

DATA SHEET

Automate .NET Development with Parasoft® .TEST™

Increase software development team productivity and software quality

Parasoft® .TEST™ is an integrated solution for automating a broad range of best practices proven to increase software development team productivity and software quality. Parasoft .TEST ensures developers that their .NET code works as expected by enabling coding policy enforcement, static analysis, and unit testing. Parasoft .TEST also saves development teams time by providing a streamlined manual code review process. Parasoft .TEST can be used both on the desktop as a Microsoft Visual Studio plugin and in batch processes via command line interface for regression testing.

Parasoft .TEST works with programming languages that target the Microsoft .NET Framework, including C#, VB.NET, ASP.NET and MC++ (Managed C++). It can test any file or assembly that has been built to take advantage of the .NET CLR. Moreover, it can test Web Site projects that are hosted in the file system or at an HTTP server.

Identify Runtime Bugs without Executing Software

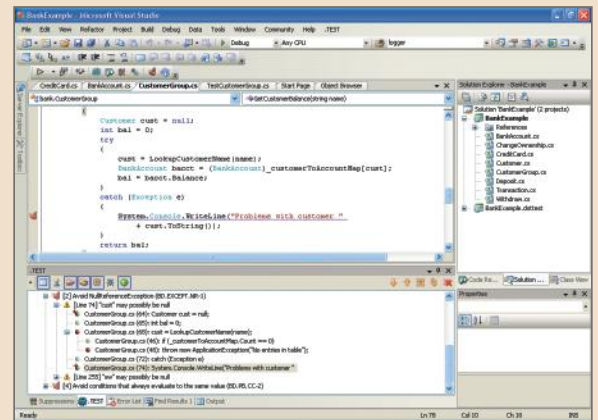
BugDetective uses data flow analysis to detect runtime errors without requiring the software to actually be executed. This enables early and effortless detection of critical runtime errors that might otherwise take weeks to find. Defects detected include NullReferenceExceptions, ArgumentNullExceptions, resource leaks, division by zero, dereferencing before checking for null, SQL injections, XSS, and other security vulnerabilities.

Such defects may also point to loopholes in the design of the code for the specific use cases corresponding to highlighted execution paths. Reviewing the violations reported by BugDetective often leads to good coding conventions that can prevent defects from recurring.

Automate Code Analysis for Compliance

A properly-implemented coding policy can eliminate entire classes of programming errors by establishing preventive coding conventions. .TEST statically analyzes code to check compliance with such a policy. To configure .TEST to enforce a coding standards policy specific to their group or organization, teams can define their own rule sets with built-in and custom rules. .TEST includes 400+ rules that cover Microsoft's .NET Framework Design Guidelines, CLS Compliance, Object Oriented Metrics, Security, and more.

In addition to rules that examine the IL code, .TEST also provides rules that examine the C# source code; this enables .TEST to check for many code issues that cannot be identified by IL-level analysis (for example, formatting issues, empty blocks, misuse of operators, etc.). Custom IL-level and C# rules, which are created with a graphical RuleWizard editor, can also enforce specific project and organizational requirements and prevent the recurrence of application-specific defects after a single instance has been found.



.TEST's BugDetective identifies critical bugs without executing the code.

Benefits

- **Increase team development productivity** – Apply a comprehensive set of best practices that reduce testing time, testing effort, and the number of defects that reach QA.
- **Achieve more with existing development resources** – Automatically vet known coding issues so more time can be dedicated to tasks that require human intelligence.
- **Build on the code base with confidence** – Efficiently construct, continuously execute, and maintain a comprehensive regression test suite that detects whether updates break existing functionality.
- **Gain instant visibility into code quality and readiness** – Access on-demand objective code assessments and track progress towards quality and schedule targets.
- **Reduce support costs** – Automate negative testing on a broad range of potential user paths to uncover problems that might otherwise surface only in “real-world” usage.

Features

- Static analysis of code for compliance with user-selected coding standards
- Graphical RuleWizard editor for creating custom coding rules
- Static code path simulation for identifying potential runtime errors
- Streamlined code review process with a graphical interface and progress tracking
- Automated generation and execution of unit tests
- Generates functional unit test cases that capture actual code behavior as the application is exercised
- Launches tests from the actual execution environment
- Flexible stub framework for use in unit tests
- Full support for regression testing
- Code coverage analysis for unit testing and beyond
- Full team deployment infrastructure for desktop and command line usage
- Seamless integration with Microsoft Visual Studio

Platforms

- Windows Vista, XP, or 2003

Compilers

- Visual Studio 2008, 2005, or 2003

Enable Effective and Comprehensive Team Code Review

The innovative Code Review module, which automates preparation, notification, and tracking of peer code reviews, addresses the known shortcomings of this very powerful development practice. .TEST automatically identifies updated code, matches the code with designated reviewers, and tracks the progress of each review item until closure. With the Code Review module, teams can establish a bulletproof review process—where all new code gets reviewed and all identified issues are resolved.

Automate Unit and Component Testing for Instant Verification and Regression Testing

.TEST's automated testing capabilities significantly reduce the work required to develop and maintain an effective test suite. .TEST's automated testing capabilities are especially helpful for supporting continuous integration and agile/iterative development.

.TEST automatically generates complete NUnit tests for any .NET language file or assembly. By using corner case conditions, these automatically-generated test cases check function responses to unexpected inputs, exposing potential reliability problems. Moreover, the generated test suite instantly establishes a baseline for regression testing, and can easily be extended to verify specific functionality that you want to verify. Collectively, these test cases establish a safety net that alerts you when modifications impact application behavior.

.TEST provides numerous ground-breaking technologies to facilitate unit testing, including:

- **Test Case Tracer:** Tracks and visualizes how designated classes/methods are used as a running application is exercised. You can select which calls you want included in test cases (and in what order), then click a button to generate flexible NUnit tests that capture the precise functionality you want to validate. This significantly reduces the time and effort required to develop realistic functional tests.
- **Application hosted testing:** Allows you to launch unit tests from virtually any point within your application—without changing your application or writing additional code. This lets you create complex objects in their natural environment and facilitates test development/maintenance.
- **Extensive coverage analysis:** Tracks coverage information for all tests—from .TEST-based unit testing to manual application testing—and can combine the coverage information from multiple test runs. This helps you accurately gauge test suite efficacy and completeness, as well as demonstrate compliance with test and validation requirements.
- **Flexible stub support:** Allows classes to be tested in isolation. This addresses one of the greatest challenges in writing unit tests: getting complex objects in different states.

Establish a Uniform Process Across the Team with a User-Friendly Workflow

An easy-to-use workflow and a good deployment strategy are both essential for making the above features work in real-world development processes. .TEST's support for team deployment standardizes testing team-wide and provides a sustainable workflow for integrating best practices into the team's existing processes—with minimal disruption. The architect defines the team's designated test configurations, then Parasoft Team Server automatically shares them across all team .TEST installations. Developers can test code directly from Visual Studio to find and fix problems before adding it to source control. Additionally, .TEST Server can test the entire project code base each evening, then email reports to the manager and responsible developers if any problems are detected. Developers can then import results into Visual Studio to review and repair the errors reported for code they authored.

.TEST also sends information from these tests to Parasoft Report Center, which provides interactive Web-based dashboards with drill-down capability, allowing teams to track project status and trends based on .TEST results and other key process metrics.

www.parasoft.com

Contact info:

Parasoft Corporation, 101 E. Huntington Dr., 2nd Flr., Monrovia, CA 91016

Ph: (888)305.0041, Fax: (626)256.6884, Email: info@parasoft.com