification and elimination of common problems, such as unhandled exceptions. You can go further and refactor the code; refactoring takes work but it is the best way to reduce the complexity and ongoing cost of the code. Any of this work can be done on an as-needed basis, focused on the most important and/or fragile parts of the code as they are about to be modified. As this work results in less fragile code, the overall team will spend much less time on legacy maintenance and enhancement. This frees staff for new applications and to start to simplify and improve other applications.

AGITARONE DEPLOYMENT PROCESS



"Code without tests is bad code. It doesn't matter how well written it is; it doesn't matter how pretty or object-oriented or well-encapsulated it is. With tests, we can change the behavior of our code quickly and verifiably. Without them, we really don't know if our code is getting better or worse."

- Michael Feathers
Working Effectively with Legacy Code

AgitarOne JUnit Generator Feature Summary

Automated JUnit Test Generation

- Complements hand-written JUnit tests
- Creates JUnit tests that provide excellent coverage at the push of a button
- Automatic extensive mock object generation provides 80% or better coverage, even on complex code
- Automatic coverage analysis measures test effectiveness
- Automated JUnit test creation can be “trained” by writing simple Java helper methods
- JUnit generation based on award-winning Agitator® engine

Code-Rule Enforcement

- Automatically detect common coding errors and standards violations
- Easy to configure to match corporate standards
- Integrated with the dashboard for comprehensive reporting

Continuous Integration and Test

- Out-of-the-box automation for continuous development-level builds
- Continuous test execution finds and catches regressions quickly
- Based on CruiseControl, the most popular open-source build system
- Supports any version control system
- Easily integrated into an overall formal build process

Management Dashboard

- Powerful project summary shows testing status at a glance
- Automatic risk assessment helps prioritize high-value candidates for more testing
- Project summary shows complexity, usage, risk, test quality, coverage, and test pass/fail status over time
- Ownership identifies who is working on which classes
- E-mail notification provides immediate feedback of status or failures

Supported Platforms and System Requirements

Operating Systems: Windows XP Professional, Windows Server 2003, Windows Vista, or Linux with kernel 2.4.22 or later
Browsers: Firefox 1.0 or later, Internet Explorer 6 or later
JDKs: Sun 1.4, 1.5, 1.6; IBM 1.4, 1.5, 1.6
IDEs: Eclipse 3.2, 3.3, or 3.4-compatible IDE, including RAD and RSA 6.0, 7.0, and 7.5, and JBuilder 2007/2008
Frameworks: Supports J2EE 1.4, Java EE 5, and common Java frameworks including Struts, Spring, and Hibernate
Hardware: Do not install AgitarOne JUnit Generator on an old PC you found lying in a closet; buy the hardware needed to ensure good performance so you do not waste developer time!

- Fast Intel Pentium D equivalent CPUs (at least two, but it will take good advantage of many more)
- RAM – 2G or more
- 200 GB of free disk space

To find out more about how AgitarOne can help you produce faster, better, and more flexible code, visit

www.agitar.com

Worldwide Headquarters Contact Information:
+1 401-572-3150
Agitar Technologies, Inc., 41 Sharpe Drive, Cranston, RI 02920 U.S.A.
www.agitar.com

Agitar™ TECHNOLOGIES