Domain-Driven Design Essentials Training

Duration: **5 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

Overview

Domain-Driven Design Essentials is an extensive five-day online training course developed for technical teams that want to understand complex business challenges, design meaningful domain models, and turn those models into maintainable software applications.

Domain-Driven Design goes beyond programming practices. It provides a structured approach to software design and problem-solving by encouraging collaboration between development teams and business domain experts. When software models reflect real business concepts and processes, organizations can manage complexity more effectively and build applications that adapt to evolving requirements.

Throughout the program, participants will examine strategic DDD practices, including Problem Space, Solution Space, Subdomains, Ubiquitous Language, Bounded Contexts, and Context Mapping. The course then introduces tactical design components such as Entities, Value Objects, Aggregates, Domain Services, Domain Events, Factories, and Repositories, demonstrating how these concepts can be implemented in software.

The training concludes with a practical coding workshop where participants analyze a real-world business scenario, develop a domain model, and implement the resulting design using C#.

How Can Domain-Driven Design Benefit Organizations?

Domain-Driven Design emphasizes understanding the business challenge before deciding on its technical implementation. Accurately representing business concepts in software models enables developers, domain experts, and stakeholders to communicate more clearly and make architectural decisions based on a common understanding.

DDD can be particularly beneficial for enterprise applications involving complicated business rules, multiple subdomains, and long-term development and maintenance requirements. However, adopting DDD does not mean that every project must implement all available patterns. The appropriate level of adoption depends on domain complexity and the specific requirements of the application.

Domain-Driven Design and Microservices Architecture

Although Domain-Driven Design and microservices are often discussed together, they address different concerns. DDD concentrates on understanding business domains and establishing meaningful model boundaries. Microservices architecture focuses on organizing applications into independently developed and deployable services.

Subdomain analysis and Bounded Contexts can support decisions about potential microservice boundaries. Nevertheless, a Bounded Context does not necessarily need to be implemented as an independent microservice. Factors such as team responsibilities, data ownership, integration complexity, and operational readiness must also be evaluated.

Domain-Driven Design Essentials Training

Duration: **5 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

Prerequisites

- Participants should have at least intermediate-level programming skills.
- Experience with a widely used object-oriented programming language is sufficient.
- Basic familiarity with C# is recommended, as the practical exercises are developed using C#.
- Participants must have Visual Studio or Rider, Docker, and C# 8.0—or a compatible version communicated before the training—installed on their computers.

Who Should Attend

- Mid-level and senior software developers
- Backend developers
- Software architects and solution architects
- Technical leads and engineering team leaders
- Technical teams collaborating with business analysts and domain specialists
- Teams building microservices or modernizing enterprise software
- Software professionals working with complex business rules

Domain-Driven Design Essentials Training

Duration: **5 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

What You Will Learn

By the end of this training, participants will be able to:

- Understand Domain-Driven Design (DDD) principles and practices.
- Analyze business problems and create effective domain models.
- Explore Problem Space, Solution Space, and Subdomain concepts.
- Establish a Ubiquitous Language between business and development teams.
- Apply Bounded Context and Context Mapping techniques.
- Analyze business processes and domain events using Event Storming.
- Implement Entities, Value Objects, Aggregates, and Domain Services.
- Use Domain Events, Factories, and Repositories.
- Apply DDD principles to Clean Architecture and microservices design.
- Develop maintainable and testable domain models using C#.

Outline

1. Introduction to Domain-Driven Design

- Understanding Domain-Driven Design
- Objectives and fundamental principles of DDD
- Software development challenges addressed by DDD
- Common misunderstandings about Domain-Driven Design
- Reasons DDD may appear complex or difficult to adopt
- Characteristics of good, poor, and effective software design
- The place of DDD within the software development lifecycle

2. Exploring the Business Domain

- Defining a business domain
- How business challenges relate to software models
- Discovering and understanding domain knowledge
- Collaboration between development teams and domain experts

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Essential principles of domain modeling
- Continuously improving and refining domain models
- Keeping the domain model aligned with the implementation

3. Problem Space and Solution Space

- Exploring the concept of Problem Space
- Understanding Solution Space
- Distinguishing business challenges from technical implementations
- Discovering and clarifying business requirements
- Building a domain model before choosing a technical solution
- Translating Problem Space findings into Solution Space decisions

4. Subdomain Analysis

- Understanding the concept of a Subdomain
- Dividing complex domains into manageable areas
- Core Domain and its characteristics
- Supporting Subdomain and its responsibilities
- Generic Subdomain and common capabilities
- Prioritizing subdomains based on business value
- Relating organizational capabilities to subdomains

5. Ubiquitous Language

- Understanding Ubiquitous Language
- Establishing common terminology across business and technical teams
- Defining domain concepts with clarity and consistency
- Applying shared terminology in analysis, documentation, and source code
- Recognizing ambiguous and conflicting domain terminology
- Refining Ubiquitous Language as the domain model evolves

6. Strategic Domain-Driven Design

- Objectives of strategic domain design

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

- Decomposing large and complex software systems
- Discovering boundaries between domains and models
- Aligning software boundaries with business capabilities
- Understanding the architectural consequences of strategic decisions

7. Bounded Context

- Understanding Bounded Contexts
- Establishing boundaries for domain models
- Recognizing different meanings of the same concept across contexts
- Exploring the relationship between Bounded Contexts and Subdomains
- Principles for identifying appropriate context boundaries
- Challenges resulting from poorly defined context boundaries

8. Context Mapping

- Understanding the purpose of a Context Map
- Representing relationships among Bounded Contexts
- Examining system and team boundaries
- Managing data and model sharing across contexts
- Discovering integration points between contexts
- Using Context Mapping findings to guide architectural design

9. Event Storming

- Introduction to Event Storming
- Applying Event Storming to collaborative domain discovery
- Discovering and identifying Domain Events
- Examining Commands, Actors, and business rules
- Facilitating collaboration between technical teams and domain experts
- Deriving potential Bounded Contexts from Event Storming results

10. Tactical Domain-Driven Design

- Understanding the objectives of tactical design

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Implementing domain models in source code
- Assigning business rules to appropriate software layers
- Principles of a Rich Domain Model
- Relationships among tactical design components
- Recognizing and avoiding Anemic Domain Models

11. Entities and Value Objects

- Understanding Entities
- Identity and object lifecycle concepts
- Understanding Value Objects
- Value-based equality
- Principles of immutability
- Key distinctions between Entities and Value Objects
- Choosing the correct component for domain modeling

12. Aggregates and Aggregate Roots

- Understanding Aggregates
- Responsibilities of an Aggregate Root
- Enforcing business rules and invariants
- Establishing appropriate Aggregate boundaries
- Managing relationships among Aggregates
- Preventing unnecessarily large Aggregate structures
- Defining transaction and consistency boundaries

13. Domain Services and Application Services

- Understanding Domain Services
- Representing domain behavior outside Entities and Value Objects
- Roles and responsibilities of Application Services
- Distinguishing application logic from domain logic
- Assigning business rules to the appropriate services

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

14. Domain Events

- Understanding Domain Events
- Representing meaningful changes in the business domain
- Creating and publishing Domain Events
- Minimizing unnecessary dependencies among software components
- Relating Domain Events to business workflows

15. Factories, Repositories, and Modules

- Understanding the purpose of the Factory pattern
- Handling complex object creation processes
- Introduction to Repositories
- Decoupling domain models from data access implementations
- Establishing Repository boundaries
- Structuring domain models with Modules

16. Domain-Driven Design and Clean Architecture

- Fundamentals of Clean Architecture
- Understanding the domain layer within the application architecture
- Organizing dependencies toward the Core Domain
- Separating Domain, Application, and Infrastructure responsibilities
- Designing domain models independently of technology choices
- Creating maintainable and testable software systems

17. Practical Coding Workshop

- Exploring the domain through a real-world business scenario
- Analyzing Problem Space and Solution Space
- Identifying Subdomains and the Core Domain
- Establishing a Ubiquitous Language
- Defining Bounded Context boundaries
- Applying strategic and tactical DDD principles

Domain-Driven Design Essentials Training

Duration: **5 Day**

BILGINÇ IT Academy
info@bilginc.com
bilginc.com

- Implementing the domain model in C#
- Developing Entities, Value Objects, and Aggregates
- Implementing Repositories and Domain Services
- Live coding exercises and solution evaluation