

MOC 10550 A: Programming in Visual Basic with Microsoft Visual Studio

Course Summary

NOTE - The course delivery has been updated to Visual Studio 2013 and .NET Framework 4.5.5!

Description

This course teaches you Visual Basic language syntax, program structure, and implementation by using Microsoft Visual Studio 2013 and the Microsoft .NET Framework 4.5. This course also provides a solid foundation in Visual Basic to the level necessary to enable students to attend other courses in the Technical Specialist tracks.

Objectives

By the end of this course, students will be able to:

- Describe the purpose of the .NET Framework, and explain how to use Microsoft Visual Basic® and Visual Studio 2013 to build .NET Framework applications.
- Describe the syntax of basic Visual Basic programming constructs.
- Describe how to create and call methods.
- Describe how to catch, handle, and throw exceptions.
- Describe how to perform basic file I/O operations in a Visual Basic application.
- Describe how to create and use new types (enumerations, classes, and structures), and explain the differences between reference types and value types.
- Describe how to control the visibility and lifetime of members in a type.
- Describe how to use inheritance to create new reference types.
- Describe how to manage the lifetime of objects and control the use of resources.
- Describe how to create properties and indexers to encapsulate data, and explain how to define operators for this data.
- Describe how to decouple an operation from the method that implements it, and explain how to use these decoupled operations to handle asynchronous events.
- Describe the purpose of collections, and explain how to use generics to implement type-safe collection classes, structures, interfaces, and methods.
- Describe how to implement custom collection classes that support enumeration.
- Describe how to query in-memory data by using Language-Integrated Query (LINQ) queries.
- Describe how to integrate code written by using a dynamic language such as Ruby and Python, or technologies such as Component Object Model (COM), into a Visual Basic application.

Topics

- Introducing Visual Basic and the .NET Framework
- Using Visual Basic Programming Constructs
- Declaring and Calling Methods
- Handling Exceptions
- Reading and Writing Files
- Creating New Types
- Encapsulating Data and Methods
- Inheriting from Classes and Implementing Interfaces
- Managing the Lifetime of Objects and Controlling Resources
- Encapsulating Data and Defining Overloaded Operators
- Decoupling Methods and Handling Events
- Using Collections and Building Generic Types
- Building and Enumerating Custom Collection Classes
- Using LINQ to Query Data
- Integrating Visual Basic Code with Dynamic Languages and COM Components

MOC 10550 A: Programming in Visual Basic with Microsoft Visual Studio

Course Summary (cont'd)

Audience

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 4.5.0. This course also focuses on new enhancement in the Visual Basic 12 language using Visual Studio 2013.

Prerequisites

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).

Duration

Five days

MOC 10550 A: Programming in Visual Basic with Microsoft Visual Studio

Course Outline

I. Introducing Visual Basic and the .NET Framework

This module describes the purpose of the .NET Framework 4.5 and how you can build applications by using Visual Studio 2013.

- A. Introduction to the .NET Framework 4.5
- B. Creating Projects Within Visual Studio 2013
- C. Writing a Visual Basic Application
- D. Building a Graphical Application
- E. Documenting an Application
- F. Debugging Applications by Using Visual Studio 2013

Lab : Introducing Visual Basic and the .NET Framework

II. 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.

- A. Declaring Variables and Assigning Values
- B. Using Expressions and Operators
- C. Creating and Using Arrays
- D. Using Decision Statements
- E. Using Iteration Statements

Lab : Using Visual Basic Programming Constructs

III. 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.

- A. Defining and Invoking Methods
- B. Specifying Optional Parameters and ByRef Parameters

Lab : Declaring and Calling Methods

IV. 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.

- A. Handling Exceptions

- B. Raising Exceptions

Lab : Handling Exceptions

V. 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.

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

Lab : Reading and Writing Files

VI. 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.

- A. Creating and Using Modules and Enumerations
- B. Creating and Using Classes
- C. Creating and Using Structures
- D. Comparing References to Values

Lab : Creating New Types

VII. 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.

- A. Controlling Visibility of Type Members
- B. Sharing Methods and Data

Lab : Encapsulating Data and Methods

VIII. 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

MOC 10550 A: Programming in Visual Basic with Microsoft Visual Studio

Course Outline (cont'd)

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.

- A. Using Inheritance to Define New Reference Types
- B. Defining and Implementing Interfaces
- C. Defining Abstract Classes

Lab : Inheriting from Classes and Implementing Interfaces

IX. 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.

- A. Introduction to Garbage Collection
- B. Managing Resources

Lab : Managing the Lifetime of Objects and Controlling Resources

X. 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.

- A. Creating and Using Properties
- B. Creating and Using Indexers
- C. Overloading Operators

Lab : Creating and Using Properties

Lab : Creating and Using Indexers

Lab : Overloading Operators

XI. Decoupling Methods and Handling Events

XII. 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.

- A. Declaring and Using Delegates
- B. Using Lambda Expressions
- C. Handling Events

Lab : Decoupling Methods and Handling Events

XIII. 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.

- A. Using Collections
- B. Creating and Using Generic Types
- C. Defining Generic Interfaces and Understanding Variance
- D. Using Generic Methods and Delegates

Lab : Using Collections

Lab : Building Generic Types

XIV. 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

MOC 10550 A: Programming in Visual Basic with Microsoft Visual Studio

Course Outline (cont'd)

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.

- A. Implementing a Custom Collection Class
- B. Adding an Enumerator to a Custom Collection Class

Lab : Building and Enumerating Custom Collection Classes

XV. 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.

- A. Using the LINQ Extension Methods and Query Operators
- B. Building Dynamic LINQ Queries and Expressions

Lab : Using LINQ to Query Data

XVI. 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 4.5 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 2013 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.

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

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