

Fundamentals of C# / .NET

Level: foundation

Length: 35 – 40 hours

Course Objective: a solid introduction of the C# programming language and its ecosystem .NET platform

What You Will Learn

- How the programs are organized, development cycle of writing software, using an integrated development environment (IDE)
- How the code is constructed starting with requirements, how we discover and implement the entities by using the language elements
- Code structure: data structures, control structures, fundamental, primitives types
- C# support for object oriented programming (OOP): data abstraction, relations between types, polymorphism, generic programming
- Basics regarding .NET framework
- Exercise OOP paradigm
- Enhance soft skills: communication, team work, presentation

Who Can Attend: programmers who want to learn the fundamentals of C# & .NET

Prerequisites

- Fundamentals of programming (data structures, flow control, interaction with the host platform)
- It is helpful the knowledge of another/any object oriented language like Java or C++, the basics of object oriented programming

Required Facilities: VGA projector, white board, workstation, a version of Microsoft Visual Studio

Related Courses: Object oriented analysis and design, Design Patterns, Advanced C# / .NET Topics, specialized courses related to .NET – Windows Presentation Foundation (WPF), Active Server Pages MVC (ASP .NET MVC)

Description

Dive into .NET programming with C#, the cornerstone language for building robust solutions on the .NET platform. This highly interactive course emphasizes hands-on coding, guiding you through real-world problem-solving to master C# and its ecosystem.

Through practical exercises, you'll analyze requirements, design entities, and write code to implement solutions. This approach serves as a foundation for exploring C# language elements, their semantics, and their application in creating functional components.

While the primary focus is C# programming, the course also addresses key .NET concepts due to the deep integration between the language and its framework.

Topics covered include a range of subjects explored through solving practical, scenario-based problems, as outlined in the course syllabus.

Note: the course is personalized on the attendees' profile, their expertise and experience

Contents

1. .NET Architecture – relation between C# and .NET, CLR – Common Language Runtime, Assemblies, framework .NET classes, namespaces, use of C# in enterprise applications
2. Introduction to C#, basics – variables, predefined types, program control, enumerations, arrays, use of namespaces, compilation, console input/output, preprocessor directives
3. Types and objects – classes and structs, class members, anonymous types, partial classes, static classes, Object class, extending the classes – extension methods
4. Inheritance – types of inheritance, implementation inheritance, virtual methods, methods hiding, abstract classes and abstract methods, sealed classes and methods, constructors, modifiers, interfaces
5. Arrays – simple arrays, multidimensional arrays, jagged arrays, Array class, array and collection interfaces, enumerations, IEnumerator, foreach, yield
6. Operators and casts – checked, unchecked, is, as, typeof, nullable types, type conversions, boxing and unboxing, comparing objects for equality, operator overloading, user defined casts
7. Delegates and events – declaring, using, multicast delegates, anonymous methods, Lambda expressions, use of events
8. Strings – String, StringBuilder, formatting strings
9. Generics – performance, type safety, creating, default values, constraints, inheritance, static members, generic interfaces, generic methods, generic delegates
10. Collections – collection interfaces and types, lists, queues, stacks, sorted lists, dictionaries, hashset, bit arrays
11. Exception handling – exception classes, catching exceptions, user defined exception classes
12. Manipulating files – files and directories, operations
13. (optional, depending on the attendees' level) Basics of programming with threads