

# The .NET Design Master Class

## From Abstract Design Patterns to Concrete Technologies and APIs

### **Duration**

5 days

### **Instructor**

Dino Esposito

### **Class Limit**

20 students

### **Prerequisite**

None

### **Price**

On-site

Please contact SPC  
for pricing (contact  
information on page 2)

Public Training

\$3,250 (5 days)

\*Discount available for  
early registration

### **Materials Provided**

- Student manual containing the course slides
- Student handouts with class exercises

A good design is the only viable recipe to build software systems with the degree of complexity, extensibility, and security required today. The .NET Design Master Class goes through all the steps that characterize the design of a system. It starts with acknowledgement of requirements and definition of use-cases. It moves to UML diagrams to render use-cases into programmable scenarios. It ends up breaking down the system in components and maps them onto layers and services. It deals with classes and their general attributes of testability, security, extensibility, maintainability, readability, performance.

This workshop is about planning, designing, and implementing a software system. It starts with the acknowledgement of requirements and definition of use-cases. It moves to UML diagrams to render use-cases into programmable scenarios. It ends up breaking down the system in components and their mapping onto layers and services. Finally, it deals with classes and their attributes.

You'll get up close and personal with basic principles low coupling, high cohesion, dependency inversion that should always inspire the design of a modern software system. You'll understand testing, design patterns and idiomatic design targeted to the .NET platform.

The class delves deep inside the sections of a typical layered architecture-presentation, services, business, data access and explains common patterns with their strengths and weaknesses. You'll understand the benefits of a domain-driven organization of the business logic. You'll figure out the general responsibilities of a data access layer and the key role played by the service layer in maintaining a low coupling between presentation and the rest of the system. Finally, you'll see the practical benefits of separation of concerns applied to the user interface.

In addition to instructor-led presentations, the seminar uses numerous conceptual demos and excerpts from sample applications. The students will also participate in a practical design session to model an end-to-end solution using a domain-based approach.

TRAINING

## The .NET Design Master Class

### **Instructor**

Dino Esposito is one of the world's authorities on web technology and software architecture. His extensive hands-on experience includes architecting and building distributed systems for leading banking and insurance companies.

A popular author, Dino's articles are published monthly on at least five different magazines and Web sites. His most recent books include *Programming ASP.NET 3.5 Core Reference* (Microsoft Press, 2008), *Introducing Microsoft ASP.NET AJAX* (Microsoft Press, 2007), and *Programming Microsoft ASP.NET 2.0 Applications Advanced Topics* (Microsoft Press, 2005).

### **Intended Audience**

Any .NET developer would benefit greatly from the .NET Design Master Class training. Basic familiarity with C# and .NET programming is recommended. No specific knowledge of products or technologies is assumed, but a working knowledge of ADO.NET, LINQ, Web, Windows development, and WCF is a plus.

---

For more information on this or other SPC Springboard courses, please visit [www.spcspringboard.com](http://www.spcspringboard.com) or e-mail SPC at [info@spc.ca](mailto:info@spc.ca)

Software Productivity Center Inc.  
Suite 2000 1066 West Hastings Street  
Vancouver, BC V6E 3X2

Vancouver: 604.662.8181 Toll Free: 1.877.548.1948

---

### **Outline**

#### *Principles and Patterns*

- Methodologies, architects, and design
- UML and design
- UML diagram essentials
- Principles of object-oriented design
- Value and use of design patterns
- Types of design patterns
- Refactoring

#### *Class Design*

- Idioms and idiomatic design
- Dependency injection
- Design for testability
- Unit testing
- Design for security
- Threat model
- Design for performance

#### *Domain and Business Logic*

- Modeling the business logic
- Procedural patterns
- Object-based patterns
- Domain model
- Domain-driven design
- Persistence ignorance

#### *Data and Presentation*

- Service layer
- Data transfer objects
- Use-cases and service layer
- Scripting the domain model
- Patterns for application/user interaction
- Model View Controller
- Model View Controller for the Web
- Model View Presenter

#### *Data Access Strategies*

- Responsibilities of the data access layer
- DAL and business layer
- DAL design
- Technologies for the DAL
- Query services
- O/RM tools



TRAINING