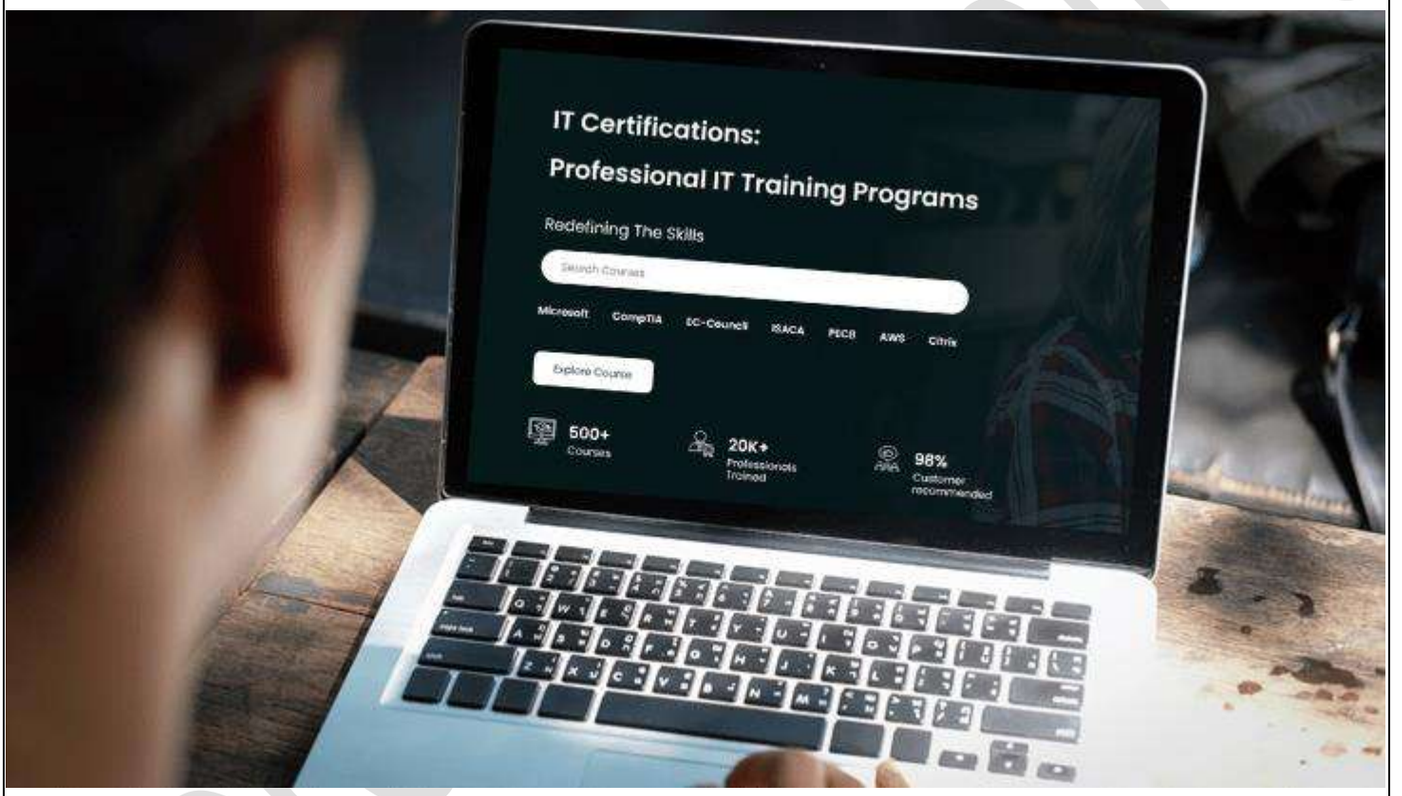
MICROTEK LEARNING | Redefining The Skills



55339: PROGRAMMING IN C# TRAINING

Duration: 5 Days

Course Description

This five-day training course teaches developers the programming skills that are required for developers to create Windows applications using the C# language.

During their five days in the classroom, students review the basics of C# program structure, language syntax, and implementation details, and then consolidate their knowledge throughout the week as they build an application that incorporates several features of the .NET 6.0.

Training Exclusives

- Live instructor-led interactive sessions with Microsoft Certified Trainers (MCT).
- Access to Microsoft Official Courseware (MOC).
- Real-time Virtual Lab Environment.
- Experience 24*7 Learner Support.
- Self-paced learning and flexible schedules.

Who should attend this course?

- This course is intended for experienced developers who already have programming experience in C, C++, JavaScript, Objective-C, Microsoft Visual Basic, or Java and understand the concepts of object-oriented programming.
- This course is not designed for students who are new to programming; it is targeted at professional developers with at least one month of experience programming in an object-oriented environment.

What you will learn

- Describe the core syntax and features of C#.
- Create methods, handle exceptions, and describe the monitoring requirements of large-scale applications.
- Implement the basic structure and essential elements of a typical desktop application.
- Create classes, define and implement interfaces, and create and use generic collections.
- Use inheritance to create a class hierarchy and to extend a .NET class.
- Read and write data by using file input/output and streams, and serialize and deserialize data in different formats.
- Create and use an entity data model for accessing a database and use LINQ to query data.
- Access and query remote data by using the types in the System.Net namespace and WCF Data Services.
- Build a graphical user interface by using XAML.
- Improve the throughput and response time of applications by using tasks and asynchronous operations.
- Integrate unmanaged libraries and dynamic components into a C# application.
- Examine the metadata of types by using reflection, create and use custom attributes, generate code at runtime, and manage assembly versions.
- Encrypt and decrypt data by using symmetric and asymmetric encryption.

Prerequisites

- Developers attending this course should already have gained some limited experience using C# to complete basic programming tasks. More specifically, students should have hands-on experience using C# that demonstrates their understanding of the following:
- How to name, declare, initialize and assign values to variables within an application.
- How to use: arithmetic operators to perform arithmetic calculations involving one or more variables; relational operators to test the relationship between two variables or expressions; logical operators to combine expressions that contain relational operators.
- How to create the code syntax for simple programming statements using C# language keywords and recognize syntax errors using the Visual Studio IDE.
- How to create a simple branching structure using an IF statement.
- How to create a simple looping structure using a For statement to iterate through a data array.
- How to use the Visual Studio IDE to locate simple logic errors.
- How to create a Function that accepts arguments (parameters and returns a value of a specified type).
- How to design and build a simple user interface using standard controls from the Visual Studio toolbox.
- How to connect to a SQL Server database and the basics of how to retrieve and store data.
- How to sort data in a loop.
- How to recognize the classes and methods used in a program.

Curriculum

Module 1: Review of C# Syntax

Microsoft .NET 6.0 provides a comprehensive development platform that you can use to build, deploy, and manage applications and services. By using .NET, you can create visually compelling applications, enable seamless communication across technology boundaries, and provide support for a wide range of business processes.

In this module, you will learn about some of the core features provided by.NET and Microsoft Visual Studio. You will also learn about some of the core C# constructs that enable you to start developing .NET applications.

Lessons

- Overview of Writing Application by Using C#
- Data Types, Operators, and Expressions
- C# Programming Language Constructs

Lab 1: Implementing Edit Functionality for the Students List

- Implementing Insert Functionality for the Students List
- Implementing Delete Functionality for the Students List
- Displaying a Student's Age

After completing this module, students will be able to:

- Describe the architecture of .NET Framework applications and the features that Visual Studio 2022 and C# provide.
- Use basic C# data types, operators, and expressions.
- Use standard C# constructs.

Module 2: Creating Methods, Handling Exceptions, and Monitoring Applications

Applications often consist of logical units of functionality that perform specific functions, such as providing access to data or triggering some logical processing. C# is an object-orientated language and uses the concept of methods to encapsulate logical units of functionality. Although a good practice is to have methods that do just one thing, they can be as simple or as complex as you like. It is also important to consider what happens to the state of your application when an exception occurs in a method.

In this module, you will learn how to create and use methods and how to handle exceptions. You will also learn how to use logging and tracing to record the details of any exceptions that occur.

Lessons

- Creating and Invoking Methods
- Creating Overloaded Methods and Using Optional and Output Parameters
- Handling Exceptions
- Monitoring Applications

Lab 1: Extending the Class Enrolment Application Functionality

- Refactoring the Enrolment Code
- Validating Student Information
- Saving Changes to the Class List

After completing this module, students will be able to:

- Create and invoke methods.
- Create overloaded methods and use optional parameters.
- Handle exceptions.
- Monitor applications by using logging, tracing, and profiling

Module 3: Basic types and constructs of C#

To create effective applications you must first learn some fundamental C# constructs. You need to know how to create simple structures to represent the data items you are working with. You need to know how to organize these structures into collections, so that you can add items, retrieve items, and iterate over your items. Finally, you need to know how to subscribe to events so that you can respond to the actions of your users.

In this module, you will learn how to create and use structs and enums, organize data into collections, and create and subscribe to events.

Lessons

- Implementing Structs and Enums
- Organizing Data into Collections
- Handling Events

Lab 1: Writing the Code for the Grades Prototype Application

- Adding Navigation Logic to the Grades Prototype Application
- Creating Data Types to Store User and Grade Information
- Displaying User and Grade Information

After completing this module, students will be able to:

- Create and use structs and enums.
- Use collection classes to organize data.
- Create and subscribe to events.

Module 4: Creating Classes and Implementing Type-safe Collections

In this module, you will learn how to use interfaces and classes to define and create your own custom, reusable types. You will also learn how to create and use enumerable type-safe collections of any type.

Lessons

- Creating Classes
- Defining and Implementing Interfaces
- Implementing Type-Safe Collections

Lab 1: Adding Data Validation and Type-Safety to the Application

- Implementing the Teacher, Student, and Grade Structs as Classes
- Adding Data Validation to the Grade Class
- Displaying Students in Name Order
- Enabling Teachers to Modify Class and Grade Data

After completing this module, students will be able to:

- Create and instantiate classes.
- Create and instantiate interfaces.
- Use generics to create type-safe collections.

Module 5: Creating a Class Hierarchy by Using Inheritance

In this module, you will learn how to use inheritance to create class hierarchies and to extend .NET types.

Lessons

- Creating Class Hierarchies
- Extending .NET Classes

Lab 1: Refactoring Common Functionality into the User Class

- Refactoring Common Functionality into the User Class
- Implementing Password Complexity by Using an Abstract Method
- Creating the ClassFullException Custom Exception

After completing this module, students will be able to:

- Create base classes and derived classes by using inheritance.
- Create classes that inherit from .NET classes.

Module 6: Reading and Writing Local Data

In this module, you will learn how to read and write data by using transactional filesystem I/O operations, how to serialize and deserialize data to the filesystem, and how to read and write data to the filesystem by using streams.

Lessons

- Reading and Writing Files
- Serializing and Deserializing Data
- Performing I/O by Using Streams

Lab 1: Generating the Grades Report

- Serializing Data for the Grades Report as XML
- Previewing the Grades Report
- Persisting the Serialized Grade Data to a File

After completing this module, students will be able to:

- Read and write data to and from the file system by using file I/O.
- Convert data into a format that can be written to or read from a file or other data source.
- Use streams to send and receive data to or from a file or data source.

Module 7: Accessing a Database

In this module, you will learn how to use Entity Framework and how to query many types of data by using Language-Integrated Query (LINQ).

Lessons

- Creating and Using Entity Data Models
- Querying Data by Using LINQ

Lab 1: Retrieving and Modifying Grade Data

- Creating an Entity Data Model from The School of Fine Arts Database
- Updating Student and Grade Data by Using the Entity Framework
- Extending the Entity Data Model to Validate Data

After completing this module, students will be able to:

- Create, use, and customize an EDM.
- Query data by using LINQ.

Module 8: Accessing Remote Data

In this module, you will learn how to use the request and response classes in the System.Net namespace to directly manipulate remote data sources. You will also learn how to use Windows Communication Foundation (WCF) Data Services to expose and consume data over the web.

Lessons

- Accessing Data Across the Web
- Accessing Data by Using OData Connected Services

Lab 1: Retrieving and Modifying Grade Data Remotely

- Creating a WCF Data Service for the SchoolGrades Database
- Integrating the Data Service into the Application
- Retrieving Student Photographs Over the Web (If Time Permits)

After completing this module, students will be able to:

- Send data to and receive data from web services and other remote data sources.
- Access data by using WF Data Services.

Module 9: Designing the User Interface for a Graphical Application

In this module, you will learn how to use Extensible Application Markup Language (XAML) and Windows Presentation Foundation (WPF) to create engaging UIs.

Lessons

- Using XAML to Design a User Interface
- Binding Controls to Data

Lab 1: Customizing Student Photographs and Styling the Application

- Customizing the Appearance of Student Photographs
- Styling the Logon View
- Animating the StudentPhoto Control (If Time Permits)

After completing this module, students will be able to:

- Use XAML to design a UI.
- Bind a XAML control to data.
- Apply styles to a XAML UI.

Module 10: Improving Application Performance and Responsiveness

In this module, you will learn how to improve the performance of your applications by distributing your operations across multiple threads.

Lessons

- Implementing Multitasking
- Performing Operations Asynchronously
- Synchronizing Concurrent Access to Data

Lab 1: Improving the Responsiveness and Performance of the Application

- Ensuring That the UI Remains Responsive When Retrieving Teacher Data
- Providing Visual Feedback During Long-Running Operations

After completing this module, students will be able to:

- Use the Task Parallel Library to implement multitasking.
- Perform long-running operations without blocking threads.
- Control how multiple threads can access resources concurrently.

Module 11: Integrating with Unmanaged Code

In this module, you will learn how to interoperate unmanaged code in your applications and how to ensure that your code releases any unmanaged resources.

Lessons

- Creating and Using Dynamic Objects
- Managing the Lifetime of Objects and Controlling Unmanaged Resources

Lab 1: Upgrading the Grades Report

- Generating the Grades Report by Using Word
- Controlling the Lifetime of Word Objects by Implementing the Dispose Pattern

After completing this module, students will be able to:

- Integrate unmanaged code into a C# application by using the Dynamic Language Runtime (DLR).
- Control the lifetime of unmanaged resources and ensure that your application releases resources.

Module 12: Creating Reusable Types and Assemblies

In this module, you will learn how to consume existing assemblies by using reflection, and how to add additional metadata to types and type members by using attributes. You will also learn how to generate code at run time by using the Code Document Object Model (CodeDOM) and how to ensure that your assemblies are signed and versioned, and available to other applications, by using the global assembly cache (GAC).

Lessons

- Examining Object Metadata
- Creating and Using Custom Attributes

- Generating Managed Code
- Versioning, Signing, and Deploying Assemblies

Lab 1: Specifying the Data to Include in the Grades Report

- Creating and Applying the IncludeInReport attribute
- Updating the Report
- Storing the Grades.Utilities Assembly Centrally (If Time Permits)

After completing this module, students will be able to:

- Use reflection to inspect and execute assemblies.
- Create and consume custom attributes.
- Generate managed code at run time by using CodeDOM.
- Version, sign, and deploy your assemblies to the GAC.

Module 13: Securing Data

In this module, you will learn how to implement symmetric and asymmetric encryption and how to use hashes to generate mathematical representations of your data. You will also learn how to create and manage X509 certificates and how to use them in the asymmetric encryption process.

Lessons

- Implementing Symmetric Encryption
- Implementing Asymmetric Encryption

Lab 1: Encrypting and Decrypting the Grades Report

- Encrypting the Grades Report
- Decrypting the Grades Report

After completing this module, students will be able to:

- Encrypt data by using symmetric encryption.
- Encrypt data by using asymmetric encryption.