

This thorough and comprehensive five-day course is a practical introduction to programming in C#, utilizing the services provided by .NET. This course emphasizes the C# language. It is current to Visual Studio 2010 and .NET 4.0, which introduces important new features such as dynamic data type, named and optional arguments, and the use of variance in generic interfaces. The new features are covered in a new chapter. A new supplement covers the fundamentals of Language Integrated Query (LINQ), which was introduced with .NET 3.5. This course is intended to be fully accessible to programmers who do not already have a strong background in object-oriented programming in C-like languages, such as C++ or Java. It is ideal, for example, for Visual Basic 6 or COBOL programmers who desire to learn C#. An important thrust of the course is to teach C# programming from an object-oriented perspective. It is often difficult for programmers trained originally in a procedural language to start “thinking in objects.” This course introduces object-oriented concepts early, and C# is developed in a way that leverages its object orientation. The course explores several important interactions between C# and the .NET Framework, and it includes an introduction to major classes for collections, delegates, and events. It includes a succinct introduction to creating GUI programs using Windows Forms. The course concludes with a chapter covering the new features in C# 4.0.

**Course Objectives:**

- Acquire a working knowledge of C# programming
- Learn how to implement programs using C# and classes from the .NET Framework
- Learn how to implement simple GUI programs using Windows Forms
- Gain a working knowledge of dynamic data type, named and optional arguments, and other new features in C# 4.0.

**Audience:** Programmers who need to design and develop C# for the .NET framework.

**Prerequisites:** The student should have programming experience in a high-level language.

**Number of Days:** 5 days

- |  |   |
|--|---|
| <p><b>1. .NET: What You Need to Know</b><br/>         Getting Started<br/>         .NET: What is Really Happening<br/>         .NET Programming in a Nutshell<br/>         Viewing the Assembly<br/>         Viewing Intermediate Language<br/>         Understanding .NET<br/>         Visual Studio 2010<br/>         Creating a Console Application<br/>         Adding a C# file<br/>         Using the Visual Studio Text Editor<br/>         IntelliSense<br/>         Build and Run the Project<br/>         Pausing the Output</p> | <p><b>2. First C# Programs</b><br/>         Hello, World<br/>         Compiling, Running (Command Line)<br/>         Program Structure<br/>         Namespaces<br/>         Variables<br/>         Expressions<br/>         Assignment<br/>         Calculations Using C#<br/>         More About Output in C#<br/>         Input in C#</p> |
|--|---|

More About Classes  
 InputWrapper Class  
 Echo Program  
 Using InputWrapper  
 Compiling Multiple Files  
 Multiple Files in Visual Studio  
 The .NET Framework

**3. Data Types in C#**

Strong Typing  
 Typing in C#  
 Typing in C++  
 Typing in Visual Basic 6  
 C# Types  
 Integer Types  
 Integer Type Range  
 Integer Literals  
 Floating Point Types  
 Floating Point Literals  
 IEEE Standard for Floating Point  
 Decimal Type  
 Decimal Literals  
 Character Type  
 Character Literals  
 string  
 Escape Characters  
 Boolean Type  
 Implicit Conversions  
 Explicit Conversions  
 Nullable Types

**4. Operators and Expressions**

Operator Cardinality  
 Arithmetic Operators  
 Multiplication  
 Division  
 Additive Operators  
 Increment and Decrement  
 Relational Operators  
 Conditional Logical Operators  
 Short-Circuit Evaluation  
 Ternary Conditional Operators  
 Bitwise Operators  
 Bitwise Logical Operators  
 Bitwise Shift Operators  
 Assignment Operators  
 Expressions  
 Precedence

Associativity  
 Checking

**5. Control Structures**

If Test  
 Blocks  
 Loops  
 while Loop  
 do while Loops  
 for Loops  
 Arrays  
 foreach Loop  
 break  
 continue  
 goto  
 Structure Programming  
 Multiple Methods  
 switch  
 switch in C# and C/C++

**6. Object-Oriented Programming**

Objects  
 Objects in the Real World  
 Object Models  
 Reusable Software Components  
 Objects in Software  
 State and Behavior  
 Abstraction  
 Encapsulation  
 Classes  
 Inheritance Concepts  
 Relationships among Classes  
 Polymorphism  
 Object-Oriented Analysis and Design  
 Use Cases  
 CRC Cards and UML

**7. Classes**

Classes as Structure Data  
 Classes and Objects  
 References  
 Instantiating and Using an Object  
 Assigning Object References  
 Garbage Collection  
 Methods  
 Public and Private  
 Abstraction  
 Encapsulation

- Initialization
- Initialization with Constructors
- Default Constructor
- this
- Static Fields and Methods
- Static Methods
- Static Constructor
- Constant and Readonly Fields
- 8. More about Types**
- Overview of Types in C#
- Structures
- Uninitialized Variables
- Coping a Structure
- Hotel.cs
- HotelCopy.cs
- Classes and Structs
- Enumeration Types
- Reference Types
- Class Types
- object
- string
- Arrays
- Default Values
- Boxing and Unboxing
- Implicitly Types Variables
- 9. Methods, Properties, and Operators**
- Static and Instance Methods
- Method Parameters
- No “Freestanding” Functions in C#
- Classes with All Static Methods
- Parameter Passing
- Parameter Terminology
- Value Parameters
- Reference Parameters
- Output Parameters
- Structure Parameters
- Class parameters
- Method Overloading
- Modifiers as Part of the Signature
- Variable Length Parameter Lists
- Properties
- Auto-Implemented Properties
- Operator Overloading
- Operator Overloading in the Class
- Library
- 10. Characters and Strings**
- Characters
- Character Codes
- ASCII and Unicode
- Escape Sequences
- Strings
- String Class
- String Literals and Initialization
- Concatenation
- Index
- Relational Operators
- String Equality
- String Comparisons
- String Input
- String Methods and Properties
- StringBuilder Class
- StringBuilder Equality
- Command Line Arguments
- Command Line Arguments in the IDE
- Command Loops
- Splitting a String
- 11. Arrays and Indexers**
- Arrays
- One Dimensional Arrays
- System.Array
- Random Number Generation
- Next Methods
- Jagged Arrays
- Rectangular Arrays
- Arrays as Collections
- Bank Case Study: Step 1
- Account Class
- Bank .Class
- TestBank Class
- Atm Class
- Running the Case Study
- Indexers
- Using the Indexer
- 12. Inheritance**
- Inheritance Fundamentals
- Inheritance in C#
- Root Class – object
- Access Control
- Public Class Accessibility
- Internal Class Accessibility
- Member Accessibility

- Member Accessibility Qualifiers
- Method Hiding
- Method Hiding and Overriding
- Initialization
- Initialization Fundamentals
- Default Constructor
- Overloaded Constructors
- Invoking Base Class Constructors
- Bank Case Study: Step 2
- Bank Case Study Analysis
- Account
- CheckingAccount
- SavingsAccount
- TestAccount
- Running the Case Study
- 13. Virtual Methods and Polymorphism**
- Introduction to Polymorphism
- Abstract and Sealed Classes
- Virtual Methods and Dynamic Binding
- Type Conversions in Inheritance
- Converting Down the Hierarchy
- Converting Up the Hierarchy
- Virtual Methods
- Virtual Method Cost
- Method overriding
- The Fragile Base Class Problem
- override Keyword
- Polymorphism
- Polymorphism Using “Type Tags”
- Polymorphism Using Virtual
- Abstract Classes
- Sealed Classes
- Heterogeneous Collections
- Bank Case Study: Step 3
- Case Study Classes
- Run the Case Study
- Account
- CheckingAccount, SavingsAccount
- Bank and Atm
- TestBank
- 14. Formatting and Conversion**
- Introduction to Formatting
- ToString
- ToString in Your Own Class
- Using Placeholders
- Format String
- Controlling Width
- Format String
- Currency
- String.Format
- PadLeft and PadRight
- Bank Case Study: Step 4
- Type Conversions
- Conversion of Built-In Types
- 15. Exceptions**
- Introduction to Exceptions
- Exception Fundamentals
- .NET Exception Handling
- Exception Flow of Control
- Context and Stack Unwinding
- System.Exception
- User-Defined Exception Classes
- Structure Exception Handling
- Family Block
- Bank Case Study: Step 5
- Inner Exceptions
- Checked Integer Arithmetic
- 16. Interfaces**
- Interfaces in C#
- Interface Inheritance
- Programming with Interfaces
- Implementing Interfaces
- Using an Interface
- Dynamic Use of Interfaces
- is Operator
- as Operator
- Bank Case Study: Step 6
- Common Interfaces in Case Study – IAccount
- Apparent Redundancy
- IStatement
- IStatement Methods
- IChecking
- ISavings
- The Implementation
- SavingsAccount
- The Client
- Resolving Ambiguity
- Access Modifier
- 17. .NET Interfaces and Collections**
- Collections

Count and Capacity  
 foreach Loop  
 Array Notation  
 Adding to the List  
 Remove Method  
 RemoveAt Method  
 Collection Interfaces  
 IEnumerable and IEnumerator  
 ICollection  
 IList  
 A Collection of User-Defined Objects  
 Duplicate Objects  
 A Correction to AccountList (Step 1)  
 Bank Case Study: Step 7  
 Copy Semantics and ICloneable  
 Copy Semantics in C#  
 Shallow Copy and Deep Copy  
 Reference Copy  
 Memberwise Clone  
 Using ICloneable  
 Comparing Objects  
 Sorting an Array  
 Anatomy of Array.Sort  
 Using the is Operator  
 The Use of Dynamic Type Checking  
 Implementing IComparable  
 Running the Program  
 Complete Solution  
 Writing Generic Code  
 Using a Class of object  
 Generic Types  
 Generic Syntax in C#  
 Generic Client Code  
 System.Collections.Generic  
 Object Initializers  
 Collection Initializers  
 Anonymous Types

**18. Delegates and Events**

Overview of Delegates and Events  
 Callbacks and Delegates  
 Usage of Delegates  
 Declaring a Delegate  
 Defining a Method  
 Creating a Delegate Object  
 Calling a Delegate  
 A Random Array

Anonymous Methods  
 Combining Delegate Objects  
 Account.cs  
 DelegateAccount.cs  
 Lambda Expressions  
 Named Method  
 Events  
 Events in C# and .NET

**19. Introduction to Windows Forms**

Creating a Windows Forms App  
 Partial Classes  
 Windows Forms Event Handling  
 Add Events for a Control  
 Events Documentation  
 Closing a Form  
 ListBox Control

**20. New Features in C#**

dynamic Type  
 dynamic versus object  
 Behavior of object  
 Behavior of dynamic  
 Names Arguments  
 Optional Arguments  
 Book Class  
 Using Optional Arguments  
 Automating Office with C# 4.0  
 Automating Excel  
 Automating Word  
 Variance in Generic Interfaces  
 Variance with IComparer<T>  
 Interfaces with Variance Support