olsen software ltd

+44 7989 401397
info@olsensoft.com

Domain-Driven Design (3 days)

Course overview

Domain-Driven Design has attracted quite a lot of attention in recent years. This course explains the concepts of DDD, and explores how it can help us model complex software systems. We'll lift the lid on key concepts such as Bounded Contexts, Aggregates and Ubiquitous Language, and take a pragmatic look at how to apply these concepts to address real challenges. The course also covers the Command-Query Responsibility Segregation (CQRS) pattern, and describes how it fits in with DDD.

The course primarily addresses architecture and design challenges. We also dip down into code where appropriate, to illustrate key points and to make things tangible. Code examples are presented in Java (other languages can be catered for, on demand)

What you'll learn

- Understanding DDD concepts and terminology
- Effective model-driven design
- Bounded contexts and ubiquitous language
- Applying DDD in practice
- CQRS
- Implementation examples

Prerequisites

- Architects, system designers, and developers

Course details

- Introduction to Domain-Driven Design: What is Domain-Driven Design (DDD); What challenges does DDD address; Essential patterns and practices of DDD
- Problem Domains: Understanding the problem domain; Techniques and practices; Core domains; Defining clean boundaries
- Effective Model-Driven Design: Domain models; Defining a Ubiquitous Language; Domain model implementation patterns; Bounded Contexts; Applying DDD in practice
- Bounded Context Integration: Integrating Bounded Contexts; Integrating distributed Bounded Contexts; Messaging; REST/RPC
- Overview of DDD Patterns: Entities, value objects, and domain services; Aggregates, factories, and repos; Domain events; Event sourcing
- Value Objects and Entities: When to use a value object; Persistence options; How to implement entities; Entity best practices

- Domain Services and Domain Events: When and how to use domain services; Domain services vs. application services; Domain service patterns; Domain event actions; Implementation options
- Aggregates: The importance of aggregates in DDD; Defining aggregate boundaries; Implementing aggregates; Persistence
- Factories and Repositories: The purpose of factories; The purpose of repositories; Patterns and anti-patterns
- Introduction to CQRS: What is CQRS; Eventual consistency; Commands and command buses; Repositories; Events and event buses
- Going Further with CQRS: Implementing a REST UI; Event-sourced aggregates; Sagas