In this document

Domain Driven Design

What is DDD?

ABP framework provides an **infrastructure** to make **Domain Driven Design** based development easier to implement. DDD is defined in the Wikipedia as below:

> **Domain-driven design** (**DDD**) is an approach to software development for complex needs by connecting the implementation to an evolving model. The premise of domain-driven design is the following:
>
> - Placing the project's primary focus on the core domain and domain logic;
> - Basing complex designs on a model of the domain;
> - Initiating a creative collaboration between technical and domain experts to iteratively refine a conceptual model that addresses particular domain problems.

Layers & Building Blocks

ABP follows DDD principles and patterns to achieve a layered application model which consists of four fundamental layers:

- **Presentation Layer**: Provides an interface to the user. Uses the *Application Layer* to achieve user interactions.
- **Application Layer**: Mediates between the Presentation and Domain Layers. Orchestrates business objects to perform specific application tasks. Implements use cases as the application logic.
- **Domain Layer**: Includes business objects and the core (domain) business rules. This is the heart of the application.
- **Infrastructure Layer**: Provides generic technical capabilities that support higher layers mostly using 3rd-party libraries.

DDD mostly interest in the **Domain** and the **Application** layers, rather than the Infrastructure and the Presentation layers. The following documents explains the **infrastructure** provided by the ABP Framework to implement **Building Blocks** of the DDD:

- **Domain Layer**
  - Entities & Aggregate Roots
  - Repositories
  - Domain Services
  - Value Objects
  - Specifications
- **Application Layer**
  - Application Services
  - Data Transfer Objects (DTOs)
  - Unit of Work

The Ultimate DDD Implementation Guide

4.1 (latest) English

Filter topics

- Getting Started
- Startup Templates
- Tutorials
- Fundamentals
- Infrastructure
- Architecture
  - Modularity
  - Domain Driven Design
    - Overall
    - Domain Layer
    - Application Layer
    - Guide: Implementing DDD
  - Multi Tenancy
  - Microservices
- API
- User Interface
- Data Access
- Real Time
- Testing
- Samples
- Application Modules
- Release Information
- Reference
- Contribution Guide

See the Implementing Domain Driven Design guide as a **complete reference**. The Guide explains the Domain Driven Design and introduces explicit **rules and examples** to give a deep understanding of the **implementation details**.

Share on :

In this document