



## Course Outline

### **Introduction to Programming Microsoft .NET Framework Applications with Microsoft Visual Studio® 2005**

Course 4994A: Five days; Instructor-Led

Note: You are viewing a Preliminary Course Syllabus. This course is not yet available. Because some parts of the course are currently in development, some elements of this syllabus are subject to change.

#### **Introduction**

This five-day instructor-led course enables introductory-level developers who are not familiar with the Microsoft® .NET Framework or Microsoft Visual Studio® 2005 to gain familiarity with the Visual Studio 2005 development environment. Students will also learn basic skills using either Microsoft Visual Basic® or Microsoft Visual C#® as a programming language.

#### **Audience**

The target audience for this course includes both novice programmers who have a minimum of three months' programming experience and intermediate-level programmers who are otherwise new to .NET Framework development, and want to learn how to use Visual Basic or C#.

#### **At Course Completion**

After completing this course, students will be able to:

- Describe the key features of the .NET Framework and Visual Studio 2005.
- Create a simple Windows Forms application.
- Explain programming fundamentals.
- Create and use data types and variables.
- Control program execution by using conditional statements and loops.
- Explain the fundamentals of object-oriented programming.
- Create simple object-oriented applications.
- Develop the user interface in a Visual Studio 2005 application.
- Validate user input on a Windows form.
- Implement debugging and exception handling in a Visual Studio 2005 application.
- Access data in a Visual Studio 2005 application.
- Create simple Web applications and XML Web services.
- Explain the key features of the .NET Framework version 3.0 technologies.
- Test and deploy Microsoft .NET Framework applications.

#### **Prerequisites**

Before attending this course, students must have:

- Exposure to developing applications in either a graphical or a non-graphical environment.
- Ability to understand and apply the basics of structured programming, including concepts such as flow control, variables, parameters, and function calls.



## Course Outline

In addition, it is recommended, but not required, that students have completed:

- Course 2667: Introduction to Programming.

### Course Outline

#### *Module 1: Getting Started*

This module introduces the .NET Framework and the software development life cycle. It also describes the key features of Visual Studio 2005.

##### Lessons

- Introduction to Microsoft .NET and the .NET Framework
- Introduction to the Software Development Life Cycle
- Exploring Visual Studio 2005

#### **Lab 1: Getting Started**

- Working in the Development Environment

After completing this module, students will be able to:

- Describe Microsoft .NET and the .NET Framework.
- Describe the software development life cycle.
- Explain the key features of Visual Studio 2005.

#### *Module 2: Creating a Simple Windows Forms Application*

This module explains how to create a Windows Forms application, how to add controls to a form, and how to compile and run the application.

##### Lessons

- Creating a Windows Forms Project
- Adding Controls to a Windows Forms Project
- Compiling and Running a Windows Forms Project

#### **Lab 2: Creating a Simple Windows Forms Application**

- Creating a Windows Forms Application



## Course Outline

- Adding Controls to the Main Form
- Compiling and Testing the Application

After completing this module, students will be able to:

- Create a Windows Forms project.
- Add controls to a Windows Forms project.
- Compile and run a Windows Forms project.

### ***Module 3: Programming Fundamentals***

This module explains important programming concepts and terminology. It also covers the main elements of a program and explains how to create and work with items such as functions, properties, and methods. Finally, this module provides guidelines on areas such as naming conventions and code documentation.

#### **Lessons**

- Understanding Programming Concepts
- Defining Program Structure and Flow
- Styling and Writing Code

#### **Lab 3: Programming Fundamentals**

- Displaying the Current Date on a Form
- Adding a New Form to the Application
- Adding Controls to the New Form

After completing this module, students will be able to:

- Explain basic programming concepts.
- Define program structure and flow.
- Explain guidelines for styling and writing code.

### ***Module 4: Data Types and Variables***

This module introduces data types, variables, and constants and explains how to use them. It also explains how to use collections and data type conversion.

#### **Lessons**

- Introduction to Data Types
- Defining and Using Variables
- Defining and Using Collections
- Converting Data Types

#### **Lab 4: Data Types and Variables**

- Implementing Variables and Constants
- Implementing Arrays and Enumerations



## Course Outline

After completing this module, students will be able to:

- Explain the main features of data types.
- Define and use variables.
- Define and use collections.
- Explain data type conversion.

### ***Module 5: Controlling Program Execution***

This module describes how to control program execution by writing expressions, conditional statements, and iteration statements.

#### **Lessons**

- Writing Expressions
- Creating Conditional Statements
- Creating Iteration Statements

#### **Lab 5: Controlling Program Execution**

- Checking User Input
- Enabling and Disabling Controls

After completing this module, students will be able to:

- Write expressions that contain operators.
- Create conditional statements.
- Create iteration statements.

### ***Module 6: Fundamentals of Object-Oriented Programming***

This module introduces students to the concepts of object-oriented programming, defines important terminology, and shows the syntax for defining classes and creating instances.

#### **Lessons**

- Introduction to Object-Oriented Programming
- Defining a Class
- Creating a Class Instance

#### **Lab 6: Fundamentals of Object-Oriented Programming**

- Creating a SalesPerson Class
- Creating and Using a SalesPerson Object

After completing this module, students will be able to:





## Course Outline

- Describe the essential features of object-oriented programming.
- Define a class.
- Create a class instance.

### ***Module 7: Creating Object-Oriented Applications***

This module describes how to design classes by using the Class Designer tool in Visual Studio, and also describes how to use inheritance and interfaces.

#### **Lessons**

- Designing Classes with the Class Designer Tool
- Implementing Inheritance
- Defining and Implementing Interfaces

#### **Lab 7: Creating Object-Oriented Applications**

- Creating a Base Class
- Creating Derived Classes

After completing this module, students will be able to:

- Design classes with the Class Designer tool.
- Implement inheritance.
- Define and implement interfaces.

### ***Module 8: Building a User Interface***

This module explains how to develop an application by using features such as modal and modeless forms, menus, toolbars, status bars, tool tips, and the HelpProvider control.

#### **Lessons**

- Managing Forms and Dialog Boxes
- Creating Menus and Toolbars
- Providing User Assistance

#### **Lab 8: Building a User Interface**

- Adding a Menu and a Toolbar to an Application
- Adding a Status Bar and Tooltips to an Application

After completing this module, students will be able to:

- Manage forms and dialog boxes.
- Create menus and toolbars.
- Provide user assistance.



## Course Outline

### ***Module 9: Validating User Input***

This module explains how to restrict user input on a form, and how to use field-level and form-level validation.

#### **Lessons**

- Restricting User Input
- Implementing Field-Level Validation
- Implementing Form-Level Validation

#### **Lab 9: Validating User Input**

- Adding an ErrorProvider Component to a Form
- Providing Visual Cues to the User by Enabling an OK Button

After completing this module, students will be able to:

- Restrict user input.
- Implement field-level validation.
- Implement form-level validation.

### ***Module 10: Debugging and Exception Handling***

This module introduces students to the types of errors that can occur in an application, and describes how to use a combination of debugging and exception handling to detect and diagnose these