+33 (0)1 48 10 09 46
formation@ambient-it.net 140 Avenue Jean Lolive 93500 Pantin - France

Updated on 08/21/2026

Sign Up

.NET Application Optimization with C# Training

3 days (21 hours)

Overview

Optimizing a .NET application with C# involves mastering CPU performance, memory management, data access, and scalability to deliver a reliable user experience.

Our .NET Application Optimization with C# training will enable you to identify the root causes of performance issues and implement a structured approach to measurement, analysis, and improvement. You'll learn to distinguish between problems related to code, I/O, data queries, concurrency, and system resources.

You'll use profiling and diagnostic tools from the .NET ecosystem to analyze CPU usage, memory allocations, costly calls, deadlocks, and response times. The course also covers garbage collection mechanisms, asynchronous tasks, and best practices for managing collections and strings.

You will learn how to optimize database access with Entity Framework Core, reduce unnecessary data transfers, apply appropriate caching strategies, and improve the performance of ASP.NET Core APIs. Configuration choices, compression, rate limiting, and request handling will be examined in light of production constraints.

By the end of the course, you will be able to define relevant metrics, reproduce a performance issue, propose measurable fixes, and validate their effects through load testing. You will have a pragmatic approach to ensuring the reliability of .NET applications in a real-world environment.

Like all our training courses, this one will introduce you to **the latest stable version** of the technology and its new features.

Objectives

- Implement a process for measuring and analyzing the performance of a .NET application
- Identify bottlenecks related to the CPU, memory, I/O, and concurrency
- Use profiling and diagnostic tools from the .NET ecosystem
- Optimize C# code, data access, and asynchronous processing
- Improve the performance and scalability of ASP.NET Core applications
- Validate optimizations using metrics, load testing, and benchmarking

Target Audience

- .NET Developer
- C# developers
- ASP.NET Core Developer
- Software Architect
- .NET Tech Lead

Requirements

- Proficiency in C# and object-oriented programming
- Experience developing applications with .NET
- Knowledge of async, await, and LINQ
- Basic understanding of data access using SQL or Entity Framework Core

Technical Requirements

- Computer running Windows 10, Windows 11, macOS, or Linux
- .NET SDK 8.0 or later installed
- Visual Studio 2022 or Visual Studio Code with C# extensions
- Computer with 16 GB of RAM (recommended) and a multicore processor
- Stable internet connection to install tools and NuGet packages

Our Training Program: Optimizing .NET Applications .NET with C#

[Day 1 - Morning]

Measure Before You Optimize

- Challenges related to performance, latency, throughput, and resource consumption

- The optimization process: establishing a baseline, formulating a hypothesis, measuring, and validating
- Key metrics: response time, CPU, memory, allocations, exceptions, and thread activity
- The pitfalls of non-reproducible measurements and the influence of development and production environments
- Introduction to Visual Studio Profiler, dotnet-counters, and dotnet-trace
- Hands-on workshop: creating a baseline profile for a .NET application with controlled performance bottlenecks

[Day 1 - Afternoon]

Optimizing C# code and CPU usage

- Identifying hotspots based on calls, execution stacks, and processor samples
- Reducing the overhead of loops, conditions, conversions, concatenations, and repetitive operations
- Choose the right collections and make effective use of Span<T>, Memory<T>, and data structures
- Master the trade-offs between readability, allocations, data copies, and micro-optimizations
- Avoid deadlocks related to Task.Result, Wait, and costly synchronous calls
- Hands-on workshop: Analyze a CPU hotspot and refactor C# code to reduce its execution time

[Day 2 - Morning]

Mastering memory and the garbage collector

- Understand the managed heap, generations, the garbage collector, and large objects
- Identify excessive allocations, memory retention, and symptoms of memory pressure
- Compare Workstation GC and Server GC modes based on the type of application workload
- Minimize allocations through object reuse, pools, buffers, and appropriate APIs
- Diagnose logical leaks, persistent references, and unnecessarily retained objects
- Hands-on Workshop: Investigating Excessive Memory Usage and Reducing Allocations in a .NET Application

[Day 2 - Afternoon]

Accelerate I/O and data access

- Identify the cost of I/O, network access, and calls to remote services
- Structure asynchronous processing with async, await, cancellation, and timeout handling
- Optimize Entity Framework Core queries: projection, filtering, pagination, and trackless queries
- Reduce round trips, prevent excessive loading, and control the number of queries generated
- Implement memory cache and distributed cache strategies based on the expected consistency

- Hands-on workshop: Optimize an endpoint accessing a database and compare metrics before and after the fix

[Day 3 - Morning]

Improve the performance of ASP.NET Core APIs

- Analyze the processing flow of an ASP.NET Core request and the order of middleware
- Optimize serialization, response size, pagination, and handling of large volumes
- Distinguish between response caching, output caching, and application caching
- Configure response compression and evaluate its performance and security trade-offs
- Protect resources with rate limiting, timeout settings, and load balancing
- Hands-on workshop: Improving the performance of an ASP.NET Core API with caching, pagination, and compression

[Day 3 - Afternoon]

Diagnose and validate under realistic conditions

- Build a load testing scenario that reflects real-world usage and volumes
- Interpret metrics for throughput, latency, CPU utilization, queues, and application errors
- Collect and analyze traces with dotnet-trace and counters with dotnet-counters
- Set up useful instrumentation: business metrics, structured logs, traces, and alerts
- Develop a prioritized optimization plan and ensure regression testing through automated tests
- Hands-on workshop: Perform a comprehensive under-load diagnosis, apply fixes, and present the measured performance gains

Target Audience

This training is intended for both individuals and companies—large or small—that wish to train their teams in a new advanced IT technology or to acquire specific business knowledge or modern methods.

Pre-training Assessment

The pre-training assessment complies with Qualiopi quality standards. Upon final registration, the learner receives a self-assessment questionnaire that allows us to evaluate their estimated proficiency level across various types of technologies, as well as their expectations and personal goals for the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could pose challenges for monitoring and ensuring the smooth operation of the training session.

Teaching Methods

Hands-On Training: 60% Practical, 40% Theoretical. Training materials distributed in

digital format to all participants.

Organization

The course alternates between theoretical input from the instructor—supported by examples and reflection sessions—and group work.

Assessment

At the end of the session, a multiple-choice quiz is administered to verify that participants have successfully acquired the skills.

Certification

A certificate will be issued to each trainee who has completed the entire training program.