Microsoft

MICROSOFT® VISUAL STUDIO 2010

Overview

Microsoft® Visual Studio® 2010

VISUAL STUDIO 2010 DELIVERS THE FOLLOWING KEY ADVANCES:

ENABLING EMERGING TRENDS

Every year the industry develops new technologies and new trends. With Visual Studio 2010, Microsoft delivers tooling and framework support for the latest innovations in application architecture, development and deployment. Two of the major advances in development trends occurring soon are the extension of development to the cloud and the ability to easily construct applications that use the latest multi-core hardware in parallel.

Cloud Development

On October 27th 2008 we announced Windows® Azure™ the comprehensive cloud environment from Microsoft. With Windows® Azure™ Tools for Microsoft® Visual Studio® you can build, debug and deploy services and applications for Windows Azure.

Windows Azure offers a scalable hosting environment for the Internet, built on geographically distributed data centers. It handles load balancing and resource management, and automatically manages the life cycle of a service based on requirements that you establish. With the service, you include code specifications for the service topology, the number of instances to deploy, and any configuration settings. Windows® Azure™ deploys the service and manages upgrades and failures to maintain availability.

The Windows Azure environment is designed as a utility computing model, so that you pay only for the resources used by your service, while benefitting from the reliability and performance provided by the hosting environment.

Windows Azure Tools provide the means to create services and applications within the framework of Visual Studio. That includes a project model specifically for Windows Azure, as well as the debugging capabilities of Visual Studio. With Visual Studio, you can build a package containing your service, and use Windows Azure Tools to deploy the package to Windows Azure through the Windows Live Developer Portal.

Parallel Development

As demands for application performance increased, customers have traditionally solved the problem by simply increasing the underlying power of the hardware that the application is running on. Over the last several years developers have seen the CPUs that their applications run on start to include 2, 4 or more cores. While the power of the hardware has increased, the transition to a multi-core environment has impacted the applications that developers write. The majority of applications will not be able to automatically take advantage of this multi-core hardware change. Developers will need to modify the way they write applications and the architectures they use for these applications.

Creating parallel capable code using current technologies is unfortunately not trivial. Multi-thread programming introduces not only application architecture challenges to complexity and robustness but also exposes the tooling developers use as being optimized for single-threaded development.

Microsoft is making a major commitment to make parallel development accessible to a wide range of developers, whether they are using native code or the .NET Framework. With Visual Studio 2010 we are delivering:

- **Visual Studio IDE support for Parallel development**
- **Native C++ libraries and compiler support for Parallel applications**

The .NET Framework 4.0 also provides the core framework support to build parallel applications through technologies such as P-LIINQ and parallel language semantics and framework components. Visual Studio 2010 provides integrated parallel development support. In Visual Studio 2010 the debugger is aware of the parallel nature of code and can present the state of the application execution during debugging across the different parallel execution units. The debugger also has custom displays for parallel code such as task & thread windows and a “multi” or “cactus” stack view window that graphically shows the execution path of the individual tasks.

Being able to develop and debug your application doesn’t mean that it takes advantage of all the available power. To help developers do this, Visual Studio 2010 also includes a parallel capable performance analyzer that enables you to extensively instrument you code to visually see the concurrency issues that are in your applications. Combine this with the features of the Visual Studio IDE, and developers have a highly productive, visual environment for building the best parallel capable applications available.

INSPIRING DEVELOPER DELIGHT

Ever since the first release of Visual Studio, Microsoft has set the bar for developer productivity and flexibility. Visual Studio 2010 continues to deliver on the core developer experience by significantly improving upon it for roles involved with the software development process.

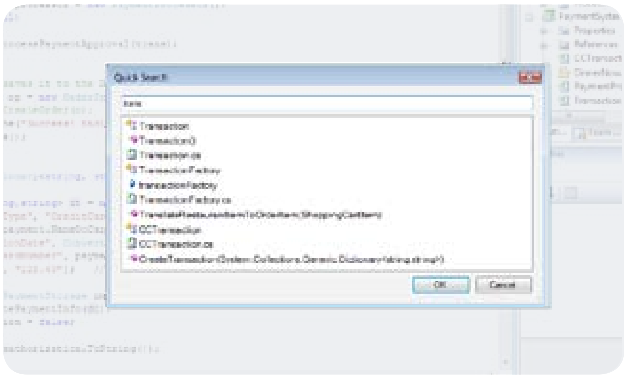
Understanding existing, and writing new, code

As the complexity of applications grows so does the challenge of understanding the code that you're working on. With Visual Studio 2010 the IDE provides integrated support for understanding what is happening in the code section that you're viewing.

The editor in Visual Studio 2010 has been rebuilt using the Windows Presentation Foundation (WPF) technology. WPF enables the editor to richly present information about the code in the context of presenting the actual source. This ability enables features such as the "Document Map Margin" to render a graphical view of the source file including information such as layout, code coverage, symbol highlights and comments.

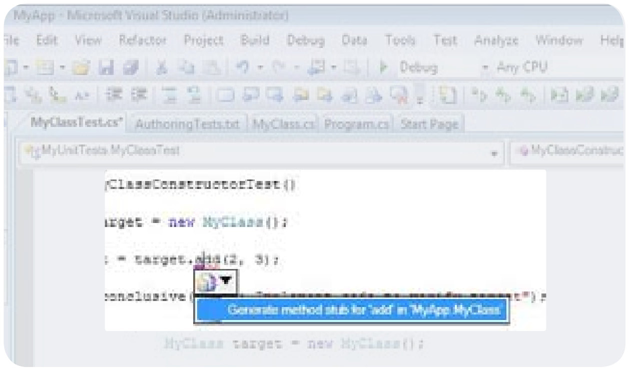
This editor ability also enables 3rd parties to create add-ins that show custom views of the underlying source file such as taking the XML Doc Comments and converting them to a rich presentation formation with fonts, colors and highlighting. It enables Visual Studio to display different layers on the editor so an add-in could represent a code-based formula in its traditional mathematical representation.

While the representation of the underlying source code is important so is the ability to understand what the code is actually doing. In Visual Studio 2010, features such as "Inline Call Hierarchy" - a feature which enables a developer to select an entity or method and see how the code calls inwards or outwards or passes the entity in and out of the code section - provide developers with the ability to understand the interaction of the code without needing to juggle multiple files. Other features such as "Highlight References", which provide a visual representation of the references to a selected entity in the code without needing to use the "Find In Files" feature, or "Quick Searching", which delivers a 'word wheel' based search tool integrated with "Highlight References", enable developers to maintain the context of where they are but gain the understanding of other locations in the code.



QUICK SEARCH EDITOR HIGHLIGHTS REFERENCES

Additionally the editor integrates with the project system to simplify the pattern of Test Driver Development (TDD). With TDD, developers build the tests that will exercise their application code before they actually write that code. In Visual Studio 2010 developers can create tests and the editor will provide functionality to automatically implement the tested classes and code in the file the developer chooses. This enables developers to quickly create the class they are consuming without needing to break out of the test development flow to declare the tested class.


MyApp - Microsoft Visual Studio (Administrator)
File Edit View Refactor Project Build Debug Data Tools Test Analyze Window Help
Debug
Any CPU
MyClassTest.cs*
AuthoringTests.txt
MyClass.cs
Program.cs
Start Page
MyUnitTests.MyClassTest
MyClassConstruc
Generate method stub for 'add' in 'MyApp.MyClass'

```
yClassConstructorTest()

arget = new MyClass();

 = target.add(2, 3);

onclusive(
```

TEST DRIVEN DEVELOPMENT– CONSUME FIRST, DECLARE SECOND

Web Development

With ASP.NET, Microsoft delivered a ground breaking productive development model that made web applications accessible to the traditional application developer. Over the last few releases, not only has the tooling in Visual Studio improved to provide developers with a more web standard set of tools, but leading features like CSS property grids and split view design surfaces have been provided.

However the industry evolves, and so do the tool requirements and patterns that developers use. Web developers are leading the push to split content from data and to use a Test Driven Development (TDD) methodology. In Visual Studio 2010, we deliver the next generation of ASP.NET web tools that make it easy for developers to use TDD to build Model-View-Controller (MVC) based web sites.

Many ASP.NET Developers have already experienced the preview release of ASP.NET MVC. All the features in that release are included in Visual Studio 2010. Ranging from Project Templates and Solutions that natively describe an ASP.NET MVC website, to automatic generation of test projects in the web solutions, to wizard support for common tasks like creating views from controllers and snippet support for HTML Markup, the Visual Studio IDE delivers all the support required.

In Visual Studio 2008, we invested heavily in supporting JavaScript in the Visual Studio IDE and debugger. In Visual Studio 2010, we're continuing that investment with a higher performance, and standards - compliant JavaScript IntelliSense engine. These investments enabled Microsoft to announce their involvement with the JQuery group, and Visual Studio 2010 will be the first version of Visual Studio to ship JQuery as a native part of the ASP.NET solution set.

Deployment of websites has been a challenge for developers for many years. Visual Studio 2010 has full IDE support for a simplified deployment process for ASP.NET websites. Called "One Click Deployment", this process and IDE support provides a wizard, dialogs and design surfaces that make it simple for developers to identify the components of a website that need to be deployed, and handle the process of moving them from the development machine to the web server, whether that is an internal server for the organization or a server hosted by a 3rd party site.

"One Click Deployment" also solves the problem of changing the settings of a website from the development machines to the final deployed site. Many times developers have sent websites to deployment with debug tracing turned on or the database connections set to the development servers. With web.config transformations, "One Click Deployment" enables a developer to create a custom set of transforms that will be applied to the website every time it is deployed and ensures that the appropriate settings are in the configuration files.

Additionally, Microsoft has just released the Silverlight 2 runtime and tooling for Visual Studio 2008. In Visual Studio 2010, Silverlight is fully supported for developers wishing to build Silverlight content. Having design surfaces for Silverlight enables developers to either author original content or to modify content as part of the designer-developer workflow that Visual Studio enabled in the last release. Visual Studio 2010 also provides full debugging support for Silverlight and provides project system integration for developers consuming this content in various applications types. For example, web developers building ASP.NET websites will be able to include existing Silverlight content, and Visual Studio will create the appropriate test pages and content includes to enable them to focus debugging on the Silverlight content in the context of the overall website solution.

C++ Development

Visual Studio 2010 marks a major renovation of our C++ IDE so that it not only supports emerging trends like parallel computing, cloud and web services, but also provides a first-class C++ development experience through an IDE that scales to the large size of code bases that are typical of C++ sources. We're also adding a significant focus on building great experiences for navigating and understanding complex C++ source bases to enable developers to figure out the best places to make source changes in their complex systems.

In Visual Studio 2010, the C++ project system has been converted to MSBuild based system that enables developers to take their existing C++ solutions that are currently sectioned to be manageable and bring them into a single solution that provides full IDE support for all the assets in the solution. With a full compiler backing IntelliSense, and a true database-based symbol system, this enables developers to work with large solutions, thousands of files, and up to 4GB of symbol information.

Visual Studio 2010 provides a C++ IDE experience that includes the return of the MFC Class Wizard, the ability to view large source files through Source Outline, integrated quick searching to find information without the confusion of the current "Find In Files" method and an easily extensible IDE model through the new Managed Extensibility Framework (MEF).

RIDING THE NEXT GENERATION PLATFORM WAVE

Microsoft continues to invest in the market-leading operating system, productivity application and server platforms to deliver increased customer value in these offerings. With Visual Studio 2010, customers will have the tooling support needed to create amazing solutions around these technologies.

Windows 7 Development

In Visual Studio 2010, we've invested heavily in C++ to make developing native Windows applications easier and more productive. We are adding tools to assist developers in building new Windows 7 applications and in making existing native applications take advantage of new Windows features. We're including full library and header support for Windows 7, significant updates to MFC to support Windows 7 UI elements like the ribbon, live icons, search access and even support for multi-touch enabled interfaces.

For developers building WPF based applications, Visual Studio 2010 delivers improvements to the WPF design surfaces with richer graphical editing features, better alignment to underlying WPF functionality and integrated data binding from the properties grid and data sources windows.

Office Business Application Development

Visual Studio 2005 delivered the first release of Visual Studio Tools for Office. Since then, Microsoft Office development has become an integrated component of Visual Studio, and as Office moves to deliver a client and server experience, so too Visual Studio.

In Visual Studio 2010, developers will be able to build Office client applications that span multiple versions of Office, either 32- or 64-bit, and deliver these as a single deployment package. The creation of the deployment packages is assisted through the provision of a deployment design surface that developers can use to graphically assemble the package that the end-user will install. Not only is the creation of the package easier by the ability to leverage "ClickOnce", CD or Web installs enable developers and IT Professionals to use the appropriate technology to get these applications onto the end-user machines.

The task of building the applications themselves has also gotten easier with the introduction of designer support for building flexible UI in either WPF or Fluent. With these designers, developers will be able to customize the Office File Menu, Task Panes, Outlook Form Regions and the Fluent UI itself (such as the ribbon view). Additionally, the task of consuming data in Office applications is made easier through improved data binding, integration of various Office data sources with LINQ and the ability to data bind to the Business Data Catalog. Finally applications will be able to interoperate with the many Office objects such as lists and action panes, and also participate with the Office Live Viewer.

LEARN MORE

http://www.microsoft.com/visualstudio/2010

MICROSOFT® VISUAL STUDIO® TEAM SYSTEM 2010

DEMOCRATIZING APPLICATION LIFECYCLE MANAGEMENT

Visual Studio Team System 2010 will deliver new capabilities that embrace the needs of the users in the lifecycle – from architects to developers, from project managers to testers.

Among the great new functionality in Visual Studio Team System 2010:

- Discover and identify existing code assets and architecture with the new Architecture Explorer.
- Design and share multiple diagram types, including use case, activity and sequence diagrams.
- Improve testing efforts with tooling for better documentation of test scenarios and more thorough collection of test data.
- Identify and run only the tests impacted by a code change easily with the new Test Impact View.
- Enhanced version control capabilities including gated check-in, branch visualization and build workflow.

Key to a shared understanding of the application is the use of modeling tools. Modeling has traditionally been done by professional architects and system designers. Our approach is to enable both technical and non-technical users to create and use models to collaborate, and to define business and system functionality graphically.

Product Overview

The marketplace has begun to mature and accept Application Lifecycle Management (ALM) as a proven discipline for creating high-quality applications. However, existing solutions in the marketplace have not kept pace with the changing needs of technical users and the expanding inclusion of non-technical users as part of the lifecycle.

Every customer today faces a similar set of business problems:
- How do we build high quality applications that deliver real business value?
- How do we embrace the Application Lifecycle Management model effectively?
- How can we ensure that all members of the team – both technical and non-technical – are part of the process?
- How can we get the most value from our existing code assets?
- How do we make powerful modeling tools available to everyone in the application lifecycle?

The third generation of Visual Studio Team System – Visual Studio Team System 2010 – will be a robust and streamlined solution that addresses these needs and concerns.

We are evolving Application Lifecycle Management by:
Building quality into the lifecycle
- Ensuring architectural consistency through the lifecycle
- Eliminating "No-Repro" bugs
- Ensuring smooth build handoffs and high quality builds
- Incorporating performance in the lifecycle

Driving efficiency into the test effort
- QA Team aligned with Business Analysts, Architects, and Developers
- Eliminating tedious tasks
- Improving setup and deployment of tests
- Choosing the right tests

Ensuring Complete Testing
- Focused test planning and progress tracking
- Transparently see the quality of requirements and level of testing
- Finding the gaps in testing and fill them
- Ensuring changes are properly tested

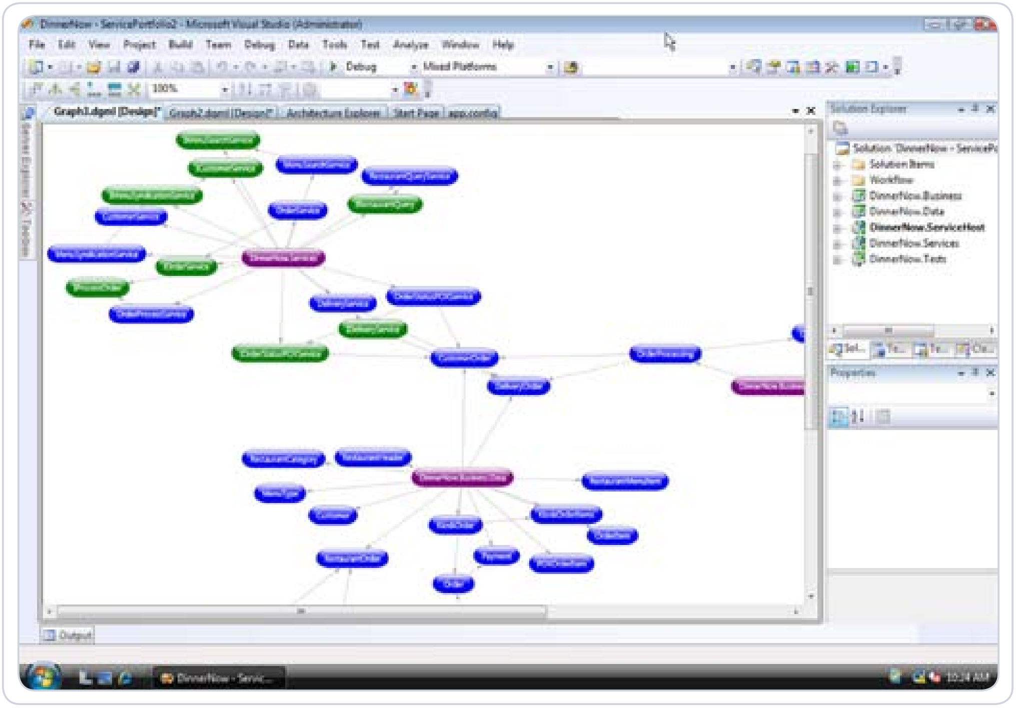
Modeling that Works with Code

For most businesses only 20% of the code being written today is for new applications; the majority of work is being done on existing code bases.

When working on existing code, architects and developers have not necessarily had good enough tools to understand the system, know what needs to be done to make required updates, or been able to anticipate the impact of changes made. Often it isn't until much later that an unexpected bug is discovered as a result of a change.

Our modeling tools have tight integration into the actual code of the application. This means that a developer or architect can use models to explore existing code assets. The new Architecture Explorer in Visual Studio Team System gives developers and architects the capability of creating a full architectural picture of existing code; understanding how they fit together; understanding how they "work." This leads to better information about using, re-using, or discarding existing code. The Architecture Explorer provides architects and developers a mechanism for visualizing code assets in a number of ways including graphs, stacked diagrams and dependency matrices.

The introduction of the Architecture Layer Diagram means that a developer or architect can use models to enforce constraints on code as well. The Architecture Layer Diagram can be coupled to code making it an active diagram that can be used for validation. For example, when an architect designs a system where the presentation layer should not talk to the data layer, you want to be able to enforce that model at check-in. Visual Studio Team System 2010 can do that. These capabilities delivered in VSTS 2010 are part of the Microsoft's overall modeling story.



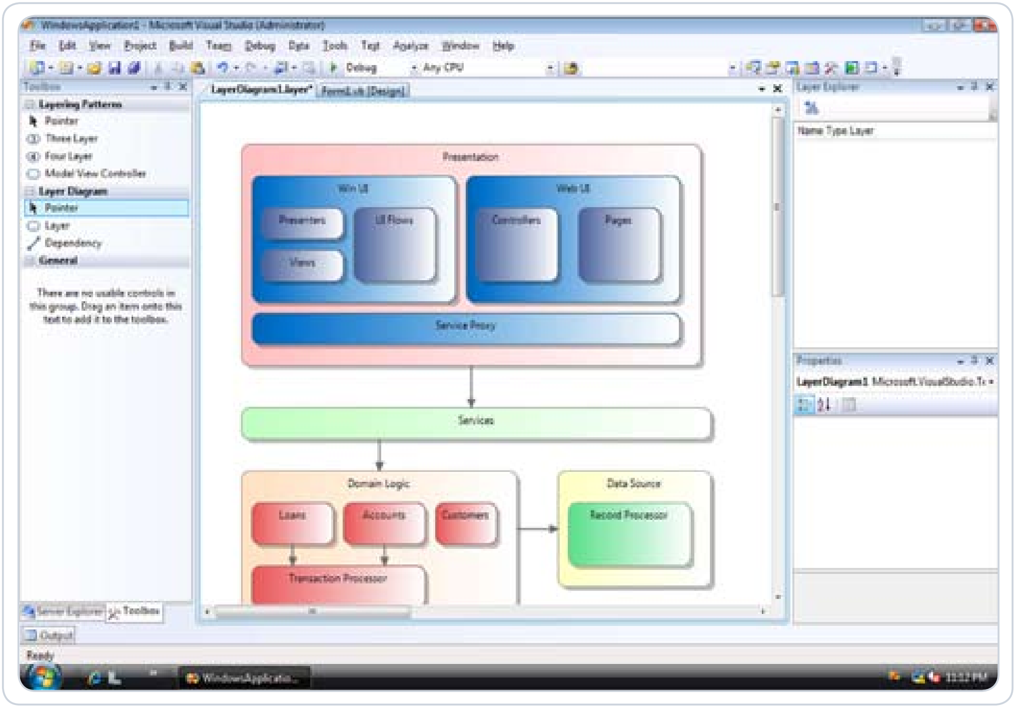
THE NEW ARCHITECTURE EXPLORER ENABLES INDIVIDUALS TO CREATE A VISUAL REPRESENTATION OF EXISTING CODE ASSETS.

Eliminating "No-Repro" Bugs

For From designing an application through developing code, finding bugs that can't be reproduced is a common problem – the "no-repro" bug. Many factors drive these types of bugs, and we've worked to create tools to help isolate the issue and enable faster fixes.

One way this is solved in Visual Studio Team System 2010 is with the use of a tool that can specify the exact state of the build used by a tester and allow a comparison to the state of the build used by the developer when trying to reproduce the bug. It is often the subtle differences between these two that create the no-repro state, and a new tool within Visual Studio Team System 2010 has been designed to specifically address this.

This tool – the Microsoft Test Runner – is a standalone tool that a tester uses to guide him through a series of steps to complete a test case. When the test case is started, the Microsoft Test Runner takes a snapshot of the system data, including OS version and Service Pack and other pertinent system data. As the test is being run, the tester can use the tool to capture images of the application under test, or even partial or full screen video of the test being run. If an issue is discovered, the tester can create a new bug in Team Foundation Server and attach these artifacts. When attached, the screen capture video is fully indexed with the test steps as bookmarks, making it easier for the developer to see what went wrong on the tester's machine. All of these artifacts help to eliminate the no-repro scenario, and help build a better bridge between development and test.



THE MICROSOFT TEST RUNNER GIVES TESTERS A SET OF DEFINED VALIDATION STEPS TO EASILY CREATE AN ACTIONABLE BUG, INCLUDING SYSTEM INFORMATION, SCREEN IMAGES, AND A FULLY INDEXED SCREEN CAPTURE VIDEO.

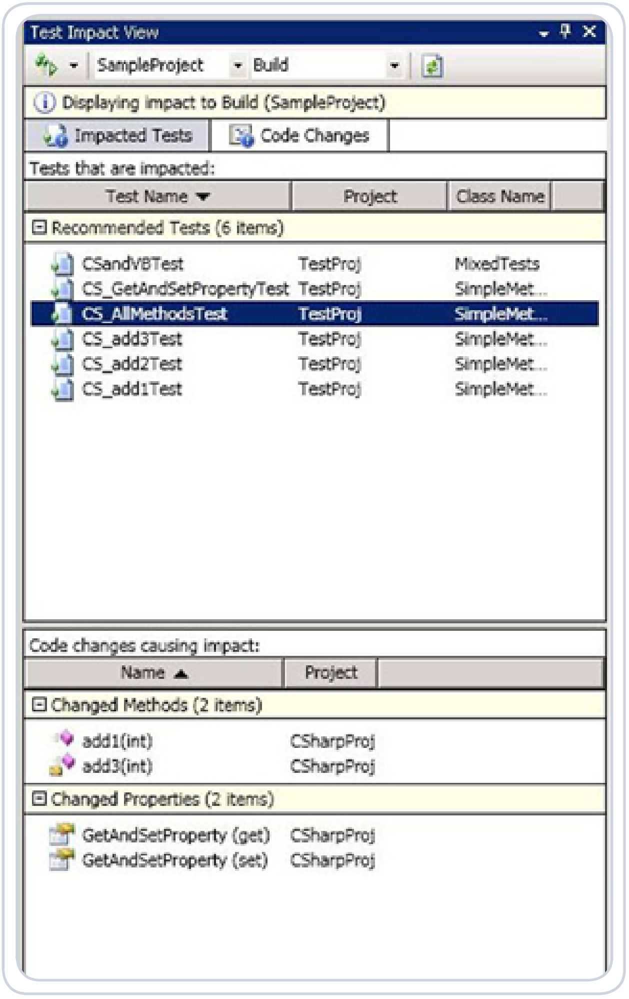
Identify the Test Impact

As developers make changes to code, it's critical for them to effectively test their changes – not only to prove the new code works, but to ensure there's no unexpected downstream effect. Test impact analysis and test prioritization identify the tests that must be run to validate the code changes. This helps developers quickly check-in code with confidence by running only the necessary tests and reduces churn created by unexpected failures.

The new Test Impact View window enables a developer to view a list of tests that need to be run as the result of a code change. The developer can toggle between an Impacted Tests view and a Code Changes view.

- The Impacted Tests view provides a list of tests that need to be run and which code changes are covered by each of the tests.
- The Code Changes view provides a list of code changes and which tests must be run in order to validate each of them.

These two views provide an easy way to discover what tests must be run in order to validate the changes to the code base without having to run all of the tests. This ensures that all changes are tested effectively.



Displaying impact to Build (SampleProject)

Tests that are impacted:

Test Name	Project	Class Name
Recommended Tests (6 items)		
CSandVBTest	TestProj	MixedTests
CS_GetAndSetPropertyTest	TestProj	SimpleMet...
CS_AllMethodsTest	TestProj	SimpleMet...
CS_add3Test	TestProj	SimpleMet...
CS_add2Test	TestProj	SimpleMet...
CS_add1Test	TestProj	SimpleMet...

Code changes causing impact:

Name	Project
Changed Methods (2 items)	
add1(int)	CSharpProj
add3(int)	CSharpProj
Changed Properties (2 items)	
GetAndSetProperty (get)	CSharpProj
GetAndSetProperty (set)	CSharpProj

THE TEST IMPACT VIEW IDENTIFIES ALL THE TESTS THAT SHOULD BE RUN TO VALIDATE A CODE CHANGE.