DevOps with TFS

Lesson 04 : Unit Testing and DevOps

Table of Contents

Table of Contents

- Developer Tools and Unit Testing
  - An Overview and Capabilities of Tools
  - Unit Testing Steps
  - Unit Testing Tools
  - Generate Unit Test with Intellitest
  - Code coverage Tools

Test with VSTS – An Overview

- Visual Studio Testing Tools enables development teams ensure higher quality applications and adopt latest testing practices
- Testing tools provide insightful information enabling development teams to reproduce and fix issues sooner and faster
- Ensuring higher quality applications and a better customer experience
- The testing team can be more productive while planning, executing and tracking tests,
  - With the web-based test tools using Visual Studio Team Services or
  - With the rich experience provided by Visual Studio and Microsoft Test Manager.

Team Services and Team Foundation Server provide rich and powerful tools everyone in the team can use to drive quality and collaboration throughout the development process. The easy-to-use, browser-based test management solution provides all the capabilities required for planned manual testing, user acceptance testing, exploratory testing, and gathering feedback from stakeholders.

Testing Types

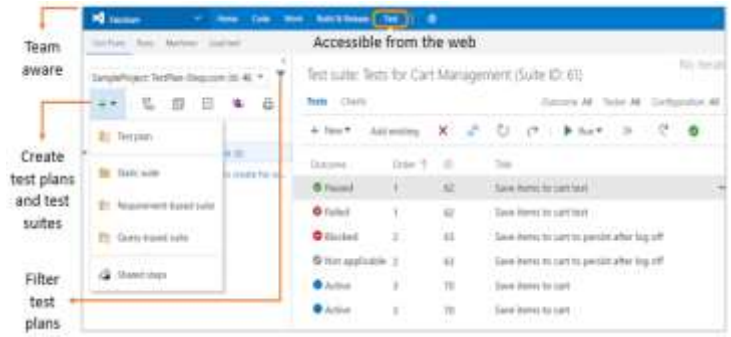
Planned manual testing

- Manual testing has evolved with the software development process into a more agile-based approach.
- Team Services and Team Foundation Server integrate manual testing into your agile processes.
- The team can begin manual testing right from their Kanban boards in the Work hub.
- Test hub provides advanced capabilities for all the test management needs.

Test hub in Team Services and Team Foundation Server provides a rich test management solution for teams that need advanced manual testing capabilities.
The Test hub includes all the capabilities required for the testing lifecycle - including test planning, authoring, execution, and tracking

Testing Types

- Steps involved in Manual Testing
  - Test planning
  - Test authoring
  - Testing web applications
  - Testing desktop apps
  - Test tracking



Test planning

Create and manage test plans and test suites

Test authoring

Create multiple test cases in one operation

Add existing test cases to a test suite.

Assign single or multiple testers to execute the tests.

Testing web applications

The Test hub provides a browser-based test runner to run tests for your web apps.

Testing desktop apps

Test your desktop apps with Microsoft Test Runner client, which is part of Microsoft Test Manager.

Test tracking

Quickly configure lightweight charts to track your manual test results using the chart types of your choice, and pin the charts to your dashboard to easily analyze these results.

Testing Types

User acceptance testing

- User acceptance testing (UAT) is a key factor in software development that ensures the value requested by customers is being delivered by the engineering team.
- Team Services and Team Foundation Server include capabilities and tools to manage user acceptance testing.
- Quickly create UAT plans and suites and invite multiple testers to execute these tests using test artifacts provided by the engineering team.
- Easily monitor UAT progress and results using lightweight charts.

Testing Types

Exploratory testing for everyone

- Maximizing quality in modern software development processes is a shared responsibility between developers, managers, product owners, user experience teams, and more.
- Collaborative testing processes and tools are the key factors in driving quality in these scenarios.

Team Services and Team Foundation Server provide a lightweight, browser-based extension called the Test & Feedback extension, which enables everyone to contribute to the quality of your web apps.

Testing Types

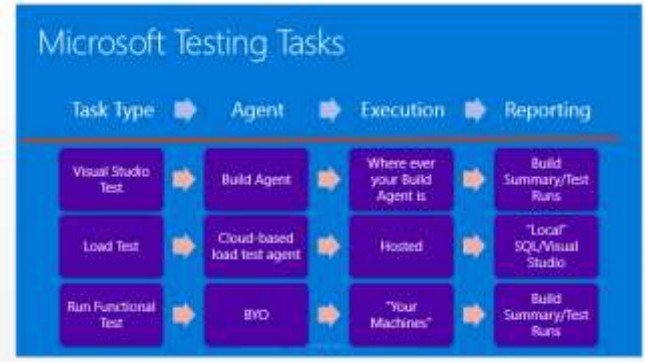
Stakeholder feedback

- Seeking feedback from stakeholders outside the development team, such as marketing and sales teams, is vital to develop good quality software.
- Using Team Services and Team Foundation Server, developers can request feedback on their user stories and features.

Stakeholders can respond to feedback requests using the browser-based Test & Feedback extension - not just to rate and send comments, but also by capturing rich diagnostic data and filing bugs and tasks directly.

Key Features of Testing in VSTS

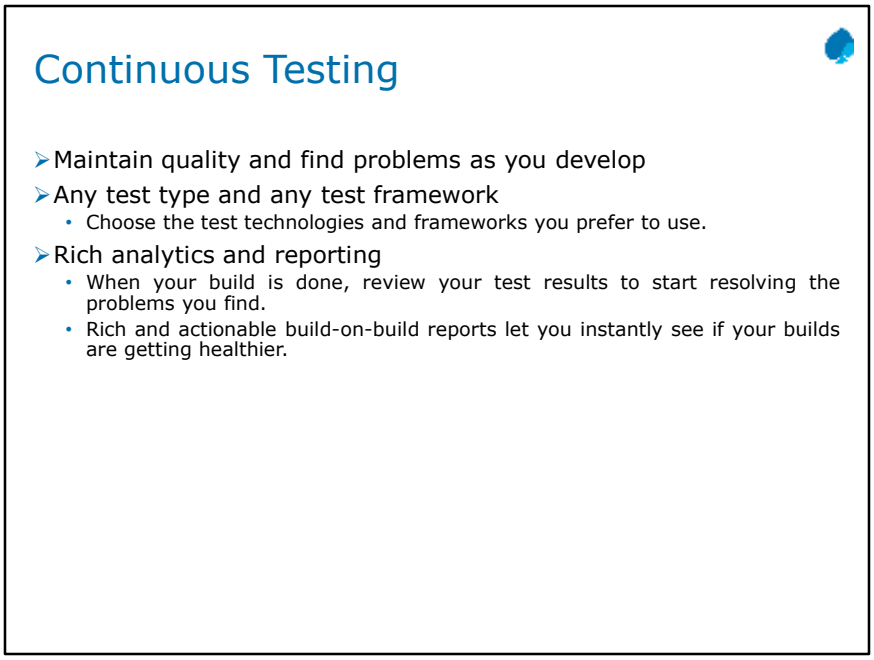
- Test on any platform.
- Rich Diagnostic data collection.
- End to End Traceability.
- Extensible platform.



Test on any platform. With the Test hub in Team Services and Team Foundation Server, you can use your browser to access all the manual testing capabilities. The Test hub enables you to create and run manual tests through an easy-to-use, web-based interface that can be accessed from all major browsers on any platform.
Rich Diagnostic data collection. Using the web-based Test Runner and Test Runner client you can collect rich diagnostic data during your tests. This includes screenshots, an image action log, screen recordings, code coverage, IntelliTrace traces, and test impact data for your apps under test. This data is automatically included in all the bugs you create during test, making it easy for developers to reproduce the issues.
End to End Traceability. Team Services and Team Foundation Server provide end-to-end traceability of your requirements, builds, tests and bugs. Users can track their requirement quality from cards on the Kanban board. Bugs created while testing are automatically linked to the requirements and builds being tested, which helps you track the quality of the requirements or builds.
Extensible platform. You can combine the tools and technologies you already know with the development tools that work best for you to integrate with and extend TFS. Use the REST APIs and contribution model available for the Test platform to create extensions that provide the experience you need for your test management lifecycle.

Continuous Testing

- Test continuously while you code, build, and deploy your app.
- Find problems before launching your app or updates into production.
- Better to assess whether your app meets the customers' needs and is ready for release.
- Whether your app is on-premises or in the cloud,
  - you can automate build-deploy-test workflows
  - choose the technologies and frameworks,
  - test your changes continuously in a fast, scalable, and efficient manner.



Maintain quality and find problems as you develop

Continuous testing with Visual Studio Team Services or Team Foundation Server ensures your app still works after every check-in and build, enabling you to find problems earlier by running tests automatically with each build. Make sure that your app still works after every check-in and build using Visual Studio Team Services.

Find problems earlier by running tests automatically with each build. When your build is done, review your test results to start resolving the problems that you find.

Developer Tools and Unit Testing

- Maintain code health with unit testing.
- Visual Studio and Visual Studio Team Services provide a wide range of powerful tools and techniques for developers to use when testing applications.

- Capabilities of Testing Tools
  - Avoid regressions and achieve code coverage with IntelliTest
  - User interface testing with Coded UI and Selenium
  - Effective unit testing with Visual Studio Code Coverage
  - Unit testing with any framework using the high performance Test Explorer

Developer Tools and Unit Testing

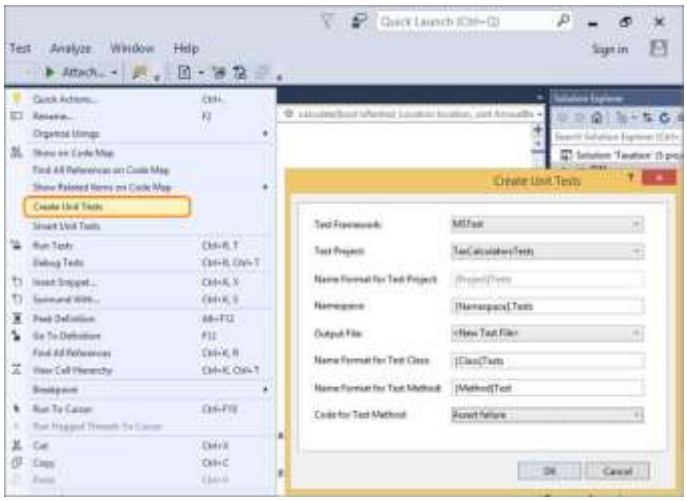
- Using Visual Studio one can define and run unit tests to maintain code health, ensure code coverage, and to find errors and faults before customers do.
- Unit Testing in Visual Studio can be done with the following steps

Step 1 : Create unit tests

- Create unit tests and run them frequently to make sure your code is working properly.
- Different unit test framework can be used to create tests for different code languages.

Adapter/Framework	Language
Boost / Google / xUnit++	C++
MbUnit /nUnit /xUnit.net	C#
Python Tools for Visual Studio	Python
Silverlight	Silverlight
TSTestAdapter	TypeScript
VsNodeTest	Node.js

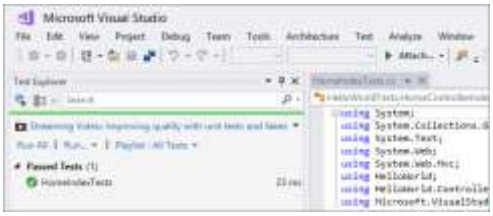
Developer Tools and Unit Testing



Developer Tools and Unit Testing

Step 2 : Run unit tests

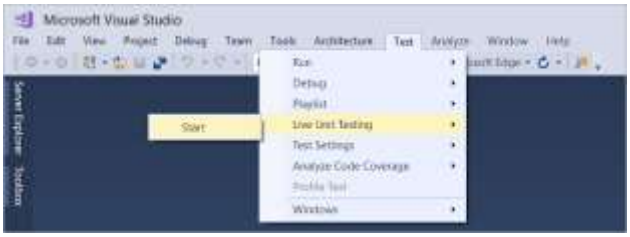
- Open Test Explorer.
- Run All Tests and see the unit tests that passed or failed in Test Explorer.



Developer Tools and Unit Testing

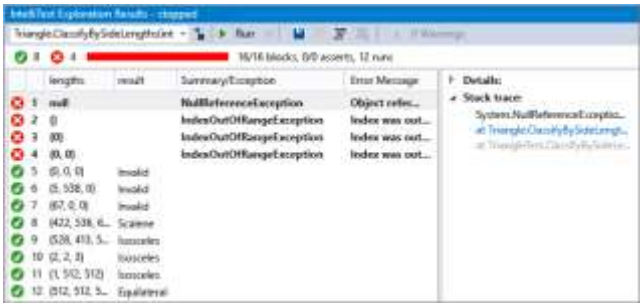
Step 3 : View live unit test results

- If you are using the MSTest, xUnit, or NUnit testing framework in Visual Studio 2017 or above, you can see live results of your unit tests within the Visual Studio UI.
- Turn on live unit testing from the **Test** menu.



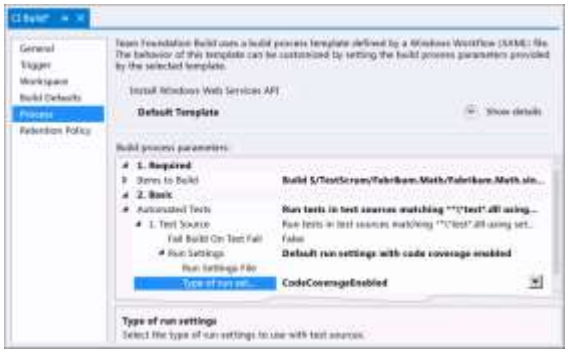
Developer Tools and Unit Testing

- Generate unit tests with IntelliTest
  - When you run IntelliTest, which tests are failing and add any necessary code to fix them.
  - You can select which of the generated tests to save into a test project to provide a regression suite.
  - As you change your code, rerun IntelliTest to keep the generated tests in sync with your code changes.



Developer Tools and Unit Testing

- Use code coverage to determine how much code is being tested
  - To determine what proportion of your project's code is actually being tested by coded tests such as unit tests, you can use the code coverage feature of Visual Studio.
  - To guard effectively against bugs, your tests should exercise or 'cover' a large proportion of your code.



Summary

- Testing in VSTS
- Types of Testing
- Features offered in VSTS
- Continuous Testing
  - Need and Advantages
- Developer Tools in VSTS
- Unit Testing Framework
- Steps and Tools for Unit Testing in VSTS