4101. Object-Oriented Programming in C#

Course ID #: 1100-301-00-W

Hours: 35

Course Content

Course Description:

Microsoft's .NET is a revolutionary advance in programming technology that greatly simplifies application development and is a good match for the emerging paradigm of Web-based services, as opposed to proprietary applications. Part of this technology is the new language from Microsoft, C#. This language combines the power of C++ and the ease of development of Visual Basic 6. It bears a striking resemblance to Java and improves on that language. C# has become the dominant language for building new applications on Microsoft platforms.

This thorough and comprehensive 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 2017, .NET Framework 4.7 and C# 7.0. Important newer features such as dynamic data type, named and optional arguments, the use of variance in generic interfaces, asynchronous programming keywords, and tuples are covered in a final chapter. A supplement covers the fundamentals of Language Integrated Query (LINQ).

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 procedural 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. A case study is used to illustrate creating a complete system using C# and .NET. Besides supporting traditional object-oriented features, such as classes, inheritance, and polymorphism, C# introduces several additional features, such as properties, indexers, delegates, events, and interfaces that make C# a compelling language for developing object-oriented and component-based systems. This course provides thorough coverage of all these features.

C# as a language is elegant and powerful. But to utilize its capabilities fully, you need to have a good understanding of how it works with the .NET Framework. 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 newer features in the language through C# 7.0.

4101. Object-Oriented Programming in C#

Course ID #: 1100-301-00-W

Hours: 35

Numerous programming examples and exercises are provided, including the case study. The student will receive a comprehensive set of materials, including course notes and all the programming examples.

The course includes four electronic supplements, provided as PDF files. They cover Visual Studio 2017, Language Integrated Query (LINQ), unsafe code and the C# pointer type, and .NET 4.7.

At Course Completion:

This course teaches you how to:

- 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.
- Learn how to do asynchronous programming using new keywords in C# 5.0.
- Become aware of new features in C# 6.0 and C# 7.0.

Prerequisites:

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

Deliver Method:

This course is delivered through a mix of instructor-led training (ILT) and hands-on labs.

Topics:

NET: What You Need To Know

- .NET Executables and the CLR
- A .NET Testbed for C# Programming
- Using Visual Studio 2017

First C# Programs

- Hello, World
- Namespaces
- Variables and Expressions
- Using C# as a Calculator
- Input/Output in C#
- .NET Framework Class Library

Data Types in C#

- Data Types
- Integer Types
- Floating Point Types
- Decimal Type
- Characters and Strings
- Boolean Type
- Conversions
- Nullable Types

4101. Object-Oriented Programming in C#

Course ID #: 1100-301-00-W

Hours: 35

Operators and Expressions
- Operator Cardinality
- Arithmetic Operators
- Relational Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators
- Expressions
- Checked and Unchecked

Control Structures
- If Tests
- Loops
- Arrays
- Foreach
- More about Control Flow
- Switch

Object-Oriented Programming
- Objects
- Classes
- Inheritance
- Polymorphism
- Object-Oriented Languages
- Components

Classes
- Classes as Structured Data
- Methods
- Constructors and Initialization
- Static Fields and Methods
- Constant and Readonly

More about Types
- Overview of Types in C#
- Value Types
- Boxing and Unboxing
- Reference Types
- Implicitly Typed Variables

Methods, Properties and Operators
- Methods
- Parameter Passing
- Method Overloading
- Variable-Length Parameter Lists
- Properties
- Auto-Implemented Properties
- Operator Overloading

Characters and Strings
- Characters
- Strings
- String Input
- String Methods
- StringBuilder Class
- Programming with Strings

Arrays and Indexers
- Arrays
- System.Array
- Random Number Generation
- Jagged Arrays
- Rectangular Arrays
- Arrays as Collections
- Bank Case Study—Step 1
- Indexers

Inheritance
- Single Inheritance
- Access Control
- Method Hiding
- Initialization
- Bank Case Study—Step 2

4101. Object-Oriented Programming in C#

Course ID #: 1100-301-00-W

Hours: 35

Virtual Methods and Polymorphism
- Virtual Methods and Dynamic Binding
- Method Overriding
- Fragile Base Class Problem
- Polymorphism
- Abstract Classes
- Sealed Classes
- Heterogeneous Collections
- Bank Case Study—Step 3

Formatting and Conversion
- ToString
- Format Strings
- String Formatting Methods
- Bank Case Study—Step 4
- Type Conversions

Exceptions
- Exception Fundamentals
- Structured Exception Handling
- User-Defined Exception Classes
- Inner Exceptions
- Bank Case Study—Step 5

Interfaces
- Interface Fundamentals
- Programming with Interfaces
- Using Interfaces at Runtime
- Bank Case Study—Step 6
- Resolving Ambiguities

.NET Interfaces and Collections
- Collections
- Bank Case Study—Step 7
- IEnumerable and IEnumerator
- Copy Semantics and ICloneable
- Comparing Objects
- Generic Types
- Type-Safe Collections
- Object Initializers
- Collection Initializers
- Anonymous Types
- Bank Case Study—Step 8

Delegates and Events
- Delegates
- Anonymous Methods
- Lambda Expressions
- Events

Introduction to Windows Forms
- Creating Windows Applications Using Visual Studio 2017
- Partial Classes
- Buttons, Labels and Textboxes
- Handling Events
- Listbox Controls

Newer Features in C#
- Dynamic Data Type
- Named Arguments
- Optional Arguments
- Variance in Generic Interfaces
- Asynchronous Programming Keywords
- New Features in C# 6.0 and C# 7.0

Appendix A. Learning Resources

Electronic File Supplements

Supplement 1. Using Visual Studio 2017
- Signing in to Visual Studio
- Overview of Visual Studio 2017
- Creating a Console Application
- Project Configurations
- Debugging
- Multiple-Project Solutions
-

4101. Object-Oriented Programming in C#

Course ID #: 1100-301-00-W

Hours: 35

Supplement 2. Language Integrated Query (LINQ)

- What Is LINQ?
- Basic Query Operators
- Filtering
- Ordering
- Aggregation

Supplement 3. Unsafe Code and Pointers in C#

- Unsafe Code
- C# Pointer Type
-

Supplement 4. Using .NET Framework 4.7

- Installing .NET Framework 4.7
- DirectX Dependency
- New Features in .NET 4.7