

## TT1260: Object-Oriented Design Patterns & Best Practices in .Net (5 days)

Geared for students with basic .Net development skills, **Object Oriented Design Patterns & Best Practices in .Net** is a five-day, comprehensive training course that explores the most common object-oriented design patterns (Gang of Four) and how to use these patterns to develop solid, robust and reusable software development applications. Students will also review essential OO programming concepts.

**COURSE SNAPSHOT**

**Duration:** 5 days  
**Skill Level:** Introductory  
**Focus:** .Net application development & design  
**Format:** 50% hands-on coding and group dynamics labs, combined with expert lecture, open discussions and detailed patterns walk-throughs.  
**Language / Tools:** This edition has a .Net focus, although we can easily present geared for Java, C++ or other programming languages  
**Delivery Format:** Available for onsite private classroom presentation, or live online / virtual presentation  
**Customizable:** Yes

► **Course Objectives: What You'll Learn**

Working in a hands-on environment, developers will explore key Creational, Structural and Behavior Design patterns and how they used most effectively in building robust, reusable applications. This course combines the use of hands-on coding labs with several "mini-projects" to be completed throughout the training to get the students using and reviewing the Patterns in a practical manner. The workshop also contains several "thinking and drawing" lab exercises as a component of the object oriented overview portion of the training course.

This comprehensive training course will begin with a review of core concepts of Object Oriented analysis & design using UML (approximately one day). Throughout the remainder of the course we will explore the following patterns, varying the levels of coverage to drill down on the most commonly used Patterns, and to simply survey others. Students will compare and contrast the patterns and explore the advantages and disadvantages of using certain patterns for explicit development functions in the .Net environment.

The course provides a solid foundation in basic terminology and concepts, extended and built upon throughout the engagement. Processes and best practices are discussed and illustrated through both discussions and group activities.

Attending students will be led through a series of advanced topics comprised of integrated lectures, group discussions and comprehensive demonstrations.

The following Patterns are addressed in this course, covered to the specified level below. This coverage level can be adjusted based on the requirements of your organization, and which patterns you use or would like to review.

Creational Patterns	
Pattern	Coverage
Abstract Factory	Full
Factory Method	Full
Singleton	Full
Builder	Brief
Prototype	Brief
Structural Patterns	
Pattern	Coverage
Composite	Full
Adapter	Full
Proxy	Full
Bridge	Brief
Façade	Brief
Decorator	Brief
Flyweight	Brief
Behavior Patterns	
Pattern	Coverage
Observer	Full
Strategy	Full
Iterator	Full
Visitor	Brief
Interpreter	Brief
Chain of Command	Brief
Mediator	Brief
State	Brief
Memento	Full
Assertion (non GoF)	Brief
Dispatcher (non GoF)	Brief

► **Course Overview & Structure**

This course consists of approximately 50% hands-on lab work (Patterns) and group dynamics exercises (for OOAD).

Throughout the course students will be led through a series of progressively advanced topics, where each topic consists of lecture and group discussion.

This class is “technology-centric”, designed to train attendees in essential OO background coupling the most current, effective techniques with the most effective practices.

#### ► Audience & Pre-requisites: Who Should Attend

This is a **basic** level OO training course, designed for developers who need to identify, design, and lead the implementation of OO projects. We will explore and apply the terminology, the specification, the processes and technologies specific to OO.

Attendees should be familiar with UML and have basic programming experience in C#. This course is not recommended for developers new to C# programming.

#### ► Related Courses – Suggested Learning Path

**Take Before:** Students should have basic development skills and experience in the following topics, or attend these courses as a pre-requisite:

- TT1100 Core Object Oriented Analysis & Design using UML 2.0

**Take Instead:** We offer other courses that provide different levels of knowledge or focus:

- If you do not require any review of OOAD concepts (as addressed in the agenda below), the **TT1200 Core Design Patterns & Frameworks** course may be more appropriate.
- For a more in-depth introduction to OOAD and UML, consider **TT1130 Applying Object-Oriented Analysis & Design Using UML** (also a candidate to take before class)
- Need a faster intro to OO / UML course? Consider **TT1100 Core OOAD with UML (3 days)**
- If you wish to emphasize design, consider **TT1310 Domain Analysis & Design using UML**
- Java/C++ developers should consider **TT1250 Object Oriented Design Patterns in Java and/or C++**

**Take After:** We offer a variety of introductory through advanced security, development, project management, engineering, architecture and design courses. Students may want to consider the following topics as a follow-on to this course.

- Advanced core programming classes for Java or .Net
- Secure programming or design courses
- Service-Oriented Analysis and Design
- Web Services – Intro through Advanced
- Architecture & Analysis courses
- Software Engineering, Design or Project Management tracks

Please note all development courses may also be offered in other programming languages or tailored to suit your unique requirements. Please contact us for recommended next steps tailored to your longer term education, project or development objectives.

#### ► Student Materials: What You’ll Receive

Our robust course materials include much more than a simple slideshow presentation handout. Trivera Technologies Student materials include a comprehensive hard-copy course manual, complete with detailed course notes, code samples, diagrams and current reference materials, all directly related to the course at hand, indexed for ease of use. Step-by-step lab instructions and project descriptions are clearly illustrated and commented for maximum learning and ease of use.

Our course kits are designed to serve as an excellent and useful reference set, long after we leave your classroom.

#### ► Experiential Learning: Hands-On Labs

This class is “technology-centric”, designed to train attendees in essential OOAD and UML development skills, coupling the most current, effective techniques with the soundest industry practices.

This workshop is about **50% dynamic hands-on and group lab exercises** and **50% lecture**. Throughout the course students will be led through a series of progressively advanced topics, where each topic consists of lecture, group discussion, comprehensive hands-on lab exercises, and lab review. Multiple detailed lab exercises are laced throughout the course, designed to reinforce fundamental skills and concepts learned in the lessons. At the end of each lesson, developers will be tested with a set of review questions to ensure that he/she has fully understands that topic.

#### ► Delivery Environment & Classroom Set Up

This course can be delivered using any C# programming configuration desired.

Our lab guides are complete with software-specific instructions, screen shots and detailed tutorials for using the software you select. In most cases we can easily port our classes to run in the environment of your choosing.

For course deliveries or virtual presentation using open-source tools, we’ll provide our unique **LoadNGo Instant Classroom Kit**, which enables students to run the entire course off of a DVD that hosts the entire course set up software, labs, and other pertinent useful educational resources, whitepapers and more. You only need to provide the hardware and appropriate O/S, and we’ll do the rest. No installation needed. **Great for secure environments.** Minimum set up burden for your team or firm, with maximum results for your students.

No matter which set up option or software your firm requires, we’re pleased to provide a detailed set up guide for all private or on-site courses, and as much assistance as you require to prepare your students or classroom for the course. Our support personnel and instructors can be contacted for any advice you may require to prepare your classroom and/or students for attendance.

**Workshop Topics Covered**

**Session: OO Pattern Background**

**Lesson: Object-Oriented Concepts**

- Modern OO Concepts: Abstraction
- Three Object-Oriented Themes
- Why Build Models?
- Notation
- Classes & Objects
- Object Services
- Rendering Objects
- Responsibilities and Operations
- Messages and Public Interfaces
- Instantiation
- Fields vs. Associations
- Methods and Algorithms
- The Three Pieces of PIE
- Encapsulation
- What Gets Inherited?
- Inheritance of Methods and Overriding
- Using the Overridden Method's Implementation
- Inheriting Associations
- Evaluating Inheritance
- Multiple Inheritance
- Polymorphism: Performance and Maintenance
- Abstract Classes
- Abstract Classes in C++ and Java; in C# and Visual Basic.NET
- Abstract Methods in C++ and Java; in C# and VB.NET
- Concrete Classes
- Interfaces
- Interfaces in C++ and Java
- Interfaces in C# and VB.NET
- Functionality Options
- Design-by-Interface
- C# and VB.Net as OO languages
- Pattern Implementation

**Lesson: UML and USDP**

- What is UML?
- Domains
- The Process of OO Analysis and Design
- OOAD Process: Requirements Capture
- OOAD Activities: Requirements Analysis
- Domain Design

- Object Discovery
- OOAD Process: Detailed Design
- Object Relationships & Interactions
- Object State & Object Activities
- OOAD Process: Architectural Design
- Packaging of Objects
- Components and Deployment
- Understanding our Understanding
- The Unified Software Development Process
- Granularity
- Levels of Detail
- Syntax and Semantics of UML
- CASE Tools
- CASE Tools for Visual Studio .NET

**Lesson: Introduction to Design Patterns**

- History and Overview of Patterns
- Problems in Development
- Patterns: Basics
- Patterns: Software Community
- Java Patterns Applicable in .NET
- .NET Specific Patterns
- Multi-tier Patterns
- Architectural Patterns
- Framework and Component Patterns
- GoF Patterns
- Pattern Elements
- Problem & Solution
- Crucial Qualities
- What is Not a Pattern?
- Relating to Methodologies
- Patterns: Aligned with Development Process
- Design Patterns
- Classification Criteria
- Creational; Structural; Behavioral
- Scope Criteria
- Full Description Format
- Design Principles
- The Selection Process

**Session: Creational Design Patterns**

**Lesson: Abstract Factory Design Pattern**

- Introduction
- Illustrative Example
- Context
- Problem
- Solution Description

- Structure
- Implementation
- Code Example
- Consequences
- Related Patterns
- Known Uses

**Lesson: Singleton Design Pattern**

- Introduction & Context
- Solution & Implementation
- C# Example; VB.NET Code; C#; VB.NET
- Consequences
- Supplementary Information

**Lesson: Builder Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Pattern: Dynamic Model
- Consequences
- Supplementary Information

**Lesson: Factory Method Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Outline; Structure & Components
- Solution Dynamics
- Consequences
- Related Patterns
- Supplementary Info

**Lesson: Prototype Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Outline; Structure & Components
- Solution Dynamics
- Consequences
- Related Patterns
- Supplementary Info

**Lesson: Survey of Creational Design Patterns**

- Comparison of Creational Patterns

**Session: Exploring Structural Design Patterns**

**Lesson: Composite Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Structure
- Solution Components
- Solution Implementation
- Code Example
- Consequences – Benefits & Liabilities

**Lesson: Adapter Design Pattern**

- Introduction
- Context and Problem
- Solution Outline; Structure & Components
- Object-Adapter: Solution Structure
- Implementation
- Solution Dynamics
- Consequences
- Related Patterns

**Lesson: Proxy Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Implementation; Proxy Types
- Solution Dynamics
- Consequences
- Code Examples
- Related Patterns

**Lesson: Bridge Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Dynamics
- Consequences
- Related Patterns
- Supplementary Info
- Bridge Code Example, Message Formats

**Lesson: Façade Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Code Example
- Consequences, Benefits

**Lesson: Decorator Design Pattern**

- Introduction & Context

- The Problem, Solution & Implementation
- Solution Outline
- Solution Structure
- Solution Components
- Implementation
- Known Uses
- Consequences
- Related Patterns

**Lesson: Flyweight Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Components
- Applicability
- Implementation
- Known Uses
- Consequences
- Related Patterns

**Lesson: Survey of Structural Design Patterns**

- Common Trends Among Structural Patterns
- Comparison of Structural Patterns

**Session: Exploring Behavioral Design Patterns**

**Lesson: Observer Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Components
- Solution Dynamics
- Example Code, Context
- Advanced Issues
- Consequences - Benefits
- Consequences - Liabilities
- Alternatives
- Related Patterns
- Known Uses

**Lesson: Strategy Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Components; Solution Dynamics
- A Business Case
- Consequences
- Supplementary Info

**Lesson: Iterator Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Components
- Implementation
- Supplementary Info

**Lesson: Visitor Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Components
- Solution Dynamics
- Code Example
- Advanced Implementation Issues
- Consequences, Advantages
- Consequence, Disadvantages

**Lesson: Interpreter Design Pattern**

- Introduction
- Context and Problem
- Solution Outline; Structure & Components
- Solution Dynamics
- Consequences
- Related Patterns

**Lesson: Chain of Responsibility Design Pattern**

- Introduction
- Context and Problem
- Solution Outline; Structure & Components
- Solution Dynamics
- Consequences
- Related Patterns

**Lesson: Command Design Pattern**

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Dynamics
- Consequences
- Related Patterns

**Lesson: Mediator Design Pattern**

- Introduction
- Context and Problem
- Solution Outline; Structure & Components

- Solution Dynamics
- Consequences
- Related Patterns

#### Lesson: Memento Design Pattern

- Introduction & Context
- The Problem, Solution & Implementation
- Solution Dynamics
- Consequences
- Related Patterns
- Implementation
- Supplementary Info

#### Lesson : State Design Pattern

- Summary
- Diagram
- Sample Code
- State Transition Table
- Anonymous State Login Example

#### Lesson : Assertion Design Pattern (non-GoF)

- Context and Problem
- Solution Outline
- Pseudocode

#### Lesson: Dispatcher Design Pattern (non-GoF)

- Context and Problem
- Solution Outline

#### Lesson: Comparison and Summary of Behavioral Patterns

- Common Trends Among Behavioral Patterns
- Encapsulation
- Objects as Arguments
- Decoupling Senders from Receivers

#### Lesson : Other Patterns

- Blackbox: Solution Outline
- Simple Factory: Solution Outline
- Template Design Pattern: Solution Outline

#### Session: Application of Patterns

#### Lesson: Patterns with Client Applications

- Typical Windows Applications
- Patterns to Consider
- Building the Graphical User Interface
- Menu Behavior using the State Pattern
- Viewing Data
- Multiple Views of a Document
- Alternative Client Patterns

#### Lesson: Patterns for the Data Tier

- Accessing Data
- Data Streams
- Related Patterns

#### Lesson: Patterns in the Business Logic Tier

- Control and Flow
- Modifying Process Flow
- Modifying Service Object Behavior
- Sharing State
- Reuse and Performance
- Performance-related Pitfalls
- Improving Reliability

#### Session: Advanced Pattern Topics

#### Lesson: Advanced Topics

- Introspection and Reflection
- The .NET Assembly

- .NET Attributes
- Extending and Writing Attributes
- Patterns for Writing Attributes
- The Data Entity Pattern: Participants
- Object Self-typing
- Determining Interface by Reflection
- Entity EJBs and Persistence
- The Self-Persistence Pattern

#### Lesson: CLR 2.0

- Generics
- Related CLR 2.0 Extensions

#### Addendum: Exploring Frameworks

- Patterns and Productivity
- Other Approaches to Re-Use
- Other Approaches to Productivity
- An Introduction
- How We Use Them
- Requirements
- Benefits & The Liabilities
- Working with Other Approaches
- Compared to Other Approaches
- Compared with Applications

**Need more info?** Please note a more detailed outline, as well as lists of lab exercises and project descriptions, is available. Please contact us at [Training@triveratech.com](mailto:Training@triveratech.com) for info.

**Need courseware?** This course is fully customizable, and also available for license with complete support for qualified organizations. Please contact [Courseware@triveratech.com](mailto:Courseware@triveratech.com) for details

### ► Why Work With Trivera Technologies?

- **We provide a solid object oriented design and development foundation.** Students will learn how to develop (and reuse!) essential OO patterns coding & design skills and concepts properly, using best design practices, grounding them for advanced curriculum. Students will be prepared for designing and implementing real solutions, right after the class ends. Students will learn the importance of developing well-designed OO applications.
- **Our courses are focused - no "fluff" included.** We offer more than a “laundry list” approach to teaching. All lessons have clear objectives, are fundamental to core OOAD development and design practices, and are reinforced by hands-on labs and solid practical examples. Each lesson has performance driven objectives that ensure students will learn technologies and skills core to fundamental OO application and patterns design – nothing more, nothing less.
- **Our materials are comprehensive, and current.** Our comprehensive manuals include not only a hard copy of the course presentation, but also detailed reference notes, pertinent diagrams and charts, current lists of suggested online resources and articles, and often technical

tutorials or white papers geared to the topics at hand.

- **We set you up!** Hands-on courses also include our unique materials for each student, complete with our **LoadNGo Instant Classroom** course set up DVD, software, and a multitude of learning resources that complement the course. Run the course right off the DVD – minimal set up for your company – maximum results for your students.
- **We foster "Learning by Doing"**. Progressive labs are designed in such a way that students get a firm grasp on fundamental skills while they work toward defending a complete application. All labs are take-home, and all solution code is presented in an easy to use self-study format for future use and review.
- **True content ownership gives us flexibility & quality above the rest.** These course materials are wholly-owned by our company and fully customizable - at little or no cost - to help you best meet your learning objectives. We have many dedicated experts available worldwide to instruct your team, and can provide services around the globe, either locally or virtually. We work closely with you to produce the most effective events and materials for your team, within your desired timeframe and budget.
- **We have to adhere to higher standards.** As a courseware provider, our content and hands-on lab materials are licensed internationally by dozens of firms, and are therefore subject to very stringent quality requirements. Not only will your organization benefit from our own technical team's technical expertise, but also the feedback of hundreds of students and trainers using these materials, worldwide, on a regular basis. This unique fact guarantees that our materials are not only robust and interesting, but also technically correct, current and of the highest quality and usability.
- **We bring years of practical, current experience into the classroom and content.** Our instructors and course authors are also skilled mentors, Java, J2EE, .Net, SOA, and web services developers, architects and security-oriented professionals. We believe that learning, using and maintaining solid software execution and delivery methods are as important as gaining sharp coding skills. Best Practices for software development and execution, beyond technical coding skills, are enforced throughout all of our courses and discussions. Our team brings this extensive experience into every classroom and engagement.
- **We're skills-centric.** Although our team has extensive experience using a variety of tools and solutions, our core content is "technology-centric". Our aim is to teach you the best skills and solutions out there – not to sell you software from any particular vendor.
- **We're Java & J2EE authors and industry speakers.** Our team was selected to write the online *J2EE, EJB, EJB CMP-CMR and Web Services Tutorial Series for IBM developerWorks®* ([www.ibm.com](http://www.ibm.com)) These are the same instructors who train our classes and author the courseware. Most of our trainers/consultants have also authored additional articles on web services, EJB< Struts, J2EE and advanced Java topics, and are recognized speakers and presenters on the industry technical seminar circuit. Our team is comprised on several successful published authors. Members of our team have written or contributed to: *Eclipse Kick Start, Mastering Eclipse; Professional Jakarta Struts; Using Java Tools for Extreme Programming; Mastering Resin; Mastering TomCat and others.*
- **Our services are guaranteed.** Whether you're a stakeholder organizing your firm's educational services, a student in our live or virtual classroom or a trainer using our materials to educate your own client or team – **Our core mission is to make YOU a success in the classroom.**

#### ► For Additional Information

**Need dedicated training?** All courses can be brought onsite for a **private presentation**, customized to suit your unique requirements or goals. Please contact [Training@triveratech.com](mailto:Training@triveratech.com) for course details, Public Schedule dates and locations, and Special Discount Offers.

**Need courseware?** Let us take the risk out of your classroom delivery! All materials are also available for corporate license with complete instructor support and free corporate branding. We guarantee our pricing and service. Samples of our course materials, as well as live client references for all of our services are available upon request. Please contact [Courseware@triveratech.com](mailto:Courseware@triveratech.com) for details.

**For more information** about our training, mentoring or courseware licensing options, or to see our complete list of course offerings and services, please visit us at [www.triveratech.com](http://www.triveratech.com), email [Training@triveratech.com](mailto:Training@triveratech.com) or call 609.953.1515.



Trivera Technologies is a 100%  
 Female-Owned Small Business Concern  
 GSA Schedule # GS-35F-0188T  
 Please contact us for details & pricing.