50166 Syllabus

Tuesday, March 23, 2010
11:10 AM

Course Syllabus

Programming the .NET Framework 3.5

Course#: 50166
Number of Days: 5
Format: Instructor-Led
Certification **Exams: None**

This five-day instructor-led course provides students with the knowledge and skills to develop applications with the .NET Framework 3.5.
Developing applications for the .NET Framework 3.5 requires familiarity with fundamental mechanisms such as Garbage Collection, Serialization, Streams and Application Domains; integrating these applications into solutions written using other programming languages requires Interoperability; taking advantage of the latest hardware requires Multithreading and Asynchronous Programming. These skills are also required for taking advantage of application frameworks on top of the core C1.R such as Windows Communication Foundation (WCF), Language Integrated Query (UNQ) and others.
The course is packed with practical code samples, demos and exercises to facilitate understanding the covered features from a .NET perspective, as well as from a more holistic system-oriented point of view.

At Course Completion
After completing this course, students will be able to:

- Develop applications that correctly interact with the .NET Garbage Collector;
- Use streams to read and write various data sources including files and memory buffers;
- Serialize and deserialize object data to different formats;
- Leverage hardware advances by using threads, thread pools, background workers and the Asynchronous Programming Model (APM);
- Isolate applications and plug-ins into separate Application Domains;
- Integrate .NET programs into mixed-language applications;
- Describe other application frameworks (such as WCF and UNQ) on top of the core CLR.

Module 1: Introduction

This module explains how to choose the appropriate version of the .NET Framework and the Common Language Runtime (CLR), how to property interact with the various parts of the CLR, how to use Visual Studio to target different versions of the .NET Framework and how to prepare for the future advances of the .NET Framework and the CLR.

After completing this module, students will be able to:

- Choose the appropriate version of the .NET framework;
- Use the Visual Studio multi-targeting support to compile applications for a specific version of the .NET framework.

Lessons

- Overview of the NET Framework
- Overview of the NET Type System
- Visual Studio and .NET Framework Versions
- .NET Backwards and Forwards Compatibility
- Visual Studio Multi-Targeting Support
- Future Roadmap

Module 2: Memory Management

This module explains how to programmatically interact with the managed Garbage Collector to achieve correctness in performance in .NET applications, how to use finalization and resource disposal for reclamation of unmanaged resources, and how to use advanced features such as weak references for fine-grained control over memory allocation.

Lessons
- Overview of Memory Management
- Garbage Collection First Steps
- GC Flavors
- Generations
- Interacting with the GC
- Weak References
- Finalization and the Dispose Pattern

Lab 1: Weak Timer
- Implementing a weak timer which does not keep a strong reference to the invoked delegate (and its target).

After completing this module, students will be able to:
- Programmatically interact with and control the .NET garbage collector;
- Design and implement applications which take advantage of advanced GC features;
- Provide proper finalization support for unmanaged resources;
- Use weak references to control memory allocation and reclamation.

Module 3: Streams and File I/O

This module explains how to read and write data from various data sources using the streams abstraction, and how to interact with file system objects (files and directories) using the System.IO classes.

Lessons
- Streams as a Data Abstraction
- File Streams
- Stream Readers / Writers
- The File and Directory Classes

Lab 2: Word Count
- Implementing an application which counts words in a set of files.

After completing this module, students will be able to:
- Take advantage of the streams abstraction to develop data-source-agnostic applications;
- Use file streams to write and read files on the lowest level of abstraction;
- Interact with text files and binary files using specialized readers and writers;
- Use the System.IO namespace classes to program file system objects (files and directories).

Module 4: Serialization

This module explains how to serialize and deserialize data objects into a variety of formats and how to design applications which take advantage of serialization and deserialization.

Lessons
- Motivation for Serialization
- Marking a Type for Serialization
- BinaryFormatter
- Controlling Serialization
- Custom Serialization
- Overview of XML Serialization
- Overview of DataContract Serialization

Lab 3: Serialization Framework
- Developing a custom serialization framework with support for compression and encryption of class fields.

After completing this module, students will be able to:
- Use .NET runtime serialization to serialize and deserialize data objects to various streams;
- Choose the appropriate serialization mechanism for the task at hand;
- Control serialization and customize it completely if necessary.

Module 5: Threading and Asynchronous Programming

After completing this module, students will be able to:
- Improve the responsiveness of UI applications by using the .NET Asynchronous Programming Model;

- Queue work items to the .NET thread pool;
- Create threads and interact with them using the Thread class;
- Synchronize access to shared data in a variety of forms.

This module explains how to take advantage of latest advances in hardware to implement multi-threaded and asynchronous applications with better responsiveness, latency, throughput and performance.

Lessons
- Taxonomy of Multi-Threading
- The Asynchronous Programming Model (APM)
- Thread Pool
- Manual Threading
- Synchronization
- Overview of Parallel Extensions for NET

Lab 4: Picture Feed
- Implementing APM at the WinForms-Ul level for a picture feed application

Lab 5: Queuing Work
- Implementing a thread-safe queue for work items and data
- Designing and implementing a producer-consumer scenario using manual threading and thread-safe input/output queues

Module 6: Application Domains

Designing and implementing a plug-in framework for isolating application add-ins (plug-ins) using Application Domains

After completing this module, students will be able to:
- Isolate applications and part of an application using Application Domains;
- Transfer serializable and marshal-by-reference objects across the AppDornain boundary;
- Create and unload AppDomains;
- Execute arbitrary code within an AppDomain.

This module explains how to isolate applications and plug-ins within an application using the Application Domain isolation boundary.

Lessons
- Application Domains as Isolation Boundaries
- Creating and Unloading AppDomains
- Executing Code In an AppDomain
- AppDomain Boundaries
- Overview of .NET Remoting

Lab 6: Plug-in Framework

Module 7: Interoperability

This module explains how to integrate managed applications into mixed applications using various unmanaged languages.

Lessons
- Platform Invoke
- COM Interoperability
- C++/CLI

Lab 7: P/Invoke and Reverse P/Invoke
- Using P/Invoke and reverse P/Invoke for interacting with Win32 APIs

Lab 8: COM Interoperability (Optional)
- Using CCWs and RCWs to explore two-way interaction between managed code and COM clients/objects

Lab 9: C++/CLI Native to Managed
- Wrapping a managed class so that it can be consumed from C++ code

Lab 10: C++/CLI Managed to Native
- Wrapping a C++ class and Win32 APIS so that they can be accessed from managed code using an object-oriented abstraction

After completing this module, students will be able to:

- Choose the appropriate form of interoperability for the task at hand;
- Interact with Win32 APIS and C-style DLLs;
- Use COM objects from managed code;
- Expose managed objects to COM clients;
- Design and implementing complex interoperability solutions using C++/CLI and its marshaling framework.

Module 8: Advanced Topics

This module explains how to improve application startup performance, how to bind to delegates and events dynamically at runtime, how to use advanced generics features, how to implement object cloning using serialization and how to diagnose complex assembly binding problems.

Lessons

- Improving Startup Performance with NGEN
- Advanced Delegates and Events
- Advanced Generics
- Object Cloning as Serialization
- Assembly Loading Problems and Contexts

After completing this module, students will be able to:

- Improve application startup performance using NGEN;
- Bind to events and delegates dynamically at runtime;
- Understand how the implementation of generics improves .NET application performance;
- Implement object cloning using serialization;
- Diagnose complex assembly loading problems and understand assembly loading (binding) contexts.

Module 9: Overviews

This module explains how to prepare for existing and future application frameworks on top of the core CLR.

Lessons

- ADO.NET
- System.Transactions
- Windows Communication Foundation (WCF)
- Windows Workflow Foundation (WF)
- Language Integrated Query (LINQ)
- Related Courses

After completing this module, students will be able to:

- Appreciate other application frameworks on top of the core CLR;
- Determine the appropriate learning plan from related courses.