

Core Foundations of Microsoft .NET 2.0 Development

Introduction

Elements of this syllabus are subject to change.

This three-day instructor-led course provides students with the enabling knowledge and skills required to create Microsoft .NET Applications with Visual Studio 2005. Students learn how to develop advanced .NET applications.

Audience

The audience for this course consists of Application Developers with the skills to develop business applications by using Visual Studio 2005 with either Visual Basic .NET or Visual C#.

At Course Completion

After completing this course, students will gain the skills to:

- Develop applications that use system types and collections.
- Implement service processes, threading, and application domains in a .NET Framework application.
- Embed configuration, diagnostic, management, and installation features into a .NET Framework application.
- Implement serialization and input/output functionality in a .NET Framework application.

Prerequisites

Before attending this course, students must be able to:

- Understand the purpose and components of the .NET 2.0 Framework and the Common Language Runtime.
- Understand the components of typical .NET 2.0 applications.
- Understand and use .NET Framework 2.0 Common Type System (CTS) and how to use variable types including dates/times, numbers, strings, objects and arrays.
- Use basic file IO classes from the Framework such as StreamReader, StreamWriter, DirectoryInfo, File and FileInfo.
- Use basic Framework provided type conversions.
- Use basic Framework provided text conversion and manipulations including StringBuilder.
- Use classes with the System.Collections namespace.
- Use the System.Math class.
- Basic language syntax for decision structures, loop structures, declaring

and using variables.

- Write code using language specific functionality such as the My. classes for Visual Basic.
- Understand classes and objects, methods, properties and functions.
- Write code to implement overridden methods.
- Understand the class hierarchy present in the .NET Framework 2.0.
- Write code to declare a class.
- Write code to create an instance of a class.
- Write code to compare if an object is equal to another object.
- Write code to dispose of an object.
- Understand the lifecycle of an object.
- Write code to handle exceptions via a try-catch block
- Write code to implement static methods and properties.
- Opening and closing solutions.
- Opening and closing projects.
- Adding projects to a solution.
- Removing projects from a solution.
- Creating new project types.
- Adding new and existing files to a project.
- Compile a project.
- Carry out basic project debugging.
- Use the object browser.
- Use the help system especially provided to help VB6.0 developers migrate to .NET.
- Understand assemblies and how they relate to deployment.
- Understand and create a deployment project.
- Be able to create deployment wizards using the Deployment Setup wizard.
- Select an appropriate deployment project based on the application.

Important: This learning product will be most useful to people who are already working in the job role of an application developer and who intend to use their new skills and knowledge on the job immediately after training.

Microsoft Certified Professional Exams

No Microsoft Certified Professional exams are associated with this course currently.

Course Materials

The student kit includes a comprehensive workbook and other necessary materials for this class.

The following software is provided in the student kit:

- Student CD

Course Outline

Module 1: Implementing System Types and Interfaces

In this module, students learn about the purpose of system types in the .NET Framework and implementation of special system types introduced in the .NET Framework 2.0. Students also learn about the purpose of interfaces in developing .NET Framework applications. Finally, students learn how to implement system types and interfaces.

Lessons

- Examining Primary System Types
- Working with Special System Types
- Working with Interfaces
- Lab: Implementing System Types and Interfaces

After completing this module, students will be able to:

- Explain the purpose of base system types.
- Implement generics, Nullable types, exception classes, and attributes.
- Implement comparison interfaces and the IConvertible, ICloneable, IFormattable, and IDisposable interfaces.

Module 2: Implementing Collections and Generics

In this module, students learn the basic information on how to work with primary collections, generic collections, specialized collections, and collection base classes.

Lessons

- Examining Collections and Collection Interfaces
- Working with Primary Collection Types
- Working with Generic Collections
- Working with Specialized Collections
- Working with Collection Base Classes
- Lab: Implementing Collections and Generics

After completing this module, students will be able to:

- Describe the purpose of collections and collection interfaces.
- Implement the various classes available in the .NET Framework 2.0.
- Implement generic list types, collections, dictionary types, and linked-list types.

- Implement the specialized string and named collection classes.
- Implement collection base classes and dictionary base types.

Module 3: Configuring and Installing Assemblies

In this module, students learn how to create, share, install, and configure assemblies in the .NET Framework. Students also learn how to perform installation tasks related to assembly installation.

Lessons

- Working with an Assembly
- Sharing an Assembly by Using the Global Assembly Cache
- Installing an Assembly by Using Installation Types
- Configuring an Assembly by Using Configuration Type
- Performing Installation Tasks
- Lab: Configuring and Installing Assemblies

After completing this module, students will be able to:

- Describe the purpose of an assembly and explain how to create the same.
- Share an assembly by using the global assembly cache.
- Install an assembly by using the Installer, AssemblyInstaller, ComponentInstaller, InstallerCollection, and InstallContext classes and the InstallEventHandler delegate available in the .NET Framework 2.0.
- Configure an assembly by using the Configuration, Configuration Element, Configuration Section classes and the configuration base types available in the .NET Framework 2.0.
- Perform various installation tasks related to assembly installation.

Module 4: Monitoring and Debugging Applications

In this module, students learn how to manage event logs and application processes. Students also learn how to monitor application performance, debug and trace applications, and embed management information and events in the .NET Framework applications.

Lessons

- Managing an Event Log
- Working with Application Processes
- Managing Application Performance
- Debugging Applications
- Tracing Applications
- Embedding Management Information and Events
- Lab: Monitoring and Debugging Applications

After completing this module, students will be able to:

- Describe event logs and explain how to manage them.
- Manage application processes by retrieving a list of all processes running on the current system, information about the current process, and a list of all modules used by a process, and by starting and stopping an application process.
- Monitor and customize application performance by using the Windows Performance Monitor and the performance counter classes available in the .NET Framework 2.0.
- Debug applications by using the Visual Studio 2005 Debugger, the Debugger, Debug, StackFrame and StackTrace classes, and the Debugger attributes available in the .NET Framework 2.0.
- Trace applications by using the Trace, TraceSource, Trace Switch, Trace Listener, and CorrelationManager classes available in the .NET Framework 2.0.
- Embed management information and events in the .NET Framework 2.0 applications.

Module 5: Reading and Writing Files

In this module, students learn how to manage drives, directories, and files. Students also learn how to work with streams, text, and strings. Finally, students learn how to compress, decompress, and search for patterns within file contents.

Lessons

- Managing the File System
- Working with Byte Streams
- Compressing and Protecting Stream Information
- Managing Application Data
- Manipulating Strings Efficiently
- Working with Regular Expressions
- Lab: Reading and Writing Files

After completing this module, students will be able to:

- Manage the file system by using the Path, File, FileInfo, Directory, DirectoryInfo, DriveInfo, and FileSystemWatcher classes.
- Work with byte streams by using the Stream, FileStream, MemoryStream, and BufferedStream classes.
- Compress and protect stream information by using the DeflateStream, GZipStream, IsolatedStorageFile, and IsolatedStorageFileStream classes.
- Manage application data by using the TextReader, TextWriter, StreamReader, StreamWriter, StringReader, StringWriter, BinaryReader, and BinaryWriter classes.
- Manipulate strings efficiently by using the StringBuilder class.
- Work with regular expressions by using the regular expression classes.

Module 6: Serializing Data

In this module, students learn how to serialize objects into binary and Simple Object Access Protocol (SOAP) formats. The students also learn how to serialize objects into custom XML and how to create custom serialization classes.

Lessons

- Generating Serialized Binary and Soap Formats
- Generating Serialized XML Formats
- Creating Custom Serialization Classes
- Lab: Serializing Data

After completing this module, students will be able to:

- Serialize objects into binary and SOAP formats by using the BinaryFormatter and SoapFormatter classes.
- Serialize objects into custom XML formats by using the XmlSerializer class, the IXmlSerializable interface, and the XML serialization attributes and delegates.
- Create custom serialization classes by using serialization types and interfaces, formatter classes, event handler attributes, and the ObjectManager class.

Module 7: Implementing Delegates and Events

In this module, students learn the concepts of delegates and events and their uses in the .NET framework.

Lessons

- Controlling Interaction Between Components by Using Delegates
- Controlling Interaction Between Components by Using Events
- Lab: Implementing Delegates and Events

After completing this module, students will be able to:

- Control interaction between components by using the Delegate class.
- Control interaction between components by using the Event statement, the EventHandler delegate, and the EventArgs class.