

This four-day course is designed to provide a sound introduction to the .NET Framework for programmers who already know the C# language and the fundamentals of Windows Forms. It is current to .NET 4.0 and Visual Studio 2010. The course focuses on core portions of the .NET Framework that are common across many application areas. The course starts with an introduction to the architecture and key concepts of .NET. Then, it covers various aspects of the .NET programming model such as metadata, memory management, and .NET remoting; it moves into .NET security, .NET interoperability, and it concludes with thorough coverage of the .NET Framework diagnostic facilities. The goal is to equip you to begin building significant applications using the .NET Framework.

**Course Objectives:**

- Gain a thorough understanding of the philosophy and architecture of .NET.
- Acquire a working knowledge of the .NET programming model and .NET Security.
- Learn how to implement database applications using ADO.NET and LINQ.
- Learn how to debug .NET applications using .NET diagnostic classes and tools.

**Audience:** Experienced application developers or architects.

**Prerequisites:** A working knowledge of C#, including building simple GUIs with Windows Forms.

**Number of Days:** 4 days

**1. .NET Fundamentals**

What Is Microsoft .NET?  
 Open Standards and Interoperability  
 Windows Development Problems  
 Common Language Runtime  
 Attribute-Based Programming  
 Metadata  
 Types  
 NET Framework Class Library  
 Interface-Based Programming  
 Everything is an Object  
 Common Type System  
 ILDASM  
 .NET Framework SDK Tools  
 Language Interoperability  
 Managed Code  
 Assemblies  
 Assembly Deployment  
 JIT Compilation  
 ASP.NET and Web Services  
 The Role of XML  
 Performance

**2. Class Libraries**

Objects and Components  
 Limitation of COM Components  
 Components in .NET  
 Class Libraries at the Command Line  
 Monolithic versus Component  
 Class Libraries Using Visual Studio  
 References in Visual Studio  
 References at Compile Time and Run Time  
 Project Dependencies  
 Specifying Version Numbers

**3. Assemblies, Deployment and Configuration**

Assemblies  
 Customer Management System  
 ILDASM  
 Assembly Manifest  
 Assembly Dependency Metadata  
 Assembly Metadata

- Versioning an Assembly
- AssemblyVersion Attribute
- Strong Names
- Digital Signatures
- Verification with Digital Signatures
- Hash Codes
- Digitally Signing an Assembly
- Digital Signing Flowchart
- Signing the Customer Assembly
- Signed Assembly Metadata
- Private Assembly Deployment
- Assembly Cache
- Deploying a Shared Assembly
- Versioning Shared Components
- How the CLR Locates Assemblies
- Resolving an Assembly Reference
- Version Policy in a Configuration File
- Finding the Assembly
- Application Settings
- Application Settings Using Visual Studio
- Application Configuration File
- User Configuration File
- 4. Metadata and Reflection**
- Metadata
- Reflection
- System.Reflection.Assembly
- System.Type
- System.Reflection.MethodInfo
- Dynamic Invocation
- Late Binding
- 5. I/O and Serialization**
- Input and Output in .NET
- Directories
- Files and Streams
- “Read” Command
- Code for “Write” Command
- Serialization
- Attributes
- 6. .NET Programming Model**
- Garbage Collection
- Finalize Method
- C# Destructor Notation
- Dispose
- Finalize/Dispose Test Program
- Garbage Collection Performance
- Generations
- Processes
- Threads
- .NET Threading Model
- Race Conditions
- Thread Synchronization
- Monitor
- Synchronization of Collections
- Asynchronous Calls
- Asynchronous Delegates
- Using a Callback Method
- BackgroundWorker
- Application Isolation
- Application Domain
- Application Domains and Assemblies
- AppDomain
- CreateDomain
- App Domain Events
- Distributed Programming in .NET
- Windows Communication Foundation
- .NET Remoting Architecture
- Remote Objects and Mobile Objects
- Object Activation and Lifetime
- Singleton and SingleCall
- 7. .NET Security**
- Fundamental Problem of Security
- Authorization
- Authentication
- The Internet and .NET Security
- Code Access Security
- Role-Based Security
- .NET Security Concepts
- Permissions
- IPermission Interface
- IPermission Demand Method
- IPermission Inheritance Hierarchy
- Stack Walking
- Assert
- Demand
- Other CAS Methods

Security Policy Simplification  
 Simple Sandboxing API  
 Setting Up Permissions  
 Creating the Sandbox  
 Role-Based Security in .NET  
 Identity Objects  
 Principal Objects  
 Windows Principal Information  
 Custom Identity and Principal  
 BasicIdentity.cs  
 BasicSecurity.cs  
 Users.cs  
 Roles.cs  
 RoleDemo.cs  
 PrincipalPermission

**8. Interoperating with COM and Win32**

Interoperating Between Managed and  
 Unmanaged Code  
 COM Interop and PInvoke  
 Calling COM Components from  
 Managed Code  
 The TlbImp.exe Utility  
 TlbImp Syntax  
 Using TlbImp  
 Register the COM Server  
 OLE/COM Object Viewer  
 Run the COM Client  
 Implement the .NET Client Program  
 Import a Type Library Using Visual  
 Studio  
 Platform Invocation Services (PInvoke)  
 Marshalling out Parameters  
 Translating Types

**9. ADO.NET and LINQ**

ADO.NET  
 ADO.NET Architecture  
 .NET Data Providers  
 ADO.NET Interfaces  
 .NET Namespaces  
 Connected Data Access  
 AcmePub Database  
 Creating a Connection  
 Using Server Explorer  
 Performing Queries  
 Connecting to a Database  
 Database Code

Using Commands  
 Creating a Command Object  
 Using a Data Reader  
 Generic Collections  
 Executing Commands  
 Parameterized Queries  
 DataSet  
 DataSet Architecture  
 Why DataSet?  
 DataSet Components  
 DataAdapter  
 Data Access Class  
 Retrieving the Data  
 Filling a DataSet  
 Accessing a DataSet  
 Using a Standalone Data Table  
 Adding a New Row  
 Searching and Updating a Row  
 Deleting a Row  
 Row Versions  
 Row State  
 Iterating Through DataRows  
 Command Builders  
 Updating a Database  
 Data Binding  
 DataGridView Control  
 Language Integrated Query  
 (LINQ)  
 Bridging Objects and Data  
 Object Relational Designer  
 IntelliSense  
 Basic LINQ Query Operators  
 Obtaining a Data Source  
 Filtering  
 Ordering  
 Aggregation  
 Obtaining Lists and Arrays  
 Deferred Execution  
 Modifying a Data Source  
 Performing Inserts via LINQ to  
 SQL  
 Performing Deletes via LINQ to  
 SQL  
 Performing Updates via LINQ to  
 SQL

**10. Debugging Fundamentals**

- Compile-Time Errors
- Runtime Errors
- Debugging
- Project Configurations
- Release Configuration
- Creating a New Configuration
- Build Settings for a Configuration
- Customizing a Toolbar
- Using the Visual Studio Debugger
- Overflow Exception
- Just-in-Time Debugging
- Standard Debugging – Breakpoints
- Standard Debugging – Watch Variables
- Stepping with the Debugger
- The Call Stack
- JIT Debugging in Windows Apps
- Configuration File

## 11. **Tracing**

- Instrumenting an Application
- Order Application
- Debugging Review
- Tracing
- Debug and Trace Classes
- Viewing Trace Output
- Debug Statements
- Debug Output
- Assert
- More Debug Output
- WriteLine Syntax
- Event Logs
- Viewing Event Logs
- Event Log Entry Types
- .NET EventLog Component
- Retrieving Entries from an Event Log
- Handling EventLog Events

## 12. **More about Tracing**

- Trace Switches
- BooleanSwitch
- Using a Configuration File
- TraceSwitch
- SwitchDemo
- Trace Listeners
- DefaultTraceListener
- A Stream Listener
- A Custom Listener
- Trace Output to a Window

- An Event Log Listener
- Tracing in the Order Application
- Trace Output