

MVS3-2 - MOC 20483 - PROGRAMMING IN C#

Categoria: **Visual Studio**

INFORMAZIONI SUL CORSO



Durata:
5 Giorni



Categoria:
Visual Studio



Qualifica Istruttore:
Microsoft Certified
Trainer



Dedicato a:
Sviluppatore



Produttore:
Microsoft

OBIETTIVI

After completing this course, students will be able to:

- Describe the core syntax and features of Visual 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 Framework 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 Visual 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.

PREREQUISITI

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.

CONTENUTI

Module 1: Review of Visual C# Syntax

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

Lab: 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

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

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

Lab: Extending the Class Enrolment Application Functionality

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

Module 3: Basic types and constructs of Visual C#

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

Lab: 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

Module 4: Creating Classes and Implementing Type-Safe Collections

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

Lab: 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

Module 5: Creating a Class Hierarchy by Using Inheritance

- Creating Class Hierarchies
- Extending .NET Framework Classes

Lab: 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

Module 6: Reading and Writing Local Data

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

Lab: Generating the Grades Report

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

Module 7: Accessing a Database

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

Lab: 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

Module 8: Accessing Remote Data

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

Lab: 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)

Module 9: Designing the User Interface for a Graphical Application

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

Lab: Customizing Student Photographs and Styling the Application

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

Module 10: Improving Application Performance and Responsiveness

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

Lab: 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

Module 11: Integrating with Unmanaged Code

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

Lab: Upgrading the Grades Report

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

Module 12: Creating Reusable Types and Assemblies

- Examining Object Metadata
- Creating and Using Custom Attributes
- Generating Managed Code
- Versioning, Signing, and Deploying Assemblies

Lab: 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)

Module 13: Encrypting and Decrypting Data

- Implementing Symmetric Encryption
- Implementing Asymmetric Encryption

Lab: Encrypting and Decrypting the Grades Report

- Encrypting the Grades Report
- Decrypting the Grades Report

INFO

Materiale didattico: Materiale didattico in formato digitale

Costo materiale didattico: incluso nel prezzo del corso a Calendario

Natura del corso: Operativo (previsti lab su PC)