Object-Oriented Programming with C#

Description: This course introduces the student to writing object-oriented programs in C#.

Prerequisites: Prior study in object-orientation and UML is helpful.

Audience: Business analysts, developers, managers and other people interested in learning how to program C#

Length: Five days.

Objectives:

After taking this course, you will be able to:

1. Write C# classes using object-oriented techniques such as encapsulation, inheritance and polymorphism.
2. Write C# programs using basic syntax elements for looping and flow of control.
3. Write C# programs that define and manipulate standard .NET data types.
4. Write C# programs that create and manipulate arrays and collections.
5. Write C# classes with constructors, overloaded methods, properties and static members.
6. Write C# programs that catch and throw exceptions.
7. Write C# programs that define and implement interfaces.
8. Write C# programs that process XML.
9. Understand the basics of ASP.NET.

Topic List

1. Introduction to the Course
   - Object-Oriented Programming with C#
   - Legal Information
   - Object-Oriented Programming with C#
   - Introductions
   - Course Description
   - Course Objectives
   - Sample Agenda
   - Sample Agenda, cont'd

2. Introduction to C#
   - Introduction to C#
   - What is C#?
   - C# History
   - C# Language Tree
   - C# Design Goals
   - The .NET Framework
   - .NET Technologies
   - ECMA Certification
   - The Mono Project
   - C# Development Process
   - Microsoft Intermediary Language
   - Hello, World
   - System.Console.WriteLine
   - Chapter Summary

3. Data Types and Assignment
   - Data Types and Assignment
   - C# Program Structure
   - C# Comments
   - C# Statements
   - Variables
   - Rules for Identifiers
   - Reserved Words
   - C# Data Types
   - Integer Types
   - Floating Point Types
   - Character Type
   - Character Escape Codes
   - The bool Type
   - The String Type
   - The decimal Type
   - Assignment
   - Cast Operator
   - Scope of Variables

6. Introduction to Object Orientation
   - Introduction to Object Orientation
   - What is Object-Oriented Programming?
   - Object-Oriented Languages
   - Goals of Object Orientation
   - The Three Pillars
   - What is an Object?
   - Why are Objects Useful?
   - Objects Respond to Messages
   - Objects and Classes
   - A Class is Like a Cookie Cutter
   - Writing a Class
   - Sample C# Class
   - Instantiating Objects
   - Methods Operate on Objects
   - The Stack and the Heap
   - Comparing References
   - Comparing References, cont'd
   - Garbage Collection
   - Introduction to Inheritance
   - Introduction to Polymorphism
   - Synonym Alert!
   - Introduction to Namespaces
   - Encapsulation
   - Using Encapsulation
   - Encapsulation Summary
   - Properties
   - Properties, cont'd
   - Introduction to Interfaces
   - Quick Practice
   - Chapter Summary

7. Methods
   - Methods
   - What is a Method?
   - Class Structure
   - Method Syntax
   - Instance Method Examples
   - Calling Instance Methods
   - Calling Methods, cont'd
   - Quick Practice
   - Overloading Methods
   - Calling Overloading Methods
   - Constructor Methods
   - Overloaded Constructors
   - Calling Constructors
   - Quick Practice
   - The No-Argument Constructor

- C# and Inheritance
- Derived Class Objects are a Superset
- Accessing the Base Class
- Building Class Hierarchies
- Quiz: What Kind of Relationship?
- Quiz: What Kind of Relationship?
- Quiz: What Kind of Relationship?
- Overriding Behaviors
- The Object Class
- Constructors and Superclasses
- Constructors and Superclasses. cont'd
- Polymorphism
- Polymorphic Reference Assignment
- Writing Polymorphic Algorithims
- Using Abstract Classes
- Review Questions
- Quick Practice
- Chapter Summary

11. Interfaces
  - Interfaces
  - What are Interfaces?
  - Why Use Interfaces?
  - Defining an Interface
  - Implementing an Interface
  - Interfaces versus Abstract Classes
  - Interface Reference Types
  - Example: Using the IComparable Interface
  - Using the IComparable Interface, cont'd
  - Using the IComparable Interface, cont'd
  - Using the IComparable Interface, cont'd
  - Using the IComparable Interface, cont'd
  - Implementing Multiple Interfaces
  - Explicit Interface Implementation
  - Explicit Interface Implementation, cont'd
  - Review Questions
  - Quick Practice
  - Chapter Summary

12. File I/O
  - File I/O
  - File I/O Fundamentals
  - Binary versus Text
  - What is a Stream?
  - What is a Reader/Writer?
  - Handling I/O Exceptions
  - Reading a Text File
  - Writing a Text File

- One Solution
- Introduction to XML Namespaces
- Namespace Syntax
- Content with Multiple Namespaces
- Defining a Default Namespace
- Well Formed Versus Valid
- Introducing XML Schema
- A Sample Schema
- Simple Types vs Complex Types
- Sample Built-in Simple Types
- Defining a New Simple Type
- Defining Complex Types
- Compositors
- Defining Elements
- Element Occurrence Constraints
- Defining Attributes
- Attribute Occurrence
- Referencing Global Elements
- Writing an XML Application
- XmlDocument vs XmlReader
- Introduction to XmlDocument
- DOM Interfaces
- Sample XML Content
- Objects in the DOM Tree for Sample
- XmlDocument Sample Program
- XmlDocument Sample Program, cont'd
- Introduction to XmlReader
- XmlReader Overvew
- XmlReader Sample Program
- XmlReader Sample Program, cont'd
- XmlReader Sample Program, cont'd
- Chapter Summary

15. Introduction to ASP.NET
  - Introduction to ASP.NET
  - HTTP and HTML
  - HTML Forms
  - HTML Forms, cont'd
  - Introduction to ASP.NET
  - ASP.NET History
  - ASP.NET Development
  - Simple ASP.NET Application
  - Simple ASP.NET Application, Design View
  - Simple ASP.NET Page
  - Simple ASP.NET Page, cont'd
  - Server Controls
  - Event Handling and Postbacks
  - Postbacks, cont'd

- Chapter Summary