C# Automation: 1

Software quality is the responsibility of every member on the development team. Learn to implement quality from the ground-up as well as build checks for existing software. Create a complete software automation system that builds, tests, and reports the results. Apply unit-testing to your software to ensure that code meets its design specifications and behaves as intended. Gain practical, hands-on experience with popular test frameworks and use them along with your knowledge of C# to create enterprise-quality automation applications with minimum friction.

Who should take this course?

Required for students taking the Software Test Engineer Certificate, this class is ideally suited to software testers or software design engineers in test wanting to expand their career in the direction of test automation using Microsoft .NET platform.

Course Objectives

• Describe the role of automation in the software development lifecycle.
• Build a basic automation application to test a library.
• Build and test your own class by leveraging the features of the NUnit test framework.
• Create a User Interface (UI) automation library using the White UI test framework.
• Create a web UI automation application using the Selenium browser test framework.
• Create a simple automated solution that builds a .NET application, runs automated tests, and reports the results using C# and scripting.

Course Details

• Length: 30 hours
• Format: Classroom
• Prerequisites: C# Programming: 2 and Structured Query Language: (SQL) Level 1

The above prerequisites are considered to be the basic skills and knowledge needed prior to taking this class. Instructors will assume your readiness for the class materials and will NOT use class time to discuss prerequisite materials.
Course Contents

Describe the role of automation in the software development lifecycle.

• Describe automation, its strengths and weaknesses, and how automation fits into the process of creating software.
• Describe a basic build system and how automation contributes to the development process.
• Identify the business stakeholders of an automation system.
• Describe the following types of automated tests: acceptance, functional, unit, stress, and performance.

Build a basic automation application to test a library.

• Create a simple automation framework application that tests a library.
• Add and run automated tests for simple math operations.
• Refactor the harness code to load tests from a text file.
• Create a test result summary that outputs to a file.

Build and test your own class by leveraging the features of the NUnit test framework.

• Describe what NUnit is, and how to use it for TDD and automated tests.
• Create a NUnit-based library that tests a class.
• Flag a test case to expect an error.
• Debug tests using a NUnit runner, NUnit-console, and a Visual Studio plugin.
• Generate both XML and plain-text result summary files from NUnit.

Create a User Interface (UI) automation library using the White UI test framework.

• Describe the test layers of a Windows application.
• Create a NUnit-based library that leverages the White UI test framework.
• Create a test that starts up and shuts down a window-based application.
• Create a test that populates and submits a Windows form.
• Create a test that manipulates a modal window.
Course Contents, continued

Create a web UI automation application using the Selenium browser test framework.

- Describe the test layers of a web application.
- Create a NUnit-based library that leverages the Selenium browser test framework.
- Create a test that accesses a website in a browser and closes the browser.
- Create a test that populates and submits a web form.
- Create a test that manipulates a popup browser window.
- Create an end-to-end test case that accepts a specific input and verifies a change occurred in the database.

Create a simple automated solution that builds a .NET application, runs automated tests, and reports the results using C# and scripting.

- Describe the differences between Command Line Interface (CLI), Windows Scripting Host (WSH), and Windows Management Instrumentation (WMI).
- Create a script that builds and tests an application.
- Create a script that emails build results.
- Connect to a remote machine and execute commands remotely.
- Add a scheduled task to run a script.