WIZARDSGROUP

Microsoft Partner
Gold Learning
Gold Business Intelligence
Gold Portals and Collaboration
Gold Enterprise Resource Planning

Course 10550A: Programming in Visual Basic with Microsoft® Visual Studio® 2010

Five Days, Instructor-Led

About this course

This course teaches you Visual Basic language syntax, program structure, and implementation by using Microsoft® Visual Studio® 2010 and the Microsoft .NET Framework 4.

This course provides a solid foundation in Visual Basic to the level necessary to enable students to attend other courses in the Technical Specialist tracks.

Audience profile

This course is intended for experienced developers who already have programming experience in Visual Basic, C, C++, C#, or Java, and understand the concepts of Object Oriented Programming. These developers will be likely to develop enterprise business solutions.

These professional developers will be attending the course so that they can quickly ramp up on Visual Basic Programming in the .NET Framework. The course focuses on Visual Basic program structure, language syntax, and implementation details with the .NET Framework 4.0. This course also focuses on new enhancement in the Visual Basic 2010 language using Visual Studio 2010.

At course completion

After completing this course, students will be able to:
- Explore ASP.NET Web applications in Microsoft Visual Studio 2010.
- Create Web applications by using Microsoft Visual Studio 2010 and Microsoft .NET–based languages.
- Create a Microsoft ASP.NET Web Form.
- Add functionality to a Microsoft ASP.NET Web Form.
- Implement master pages and user controls.
- Validate user input.
- Debug Microsoft ASP.NET Web applications.
- Manage data in an ASP.NET 3.5 Web application.
- Manage data access tasks by using LINQ.
- Manage data by using ASP.NET Dynamic Data.
- Create a Microsoft ASP.NET AJAX application.
- Consume and Windows Communication Foundation (WCF) services.
- Manage state in Web applications.
- Configure and deploy a Microsoft ASP.NET Web application.
- Secure a Microsoft ASP.NET Web application.
- Implement new technologies supported by Visual Studio 2010 for Web development.

Pre-requisite

This course requires that you meet the following prerequisites:
- This course is targeted at developers who already have Visual Basic knowledge.
- This course is not for new developers; at least 12 months experience working with an Object Oriented language is expected.
- Creating classes
- Inheritance and abstraction
- Polymorphism
- Interfaces
- Delegates
- Events
- Exceptions
- Experience with the Microsoft .NET Framework
- Knowledge of the Visual Studio integrated development environment (IDE).

Course Outline

Module 1: Introducing Visual Basic and the .NET Framework

This module describes the purpose of the .NET Framework 4 and how you can build applications by using Visual Studio 2010.

Lessons
- Introduction to the .NET Framework 4
- Creating Projects Within Visual Studio 2010
- Writing a Visual Basic Application
- Building a Graphical Application
- Documenting an Application
- Debugging Applications by Using Visual Studio 2010

Lab : Introducing Visual Basic and the .NET Framework
- Creating a Simple Console Application
- Creating a WPF Application
- Verifying the Application
- Generating Documentation for an Application

After completing this module, students will be able to:

- Explain the purpose of the .NET Framework 4.
- Create Visual Basic projects by using Visual Studio 2010.
- Explain the structure of a Visual Basic application.
- Use the Windows Presentation Foundation (WPF) Application template to build a simple graphical application.
- Use XML comments to document an application.
- Use the debugger to step through a program.

Module 2: Using Visual Basic Programming Constructs

This module introduces many of the basic Visual Basic language data types and programming constructs, and describes the syntax and semantics of these constructs.

Lessons
- Declaring Variables and Assigning Values
- Using Expressions and Operators

WIZARDSGROUP

Microsoft Partner
Gold Learning
Gold Business Intelligence
Gold Portals and Collaboration
Gold Enterprise Resource Planning

- Creating and Using Arrays
- Using Decision Statements
- Using Iteration Statements

Lab : Using Visual Basic Programming Constructs

- Calculating Square Roots with Improved Accuracy
- Converting Integer Numeric Data to Binary
- Multiplying Matrices

After completing this module, students will be able to:

- Explain how to declare variables and assign values.
- Use operators to construct expressions.
- Create and use arrays.
- Use decision statements.
- Use iteration statements.

Module 3: Declaring and Calling Methods

A key part of developing any application is dividing the solution into logical components. In object-oriented languages such as Microsoft Visual Basic, a method is a unit of code that is designed to perform a discrete piece of work. This module introduces methods and describes how to define and use them.

Lessons

- Defining and Invoking Methods
- Specifying Optional Parameters and ByRef Parameters

Lab : Declaring and Calling Methods

- Calculating the Greatest Common Divisor of Two Integers by Using Euclid's Algorithm
- Calculating the GCD of Three, Four, or Five Integers
- Comparing the Efficiency of Two Algorithms
- Displaying Results Graphically

- Solving Simultaneous Equations (optional)

After completing this module, students will be able to:

- Describe how to create and invoke methods.
- Define and call methods that can take optional parameters and ByRef parameters.

Module 4: Handling Exceptions

Exception handling is an important concept and your applications should be designed with exception handling in mind. This module explains how you can implement effective exception handling in your applications, and how you can use exceptions in your methods to elegantly indicate an error condition to the code that calls your methods.

Lessons

- Handling Exceptions
- Raising Exceptions

Lab : Handling Exceptions

- Making a Method Fail-Safe
- Detecting an Exceptional Condition

After completing this module, students will be able to:

- Describe how to catch and handle exceptions.
- Describe how to create and raise exceptions.

Module 5: Reading and Writing Files

The ability to access and manipulate the files on the file system is a common requirement for many applications. This module shows how to read and write to files by using the classes in the Microsoft .NET Framework. This module also describes the different approaches that you can take, and how to read and write different formats of data.

Lessons

- Accessing the File System
- Reading and Writing Files by Using Streams

Lab : Reading and Writing Files

- Building a Simple File Editor
- Making the Editor XML Aware

After completing this module, students will be able to:

- Describe how to access the file system by using the classes that the .NET Framework provides.
- Describe how to read and write files by using streams.
- Describe how to use the **My** namespace for reading and writing files.

Module 6: Creating New Types

The Microsoft.NET Framework base class library consists of many types that you can use in your applications. However, in all applications, you must also build your own types that implement the logic for your solution.

This module explains how to create your own modules and types and describes the differences between reference types and value types.

Lessons

- Creating and Using Modules and Enumerations
- Creating and Using Classes
- Creating and Using Structures
- Comparing References to Values

Lab : Creating New Types

- Using Enumerations to Specify Domains
- Using a Structure to Model a Simple Type
- Using a Class to Model a More Complex Type
- Using a Nullable Structure

After completing this module, students will be able to:

- Describe how to create and use modules.
- Describe how to create and use enumerations.
- Describe how to create and use classes.
- Describe how to create and use structures.
- Explain the differences between reference and value types.

Module 7: Encapsulating Data and Methods

This module describes how to use some of the access modifiers that Visual Basic provides to enable you to implement encapsulation. This module also introduces the Shared modifier, which enables you to define members that can be shared over multiple instances of the same type.

Lessons

- Controlling Visibility of Type Members
- Sharing Methods and Data

Lab : Encapsulating Data and Methods

- Hiding Data Members
- Using Shared Members to Share Data
- Implementing an Extension Method

After completing this module, students will be able to:

- Describe how to control the visibility of type members.
- Describe how to share methods and data.

Module 8: Inheriting from Classes and Implementing Interfaces

This module introduces inheritance and interfaces in the Microsoft .NET Framework, and how you can use them to simplify complex problems, reduce code duplication, and speed up development.

Inheritance is a key concept in an object-oriented language. You can use inheritance, interfaces, and abstract classes to develop object hierarchies in your code. These object hierarchies can help

reduce bugs by defining clear contracts for what a class will expose and by providing default implementations where you can sensibly abstract code into a base type.

Lessons

- Using Inheritance to Define New Reference Types
- Defining and Implementing Interfaces
- Defining Abstract Classes

Lab : Inheriting from Classes and Implementing Interfaces

- Defining an Interface
- Implementing an Interface
- Creating an Abstract Class

After completing this module, students will be able to:

- Use inheritance to define new reference types.
- Define and implement interfaces.
- Define abstract classes.

Module 9: Managing the Lifetime of Objects and Controlling Resources

All applications use resources. When you build a Microsoft Visual Basic application, resources fall into two broad categories: managed resources that are handled by the common language runtime (CLR) and unmanaged resources that are maintained by the operating system outside the scope of the CLR. A managed resource is typically an object based on a class defined by using a managed language, such as Visual Basic. Examples of unmanaged resources include items implemented outside the Microsoft .NET Framework, such as Component Object Model (COM) components, file handles, database connections, and network connections.

Resource management is important in any applications that you develop. The NET Framework simplifies resource management by automatically reclaiming the resources by a managed object when it is no longer referenced by an application. Managed resources are handled by the .NET Framework garbage collector. However, unmanaged resources are not controlled by the garbage collector; you must take special steps to dispose them properly and prevent them from being held longer than necessary.

Lessons

- Introduction to Garbage Collection
- Managing Resources

Lab : Managing the Lifetime of Objects and Controlling Resources

- Implementing the IDisposable Interface
- Managing Resources Used by an Object

After completing this module, students will be able to:

- Describe how garbage collection works in the .NET Framework.
- Manage resources effectively in an application.

Module 10: Encapsulating Data and Defining Overloaded Operators

Many operators have well-defined behavior for the built-in Visual Basic types, but you can also define operators for your own types. This module describes how to implement operators for your types by using overloading.

Lessons

- Creating and Using Properties
- Creating and Using Indexers
- Overloading Operators

Lab : Creating and Using Properties

- Defining Properties in an Interface
- Implementing Properties in a Class
- Using Properties Exposed by a Class

Lab : Creating and Using Indexers

- Implementing a Default Property to Access Bits in a Control Register
- Using an Indexer Exposed by a Class

Lab : Overloading Operators

- Defining the Matrix and Matrix Not Compatible Exception Types

WIZARDSGROUP

Microsoft Partner
Gold Learning
Gold Business Intelligence
Gold Portals and Collaboration
Gold Enterprise Resource Planning

- Implementing Operators for the Matrix Type
- Testing the Operators for the Matrix Type

After completing this module, students will be able to:

- Explain how properties work and use them to encapsulate data.
- Describe how to use default properties to provide access to data through an array-like syntax.
- Describe how to use operator overloading to define operators for your own types.

Module 11: Decoupling Methods and Handling Events

This module explains how to decouple an operation from the method that implements it and how to use anonymous methods to implement decoupled operations. This module also explains how to use events to inform consuming applications of a change or notable occurrence in a type.

Lessons

- Declaring and Using Delegates
- Using Lambda Expressions
- Handling Events

Lab : Decoupling Methods and Handling Events

- Raising and Handling Events
- Using Lambda Expressions to Specify Code

After completing this module, students will be able to:

- Describe the purpose of delegates and explain how to use a delegate to decouple an operation from the implementing method.
- Explain the purpose of lambda expressions and describe how to use a lambda expression to define an anonymous method.
- Explain the purpose of events and describe how to use events to report that something significant has happened in a type that other parts of the application need to be aware of.

Module 12: Using Collections and Building Generic Types

The basic collection classes introduce a new problem. Classes that act on other types are often not type-safe. For example, many collection classes frequently use the Object type to store items, and must then be cast or converted back to their original type before they can be used.

It is the programmer's responsibility to ensure that the correct casts or conversions are performed, and it is easy to introduce errors by casting or converting an item to the wrong type. This module introduces generics and how you can use generic classes to maintain type-integrity and avoid issues that are associated with a lack of type safety.

Lessons

- Using Collections
- Creating and Using Generic Types
- Defining Generic Interfaces and Understanding Variance
- Using Generic Methods and Delegates

Lab : Using Collections

- Optimizing a Method by Caching Data

Lab : Building Generic Types

- Defining a Generic Interface
- Implementing a Generic Interface
- Implementing a Test Harness for the BinaryTree Project
- Implementing a Generic Method

After completing this module, students will be able to:

- Use collection classes.
- Define and use generic types.

- Define generic interfaces and explain the concepts of covariance and contra-variance.
- Define and use generic methods and delegates.

Module 13: Building and Enumerating Custom Collection Classes

When you develop applications, you often need to store collections of objects. In many circumstances, you can use the collection classes that the Microsoft .NET Framework includes; however, sometimes these collection classes do not provide the functionality that you require. For example, you may need to store objects in a sorted order that is based on a custom sorting algorithm.

This module introduces you to custom collection classes. It also explains how you can develop collection classes that support the language constructs that Visual Basic provides, such as enumeration and collection initialization.

Lessons
- Implementing a Custom Collection Class
- Adding an Enumerator to a Custom Collection Class

Lab : Building and Enumerating Custom Collection Classes
- Implementing the IList(Of TItem) Interface
- Implementing an Enumerator by Writing Code
- Implementing an Enumerator by Using an Iterator

After completing this module, students will be able to:

- Implement a custom collection class.
- Define an enumerator in a custom collection class

Module 14: Using LINQ to Query Data

This module introduces you to Language-Integrated Query (LINQ) queries and explains how you can use them to process data in your Microsoft .NET Framework applications.

This module also explains the difference between shared and dynamic LINQ queries, and describes how you can use dynamic LINQ to create highly flexible queries that you build at run time.

Lessons
- Using the LINQ Extension Methods and Query Operators
- Building Dynamic LINQ Queries and Expressions

Lab : Using LINQ to Query Data
- Using the LINQ Query Operators
- Building Dynamic LINQ Queries

After completing this module, students will be able to:

- Describe how to use the LINQ extension methods and query operators.
- Describe how to build dynamic LINQ queries and expressions.

Module 15: Integrating Visual Basic Code with Dynamic Languages and COM Components

Integration with other technologies is a key feature of the Microsoft.NET Framework. Previous versions of the .NET Framework enabled you to combine components that were developed by using different languages that have compilers that the .NET Framework supports.

The .NET Framework 4 now supports integration of components built by using dynamic languages. This enables you to re-use items built by using a wide range of scripting languages that are not easily accessible from Microsoft Visual Basic code.

In addition, previous versions of the .NET Framework have always enabled you to integrate Component Object Model (COM) services and components into your managed applications. The integration did however, require a good

understanding of the differences between the way in which the common language runtime (CLR) and the COM environment operated. The new features of Visual Basic 2010 have simplified the way in which you can invoke COM components, so it is easier for you to re-use these items in a Visual Basic application.

This module describes how to integrate code written by using a dynamic language such as Ruby and Python, or technologies such as COM, into a Visual Basic application.

Lessons

- Integrating Visual Basic Code with Ruby and Python
- Accessing COM Components from Visual Basic

Lab : Integrating Visual Basic Code with Dynamic Languages and COM Components

- Integrating Code Written by Using a Dynamic Language into a Visual Basic Application
- Using a COM Component from a Visual Basic Application

After completing this module, students will be able to:

- Integrate Ruby and Python code into a Visual Basic application.
- Invoke COM components and services from a Visual Basic application.

Course Inclusions:

- Microsoft Official Curriculum (MOC) and/or Wizards Learning Courseware (WLC)
- Microsoft Certified Trainer (MCT)
- Lunch, AM and PM Snacks
- Certificate of Achievement
- Course Note