

## Advanced Foundations of Microsoft .NET 2.0 Development

### Course 2957A: Three days; Instructor-Led Preliminary Course Syllabus

**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

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 secured .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:

- Improve the security of .NET Framework applications by using the .NET Framework 2.0 security features.
- Implement interoperability, reflection, and mailing functionality in a .NET Framework application.
- Implement globalization, drawing, and text manipulation 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, Directory, 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: Creating Globalized Applications

In this module, students are introduced to the benefits of globalization and localization. Students also learn about the globalization and localization techniques.

#### Lessons

- Culture Information by Using Globalization Classes
- Creating a Custom Culture
- Working with Primary Encoding Classes
- Working with Advanced Encoding Classes
- Lab: Creating Globalized Applications

After completing this module, students will be able to:

- Work with culture information by using the `CultureInfo`, `RegionInfo`, `DateTimeFormatInfo`, `NumberFormatInfo`, and `CompareInfo` classes.
- Create a custom culture by using the `CultureAndRegionInfoBuilder` class.
- Encode characters by using the `Encoding`, `EncodingInfo`, `ASCIIEncoding`, `UTF8Encoding`, and `UnicodeEncoding` classes.
- Handle failure events by using the `Encoder`, `EncoderFallback`, `Decoder`, and `DecoderFallback` classes.

### Module 2: Working with GDI+ in Windows-based Applications

In this module, students learn how to use the Graphics Device Interface (GDI+) in applications that are based on Windows Forms by using the .NET Framework.

#### Lessons

- Working with Graphics, Brushes, Pens, Colors, and Fonts
- Manipulating the Shapes and Sizes of Graphical Objects
- Working with Images, Bitmaps, and Icons
- Lab: Working with GDI+ in Windows-based Applications

After completing this module, students will be able to:

- Create graphical objects by using the `Graphics`, `Pen`, `Brush`, and `Font` classes and `Color` types.
- Manipulate the shapes and sizes of graphical objects by using the `Point` and `Size` types.
- Add images and icons to the drawing surface by using the `Image`, `Bitmap`, and `Icon` classes.

### Module 3: Implementing Code Access Security

In this module, students learn about the code access security mechanisms that can help protect applications not only against untrusted users, but also against some of the subtler problems of malicious code, which may be executed unsuspectingly by trusted users.

#### Lessons

- Configuring Code Access Security
- Managing Security Policy

- Managing Permissions
- Managing Access Control
- Managing User Identity Information
- Lab: Implementing Code Access Security

After completing this module, students will be able to:

- Configure code access security by using the .NET Framework 2.0 Configuration tool and Evidence types.
- Manage security policy by using the SecurityManager, Code Group, PolicyLevel, PolicyStatement, Condition, IApplicationTrustManager, and IMembershipCondition types.
- Manage permissions by using the CodeAccessPermission, PermissionSet, and NamedPermissionSet classes and security permission types.
- Manage access control by using the access control list (ACL) and resource security classes.
- Manage user identity information by using the GenericIdentity, GenericPrincipal, WindowsIdentity, WindowsPrincipal, Identity Reference, and WindowsImpersonationContext classes.

#### **Module 4: Implementing Cryptography**

In this module, students learn about the new cryptographic types offered by the .NET Framework 2.0, and significant enhancements to the existing types that support symmetric and asymmetric encryption and hashing. Students also learn how to use cryptographic types in .NET Framework applications to ensure secure communication and the protection of sensitive data.

##### **Lessons**

- Encrypting Data
- Hashing Data
- Extending the Cryptographic Behavior

- Lab: Implementing Cryptography

After completing this module, students will be able to:

- Encrypt data by using symmetric and asymmetric algorithm classes and the SslStream class.
- Hash data by using Message Digest Algorithm 5 (MD5), Secure HashAlgorithm 1 (SHA1), and Hash-based Message Authentication Code (HMAC) classes.
- Extend the cryptographic behavior by using CryptoStream, CryptoConfig, ProtectedData, ProtectedMemory, CspParameters, CryptoAPITransform, and RandomNumberGenerator classes.

#### **Module 5: Interoperating Between COM Components and Assemblies**

In this module, students learn how to create .NET Framework applications that can communicate with COM components and unmanaged DLLs. Students also explore how to use COM components in a .NET Framework application and design your .NET Framework application so that it can be called by a COM component.

##### **Lessons**

- Accessing COM Components by Using Interop Services
- Exposing an Assembly to COM Components by Using Interop Services
- Accessing COM Components by Using Platform Invocation Services

- Lab: Interoperating Between COM Components and Assemblies

After completing this module, students will be able to:

- Access COM components by using Interop services.

- Expose an assembly to COM components by using Interop services.
- Access COM components by using Platform Invocation Services.

### **Module 6: Working with Service Applications and E-mail Messages**

In this module, students learn how the .NET Framework simplifies the process of creating service applications by providing the classes necessary to create, install, debug, and monitor service applications. Students also learn how to send e-mail messages from your service application.

#### **Lessons**

- Working with a Windows Service Application
- Working with E-mail Messages
- Lab: Working with Service Applications and E-mail Messages

After completing this module, students will be able to:

- Manage a Windows service application by using the ServiceBase, ServiceInstaller, ServiceProcessInstaller, and ServiceController classes.
- Work with e-mail messages by using the MailMessage, MailAddress, MailAddressCollection, MailAttachment, SmtpClient, SmtpException, and SmtpFailedRecipientException classes and the SendCompleteEventHandler delegate.

### **Module 7: Working with Type Metadata**

In this module, students learn how to retrieve the type metadata for an assembly. Students also learn how to use attributes to control the metadata that is created for their assembly. Finally, students also learn how to dynamically create assemblies at runtime by using the builder classes in the System.Reflection namespace.

#### **Lessons**

- Working with Type Metadata by Using Pre-defined Assembly Classes
- Working with Assemblies Dynamically by Using Custom Classes
- Lab: Working with Type Metadata

After completing this module, students will be able to:

- Work with type metadata by using the Assembly, MemberInfo, MethodBody, and LocalVariableInfo types and assembly attributes.
- Work with assemblies dynamically by using builder classes and binding types.

### **Module 8: Creating Multithreaded Applications and Application Domains**

In this module, students learn about several classes in the System.Threading namespace, provided by the .NET Framework, to manage threads of execution.

#### **Lessons**

- Managing Threads in a Synchronous Environment
- Synchronizing Threads
- Managing Threads in an Asynchronous Environment
- Working with Application Domains
- Lab: Creating Multithreaded Applications and Application Domains

After completing this module, students will be able to:

- Manage threads in a synchronous environment by using the Thread and ThreadPool classes.
- Synchronize threads by using the Monitor, Mutex, ReaderWriterLock, Semaphore, EventWaitHandle, RegisteredWaitHandle, and Interlocked classes.
- Manage threads in an asynchronous environment by using asynchronous, execution context, SynchronizationContext, and thread exception types.
- Work with application domains by using the AppDomainSetup and AppDomain classes.