.NET Core Frameworks

Course Summary

Description

Microsoft .NET is a programming technology that greatly simplifies application development, both for traditional, proprietary applications and for web-based applications and services. The original .NET was a complete restructuring of Microsoft's whole system infrastructure and represented a major learning challenge for programmers developing applications on Microsoft platforms. Part of .NET included a major new object-oriented programming language, C#. But learning the new programming language is only part of the challenge. The much greater challenge is learning the .NET Framework and its many capabilities.

A major restructuring of the .NET platform, .NET Core is cross-platform, open source, and modular designed for creating modern web applications and services, libraries and console applications. It is available on Windows, OS X and Linux. .NET now is a family of frameworks, including both the classical .NET available on Windows and the new .NET Core package-based frameworks that are cross-platform.

This two-day course is designed to provide a sound introduction to.NET Core for programmers who already know the C# language. It is current to .NET Core 1.0.1 and Visual Studio 2015. The course focuses on core portions of the .NET Framework that are common across many application areas. It starts with an introduction to the architecture and key concepts of .NET. The course then discusses class libraries, packages, metapackages and frameworks. The following chapters discuss important topics in the .NET programming model, including delegates and events, I/O and serialization, memory management, processes and threads. The course concludes with a chapter on threading, which includes an introduction to the Task Parallel Library (TPL).

The course is hands-on, with many programming examples. The goal is to equip you with the foundations of this important new technology from Microsoft. The student will receive a comprehensive set of materials, including course notes and all the programming examples.

Objectives

By the end of this course, students will be able to:

- Gain a thorough understanding of the philosophy and architecture of .NET Core
- Understand packages, metapackages and frameworks
- Acquire a working knowledge of the .NET programming model
- Implement multi-threading effectively in .NET applications

Topics

- .NET Fundamentals
- Class Libraries
- Packages and Frameworks
- I/O and Serialization
- Delegates and Events
- .NET Programming Model
- .NET Threading

Audience

This course is designed for programmers who already know the C# language

Prerequisites

The student should be an experienced application developer or architect with a working knowledge of C#.

Duration

Two days

.NET Core Frameworks

Course Outline

I. .NET Fundamentals
- A. What is Microsoft .NET?
- B. Common Language Runtime
- C. Framework Class Library
- D. Language Interoperability
- E. Managed Code
- F. .NET Core and Cross-Platform Development

II. Class Libraries
- A. Components in .NET
- B. Class Libraries Using Visual Studio
- C. Using References

III. Packages and Frameworks
- A. NuGet Packages and Gallery
- B. Metapackages and Frameworks
- C. Packages in .NET Core
- D. Porting from .NET 4.6 to .NET Core
- E. Visual Studio Package Manager
- F. Installing Packages
- G. Creating Packages

IV. I/O and Serialization
- A. Directories
- B. Files and Streams
- C. XML Serialization

V. Delegates and Events
- A. Delegates
- B. Random Number Generation
- C. Events

VI. .NET Programming Model
- A. Garbage Collection
- B. Finalize and Dispose
- C. Processes
- D. Command-Line Arguments
- E. Threads

VII. .NET Threading
- A. Threading Fundamentals
- B. ThreadPool
- C. Foreground and Background Threads
- D. Synchronization
- E. Task Parallel Library