

**ASP.NET 8 Test-Driven Development**

**Course Number:** NET-324  
**Duration:** 4 days

**Overview**

This ASP.NET Test-Driven Development (TDD) training course teaches TDD techniques with ASP.NET web applications. The course covers ASP.NET MVC development, including how to unit test both server-side C# code and client-side JavaScript/TypeScript code. The course can be customized to omit JavaScript or replace JavaScript with TypeScript. Optionally, the course includes ASP.NET Blazor and the unit testing of Razor Components. The course focuses on TDD's principles, best practices, and application in ASP.NET and client-side JavaScript code.

**Prerequisites**

All students must have experience with the following:

* C# and ASP.NET technologies
* JavaScript or TypeScript

**Materials**

All TDD training attendees receive courseware and code covering the topics in the class.

**Software Needed on Each Student PC**

* Visual Studio 2022, Visual Studio Code, or JetBrains Rider
* Windows, macOS, or Linux
* .NET 8.0 SDK
* Node.js (latest LTS version)
* Ability to download NuGet and NPM packages
* LocalDB or SQL Server
* For classes delivered online, all participants need either dual monitors or a separate device logged into the online session so that they can do their work on one screen and watch the instructor on the other. A separate computer connected to a projector or large screen TV would be another way for students to see the instructor's screen simultaneously with working on their own.

**Objectives**

* Learn the principle and best practices of Test-Driven Development
* Apply TDD in ASP.NET and JavaScript/TypeScript code
* Explore the parts of unit testing and E2E testing
* Practice how to organize unit tests
* Understand what needs to be tested for different parts of the application
* Enable testing in isolation with mocks, stubs, and fakes
* Integrate testing into local development tools and CICD pipelines (Azure DevOps or GitHub)

**Outline**

* Introduction
  + What’s New in .NET 7
  + What is Test-Driven Development?
  + Benefits of TDD
  + Challenges of TDD
* Principles of TDD (inspired by “Uncle” Bob Martin)
  + Three Laws of TDD
  + Clean Tests
  + One Assert Per Test
  + Five Rules: FIRST
    - Fast
    - Independent
    - Repeatable
    - Self-Validating
    - Timely
  + Red, Green, Refactor Technique
* Kinds of Testing
  + Unit Tests
  + Integration Tests
  + E2E Testing
  + Automated vs. Manual Testing
  + Testing & DevOps
* Testing Parts
  + Tests
  + Test Suites
  + Assertions
  + Setup/Teardown
  + Mocks, Fakes, Stubs
  + Arrange, Act, Assert
  + Test Frameworks
  + Test Runners
  + Code Coverage
* Overview of .NET Core and Testing
  + Testing Frameworks
    - MSTest
    - NUnit
    - XUnit
  + Test Runners
    - Command-Line
    - Visual Studio
    - Visual Studio Code
  + Testing Libraries
    - Mocking with Moq
    - Fluent Assertions
    - BUnit (for classes using Blazor)
* xUnit
  + What is xUnit?
  + Testing Framework
  + Test Parallelism
  + Shared Test Context
  + Facts vs. Theory
  + Assertions
  + Integration with Visual Studio
  + Debugging Unit Tests in Visual Studio
  + Debugging Unit Tests in Visual Studio Code
* ASP.NET MVC Test-Driven Development (focus is on testing REST APIs)
  + What Should be Tested on an MVC application?
  + Integrate Test Projects into a Solution
  + Testing Controllers
  + Testing APIs
  + Integration Testing of APIs
* ASP.NET Blazor Test-Driven Development
  + What Should be Tested on a Razor Component?
  + What is bUnit?
  + Using bUnit with xUnit
  + Setup and define components under tests in C# or Razor syntax
  + Verify outcome using semantic HTML comparer
  + Interact with and inspect components
  + Trigger event handlers
  + Provide cascading values
  + Inject services
  + Mock IJsRuntime
  + Perform snapshot testing
* Mocking Databases (choose 1)
  + Entity Framework
  + Dapper
* JavaScript Test-Driven Development
  + Is JavaScript Unit Testing a Thing?
  + Benefits of Unit Testing JavaScript
  + Challenges of Unit Testing JavaScript
* JavaScript Testing Frameworks Overview
  + Jest
  + Jasmine
  + Mocha
* JavaScript Test Runners Overview
  + Jest
  + Karma
* JavaScript E2E Testing Overview
  + What is End-To-End Testing?
  + Selenium WebDriver
  + Cypress
* JavaScript TDD with BDD
  + What is Behavior-Driven Development?
  + Coding Unit Tests around Behavior vs Functions
  + Given-When-Then
* JavaScript Unit Testing (will be covered with Jest or Jasmine)
  + Test Suites
  + Tests
  + Assertions
  + Mocks and Spies
* JavaScript E2E Testing (will be covered with Selenium)
  + Finding Elements in the DOM Tree
  + Page Objects
* Testing JavaScript Code Frameworks
  + Plain Vanilla JavaScript (can use Jest or Jasmine)
    - Configure Unit Testing for a JavaScript/TypeScript project
    - Organize Code for Efficient Testing
    - Mocking the DOM
    - Testing DOM Manipulation Code
    - Testing AJAX Code
    - Running Tests
    - Debugging Tests
* Code Coverage
  + What is Code Coverage?
  + What can be understood from Code Coverage?
  + What are the limitations of Code Coverage?
  + Generate Code Coverage reports for ASP.NET C# code
  + Generate Code Coverage reports for JavaScript code
  + Integrate code coverage into CICD systems (Azure DevOps or GitHub Actions)
* Conclusion