

Programming with C#

Introduction

The goal of this course is to provide students with the knowledge and skills they need to develop C# applications for the Microsoft .NET Platform. The course focuses on C# program structure, language syntax, and implementation details.

C# was created to be the programming language best suited for writing enterprise applications for .NET. C# combines the high productivity of Microsoft Visual Basic with the raw power of C++. It is a simple, object-oriented, and type-safe programming language that is based on the C and C++ family of languages.

Pre-Requisites

Before attending this course, students must have:

- Experience with programming in C, C++, Visual Basic, Java, or another programming language.
- Familiarity with the Microsoft .NET strategy as described on the Microsoft .NET Web site: <http://www.microsoft.com/net/>
- Familiarity with the .NET Framework as described on the MSDN Magazine Web site: <http://msdn.microsoft.com/msdnmag/issues/0900/Framework/Framework.asp> or <http://msdn.microsoft.com/msdnmag/issues/1000/Framework2/Framework2.asp>

Outcomes

At the completion of this course, participants will be able to:

- List the major elements of the .NET Framework and explain how C# fits into the .NET Platform.
- Analyze the basic structure of a C# application and be able to document, debug, compile, and run a simple application.
- Create, name, and assign values to variables.
- Use common statements to implement flow control, looping, and exception handling.
- Create methods (functions and subroutines) that can return values and take parameters.
- Create, initialize, and use arrays.
- Explain the basic concepts and terminology of object-oriented programming.
- Use common objects and reference types.
- Create, initialize, and destroy objects in a C# application.
- Build new C# classes from existing classes.
- Create self-contained classes and frameworks in a C# application.
- Define operators, use delegates, and add event specifications.
- Implement properties and indexers.
- Use predefined and custom attributes.

Course Details

Course Code: MS 2124

Duration: 5 days

Starting time: 9.00 am

Finishing time: 4.30 pm

Lunch and refreshments are provided.

Booking guidelines

Contact our Learning Consultants on 1300 86 87246 and we will assist you with your booking.



Learning Solutions

(1300 86 87246
1300 TO TRAIN

Course Outline

Ø Module 1 Overview of the Microsoft .NET Platform

The following topics are covered in this module:

- Introduction to the .NET Platform
- Overview of the .NET Framework
- Benefits of the .NET Framework
- The .NET Framework Components
- Languages in the .NET Framework

After completing this module, you will be:

- Describing the .NET Platform.
- Listing the main elements of the .NET Platform.
- Explaining the language support in the .NET Framework.
- Describing the .NET Framework and its components.

Ø Module 2: Overview of C#

The following topics are covered in this module:

- Structure of a C# Program.
- Basic Input/Output Operations.
- Recommended Practices.
- Compiling, Running, and Debugging.

After completing this module, you will be able to:

- Explain the structure of a simple C# program.
- Use the Console class of the System namespace to perform basic input/output operations.
- Handle exceptions in a C# program.
- Generate Extensible Markup Language (XML) documentation for a C# application.
- Compile and execute a C# program.

Ø Module 3: Using Value-Type Variables

The following topics are covered in this module:

- Common Type System
- Naming Variables
- Using Built-In Data Types
- Creating User-Defined Data Types
- Converting Data Types

After completing this module, you will be able to:

- Describe the types of variables that you can use in C# applications.
- Name your variables according to standard C# naming conventions.
- Declare variables by using built-in data types.
- Assign values to variables.
- Convert existing variables from one data type to another.
- Create and use your own data types.

Ø Module 4: Statements and Exceptions

The following topics are covered in this module:

- Introduction to Statements
- Using Selection Statements
- Using Iteration Statements
- Using Jump Statements
- Handling Basic Exceptions
- Raising Exceptions

After completing this module, you will be able to:

- Handling and raising exceptions.
- Using jump statements.
- Using selection statements.
- Using iteration statements.
- Describing the different types of control statements

Ø Module 5: Methods and Parameters

- The following topics are covered in this module:
- Using Methods
- Using Parameters
- Using Overloaded Methods

After completing this module, you will be able to:

- Create static methods that accept parameters and return values.

- Pass parameters to methods in different ways.
- Declare and use overloaded methods.

Ø Module 6: Arrays

The following topics are covered in this module:

- Overview of Arrays
- Creating Arrays
- Using Arrays

After completing this module, you will be able to:

- Create, initialize, and use arrays of varying rank.
- Use command-line arguments in a C# program.
- Describe the relationship between an array variable and an array instance.
- Use arrays as parameters for methods.
- Return arrays from methods.

Ø Module 7: Essentials of Object-Oriented Programming

The following topics are covered in this module:

- Classes and Objects
- Using Encapsulation
- C# and Object Orientation
- Defining Object-Oriented Systems

After completing this module, you will be able to:

- Define the terms object and class in the context of object-oriented programming.
- Describe the three core aspects of an object: identity, state, and behaviour.
- Describe abstraction and how it helps you to create reusable classes that are easy to maintain.
- Use encapsulation to combine methods and data in a single class and enforce abstraction.
- Explain the concepts of inheritance and polymorphism.
- Create and use classes in C#.

Course Outline

Ø Module 8: Using Reference-Type Variables The following

The following topics are covered in this module:

- Using Reference-Type Variables
- Using Common Reference Types
- The Object Hierarchy
- Namespaces in the .NET Framework
- Data Conversions

After completing this module, you will be able to:

- Describe the key differences between reference types and value types.
- Use common reference types such as string.
- Explain how the object type works and becoming familiar with the methods it supplies.
- Describe common namespaces in the .NET Framework.
- Determine whether different types and objects are compatible.
- Explicitly and implicitly convert data types between reference types.
- Perform boxing and unboxing conversions between reference and value data.

Ø Module 9: Creating and Destroying Objects

The following topics are covered in this module:

- Using Constructors
- Initializing Data
- Objects and Memory
- Resource Managements

After completing this module, you will be able to:

- Use constructors to initialize objects.
- Create overloaded constructors that can accept varying parameters.
- Describe the lifetime of an object and what happens when it is destroyed.
- Create destructors and using Finalize methods.

Ø Module 10: Inheritance in C#

The following topics are covered in this module:

- Deriving Classes
- Implementing Methods
- Using Sealed Classes
- Using Interfaces
- Using Abstract Classes

After completing this module, you will be able to:

- Derive a new class from a base class and calling members and constructors of the base class from the derived class.
- Declare methods as virtual and override or hiding them as required.
- Seal a class so that it cannot be derived from.
- Implement interfaces by using both the implicit and explicit methods.
- Describe the use of abstract classes and their implementation of interfaces

Ø Module 11: Aggregation, Namespaces, and Advanced Scope

The following topics are covered in this module:

- Using Internal Classes, Methods, and Data
- Using Aggregation
- Using Namespaces
- Using Modules and Assemblies

After completing this module, you will be able to:

- Use internal access to allow classes to have privileged access to each other.
- Use aggregation to implement powerful patterns such as Factories.
- Use namespaces to organize classes.
- Create simple modules and assemblies.

Ø Module 12: Operators and Events

The following topics are covered in this module:

- Introduction to Operators
- Operator Overloading
- Creating and Using Delegates
- Defining and Using Events

After completing this module, you will be able to:

- Define operators to make a class or struct easier to use.
- Use delegates to decouple a method call from a method implementation.
- Add event specifications to a class to allow subscribing classes to be notified of changes in object state.

Ø Module 13: Properties and Indexers

The following topics are covered in this module:

- Using Properties
- Using Indexers

After completing this module, you will be able to:

- Create properties to encapsulate data within a class.
- Define indexers to gain access to classes by using array-like notation.

Ø Module 14: Attributes

The following topics are covered in this module:

- Overview of Attributes
- Defining Custom Attributes
- Retrieving Attribute Values

After completing this module, you will be able to:

- Use common predefined attributes.
- Create simple custom attributes.
- Query attribute information at run time.