



Course 451: Object-Oriented Programming with .NET

Course Description...

This course is an intermediate level course in the object-oriented features of .NET. It introduces encapsulation, inheritance (both implementation and interface) and polymorphism; describes how to declare and use events; and briefly introduces the concepts of components and assemblies.

Learning Objectives...

- Understand implementation and interface inheritance and when to use them
- Discover the power of loosely coupled events, how to declare them in classes and use them effectively in client side code
- Explore the significance of polymorphism and its role in supporting robust and well-formed classes
- Briefly introduce the concept of components (private and shared assemblies)
- Briefly introduce the .NET data access model

Who should attend...

Audience includes individuals who are familiar with simple programming constructs such as variable declaration, conditional branching and conditional looping.

Prerequisites...

Course 460 or its equivalent.

See next page for a detailed course outline...



Course Outline...

Introduction and Overview

Course Objectives

Unit 1: Review of Prerequisite Material (Optional)

What Is .NET?

- .NET Framework
- Microsoft Intermediate Language
- Common Language Runtime

Why Is .NET?

- Safer, more secure and more stable
- Run once, run always
- XCOPY installation

Introduction to Visual Studio .NET

- Tour of RAD Environment
- Tour of important files
- Tour of important project types

Expectations, You Know How To

- Write procedural code
- Build simple windows forms
- Use a debugger to step through code and watch variables

Unit 2: Introduction to Object-Oriented Programming

The Three Elements of an Object-Oriented Program

- Encapsulation
- Inheritance
- Polymorphism

The Reasons For Object-Oriented Programming

- Reusability
- Maintenance
- Clarity

Unit 3: Encapsulation

Principles

- Code and data together
- Data hidden behind code
- Interface and implementation as separate entities



Classes

- What is a class? (classes as blueprints, objects as homes)
- Building a class (fields, methods, property procedures)
- Static variables, methods and classes
- Property procedures
- Namespaces

Objects

- Declaring a class (reference types vs. value types, declaration syntax)
- The difference between classes and structures (boxing and unboxing)
- Potential for memory leaks with reference types
- The .NET Garbage Collector

Other Topics

- Constructors (including overloading)
- Visual Studio tools for debugging classes
- The Object Browser

Unit 4: Inheritance and Polymorphism

Implementation Inheritance

- Inheriting from a single parent
- Extending a parent
- Overriding a parent
- Calling a parent's constructor
- Abstract classes
- Abstract methods
- Visual inheritance
- Cross-language inheritance and debugging
- The Object Browser revisited

Interface Inheritance

- Inheriting interfaces from many parents
- Interfaces as contracts
- Distinguishing implementation and interface inheritance
- Examples

Polymorphism

- Polymorphism as a substitute for some conditional branches
- Polymorphism via both implementation and interface inheritance
- Polymorphism as the vehicle for robust, reusable classes
- Examples

The Collection Namespace

- Collection objects (hashtable, arraylist, stack, queue, etc)
- Debugging with objects and collections of objects
- Examples (using polymorphism as a key example)



Unit 5: Events

Loose Coupling of Events

- Binding an object's events to an event handler
- Defining events within a class
- Examples

The Standard Event Signature

- The role of "sender as object" in the standard event signature
- The role of "e as EventArgs" in the standard event signature
- Defining events within a class using the standard event signature
- Using the standard event signature in an event handler
- Examples

Delegates (briefly)

Unit 6: Other Topics

Structured Exception Handling

- Standard exceptions and the "try ... catch ... finally" syntax
- The exception object and its properties
- Throwing new exceptions
- Walking the call stack

Components

- Private assemblies
- Shared assemblies
- Legacy components (COM components and VB6 conversion)

Database Access

- File I/O
- Where we are coming from (ODBC and OLE DB)
- The need for the managed provider
- Comparing ADO with ADO.NET
- ADO and ADO.NET Connection and Command objects
- The evolution of the Recordset Object
- DataReader objects
- DataAdapter and Dataset objects
- Dataset objects and XML
- Dataset objects and the IEnumerable interface

Unit 7: Putting It All Together

What Has Been Learned

- Understand implementation and interface inheritance and when to use them
- Discover the power of loosely coupled events, how to declare them in classes and use them effectively in client side code
- Explore the significance of polymorphism and its role in supporting robust and well-formed classes
- Briefly introduce the concept of components (private and shared assemblies)
- Briefly introduce the .NET data access model