

## Advanced Foundations of Microsoft .NET 2.0 Development

**COURSE CODE: MS2957 Three days; Instructor-Led**

### Prerequisites

Before attending this course, students must have:

- Understand the purpose and components of the .NET Framework 2.0 and the common language runtime.
- Understand and use the .NET Framework 2.0 common type system (CTS).
- Understand basic language syntax for decision structures, loop structures, and variables.
- Write code by using language-specific functionality such as the My. classes for Visual Basic.
- Understand and use classes, objects, methods, properties, and functions.
- Write code to implement overridden methods, static (Visual C#) or Shared (Visual Basic) methods, and properties.
- Use type conversions and text conversions.
- Create and use solutions and projects by using Visual Studio 2005.
- Use the Visual Studio 2005 object browser and the Visual Studio help system.

### Course Outline

#### Module 1: Enhancing User Interfaces by Using System.Drawing

This module describes the key features of the **System.Drawing** namespace that the .NET Framework provides. It also explains how to create and modify your own custom drawings.

#### Lessons

- Drawing Fundamentals
- Drawing Lines and Shapes
- Rendering Bitmaps and Icons

#### Lab : Drawing to a Windows Form

- Drawing a Feedback Bar
- Drawing a Feedback Pie Chart
- Implementing an Automatic Double Buffer
- Adding Fonts to Your Application
- Saving Your Scaled Image

#### AVANTUS TRAINING PTE LTD

79 Robinson Road #15-04 CPF Building Singapore 068897

Sales Hotline: (65)64163078

Email: [enquiries@AvantusTraining.com](mailto:enquiries@AvantusTraining.com)

[www.AvantusTraining.com](http://www.AvantusTraining.com)

After completing this module, students will be able to:

- Use points, sizes, brushes, pens, colors, and fonts.
- Draw lines and shapes.
- Create and use images, bitmaps, and icons.

## **Module 2: Working with Cultures by Using System.Globalization**

This module describes how to use the **System.Globalization** namespace to work with culture information and perform culture-sensitive string comparisons. It also describes how to create a custom culture.

### **Lessons**

- Working with Culture Information
- Formatting and Sorting Culture-Sensitive Data
- Creating a Custom Culture

### **Lab : Working with Cultures by Using System.Globalization**

- Managing Culture Information
- Creating a Custom Culture

After completing this module, students will be able to:

- Explain the purpose of the **System.Globalization** namespace and describe how to access culture information by using the **CultureInfo** class.
- Format values by using the supporting classes in the **System.Globalization** namespace and explain how to perform culture-sensitive string comparisons.
- Create a custom culture by using the **CultureAndRegionInfoBuilder** class.

### **Module 3: Processing Text by Using Regular Expressions and Encodings**

This module describes the key features of the **System.Text** namespace that the .NET Framework provides. It explains how to store and manipulate strings, how and when to implement regular expressions, and how to customize encodings to produce the correct results when you process text

## Lessons

- Handling Text and Large Strings
- Using Regular Expressions
- Encoding Text

Lab : Processing Text by Using Regular Expressions and Encodings

- Handling Text and Strings
- Creating and Using Regular Expressions
- Working with **Encodi**

After completing this module, students will be able to:

- Explain the purpose of and use the **StringBuilder** class.
- Describe the purpose of and create regular expressions by using the classes in the **System.Text.RegularExpressions** namespace.
- Describe text encoding and how to encode and decode text by using the encoding classes.

## Module 4: Encrypting and Hashing Data by Using Cryptography

This module describes when to use data encryption and hashing and explains how to use the classes in the .NET Framework 2.0 to perform these cryptographic tasks. It also discusses how to customize the implementation of specific algorithms by extending base classes in the **System.Security.Cryptography** namespace.

## Lessons

- Working with Encryption and Hashing
- Encrypting and Decrypting Data
- Hashing Data
- Extending Cryptography

## Lab : Creating a Cryptographic Application

- Creating an Asymmetric Key
- Encrypting a File
- Decrypting a File
- Exporting and Importing a Public Key
- Getting a Private Key

After completing this module, students will be able to:

- Explain the purpose of encryption and hashing algorithms.
- Describe the algorithms available for hashing and encryption.
- Describe the difference between symmetrical and asymmetrical algorithms.
- Encrypt and decrypt data by using the classes in the **System.Security.Cryptography** namespace.
- Create custom classes that extend the .NET Framework cryptography model.

## Module 5: Securing Code Execution and Resources

This module describes how code access security (CAS) works, how to make CAS permission checks in code, and how to configure CAS security policy. It also introduces Windows operating system access checks and explains how to use the .NET Framework base class library to read and modify Windows access control lists (ACLs). This module also describes how .NET Framework security performs authorization checks and how to customize the mechanism.

### Lessons

- Using Code Access Security
- Securing Code Execution by Using Policy
- Securing Resources by Using Access Control
- Customizing Authentication and Authorization

### Lab : Custom Authentication by Using Principal Objects

- Viewing the Starter Solution
- Adding Role-Based Security
- Testing the Application

After completing this module, students will be able to:

- Describe and implement CAS by using the **System.Security** and **System.Security.Permissions** namespaces.
- Explain how to control code privileges by using the **System.Security.Policy** namespace, and describe how to use the utilities provided with the .NET Framework to manage computer, user, and enterprise-level policy.
- Describe and use access control by using the classes in the **System.Security.AccessControl** namespace.
- Explain how to authenticate and authorize users by using the **System.Security.Principal** namespace.

## Module 6: Application Interoperability

This module describes how to invoke functions that are implemented in unmanaged DLLs, and how to use various techniques for integrating Component Object Model (COM) components into managed applications. It also explains how to make managed components that are built by using the .NET Framework available to unmanaged COM client applications

### Lessons

- Using the Platform Invoke Service
- Integrating COM Components into a .NET Framework Application
- Integrating Managed Components into an Unmanaged Application

### Lab : Application Interoperability

- Integrating Unmanaged Functions into a Managed Application
- Integrating a COM Component into a Managed Application by Creating an **Interop** Assembly
- Integrating a COM Component into a Managed Application by Using Late Binding
- Manually Creating an **Interop** Assembly for a COM Component

After completing this module, students will be able to:

- Use the Platform Invoke service to incorporate unmanaged functions into a .NET Framework application.
- Integrate unmanaged COM components into a .NET Framework application.
- Incorporate components that are built by using the .NET Framework into unmanaged applications.

## Module 7: Reflection, Metadata, and Emitting Objects

This module describes how to use the classes in the .NET Framework 2.0 class library to examine a program, alter the behavior or structure of the program as it runs, and create and run new code.

### Lessons

- Reflecting on Objects
- Adding Assembly Metadata
- Emitting Objects by Using Builder Classes

### Lab : Creating an Add-in Framework by Using Reflection

- Creating a Custom Attribute to Decorate Add-in Classes
- Using Reflection to Discover the Add-in Classes
- Creating a Mapping from Columns to Properties

- Creating a Dynamic Method to Set a Property Value
- Invoking the Row Handler

After completing this module, students will be able to:

- Explain and use reflection in .NET Framework applications by using the **System.Reflection** namespace.
- Describe and create application metadata.
- Describe and create Microsoft intermediate language (MSIL) and portable executable (PE) files by using the **System.Reflection.Emit** namespace.

## **Module 8: Services, Threading, and Application Domains**

This module describes how to use the Microsoft .NET Framework classes to write Windows services and classes to install services. It also explains how to use the .NET Framework classes to create thread objects to execute code and to synchronize execution between threads. Finally, this module describes how to access and configure application domains.

### **Lessons**

- Creating and Installing Windows Services
- Creating Multithreaded Applications
- Manually Working with Application Domains

### **Lab : Creating a Windows Service Application**

- Creating a Service Project
- Creating the Installation Project
- Creating a Client Application
- Writing the Service Code

After completing this module, students will be able to:

- Explain how to create, install, and control a Windows service by using the **System.ServiceProcess** namespace.
- Describe and implement multithreaded applications by using the **System.Threading** namespace.
- Describe and create application domains.