30주차 발표_민서연

DDD

01 DDD

Domain Driven Design

도메인 주도 설계

도메인 패턴을 중심에 놓고 설계하는 패턴으로, 도메인 간의 상호 작용이 설계의 중심이 된다.

비즈니스 도메인의 개념과 용어를 소프트웨어 코드의 구조와 언어로 표현하여 개발 과정에서
도메인 전문가와 개발자 사이의 의사 소통을 원활하게 하며 도메인 지식을 더 잘 반영할 수 있다.

01 DDD

도메인이란

영역, 집합이라는 뜻

DDD에서의 Domain은 비즈니스 Domain의 의미로, 유사한 업무의 집합이다.

비즈니스 Domain: 유사한 업무의 집합, 기업의 활동 영역 정의

예) 카카오: 채팅 서비스 제공, 배달의 민족: 음식 배달 서비스 제공, 토스: 금융 서비스 제공

01 DDD

원칙과 패턴

유비쿼터스 랭귀지

도메인에 대해 모든 관계자가 공통으로 사용하는 언어
코드, 문서, 대화 등 모든 커뮤니케이션에서 일관되게 사용되어야 하고
도메인 지식을 공유하며 오해를 줄이고 협업을 촉진할 수 있음

01 DDD

원칙과 패턴

애그리거트

서로 관련된 도메인 객체들의 묶음으로 하나의 루트 엔티티를 가지고 루트 엔티티를 통해서만
 다른 객체에 접근 할 수 있음.
애그리거트는 일관성을 유지하기 위해 경계를 정하고 불변식을 지키는데 이는 도메인 모델의 구성요소로
도메인 객체들간의 관계와 상호작용을 명확하게 제한 함

01 DDD

원칙과 패턴

도메인 서비스

엔티티나 값 객체로 표현하기 어려운 도메인 로직을 수행하는 서비스.
도메인 서비스는 상태를 가지고 있지 않으며 파라미터로 엔티티나 값 객체를 받아서 결과를 반환함
비즈니스 로직을 명확하게 구현하고 중복을 줄이며 엔티티나 값 객체를 받아서 결과를 반환함.

01 DDD

원칙과 패턴

리포지토리

애그리거트의 영속성을 관리하는 저장소로 애그리거트의 생성, 조회, 수정, 삭제 등의 기능을 제공한다.

인터페이스와 구현으로 분리되어야하며, 구현은 메모리, 데이터베이스, 파일 등 다양한 방식으로 이루어 질 수 있다.

01 DDD

장점

- 비즈니스 도메인에 집중

  개발자들은 도메인 전문가와 협력을 통해 비즈니스 도메인에 대한 이해를 깊이 있게 할 수 있다.
  도메인의 복잡성을 이해하고 실제 요구 사항을 반영하는 소프트웨어를 개발 할 수 있도록 돕는다.

- 의사소통 협업의 강화

  비즈니스 도메인의 용어와 개념을 코드에 반영함으로써 비즈니스 전문가와 개발자는 동일한 언어를 사용하여 서로간의 의사소통을 원활하게 할 수 있다.

01 DDD

단점

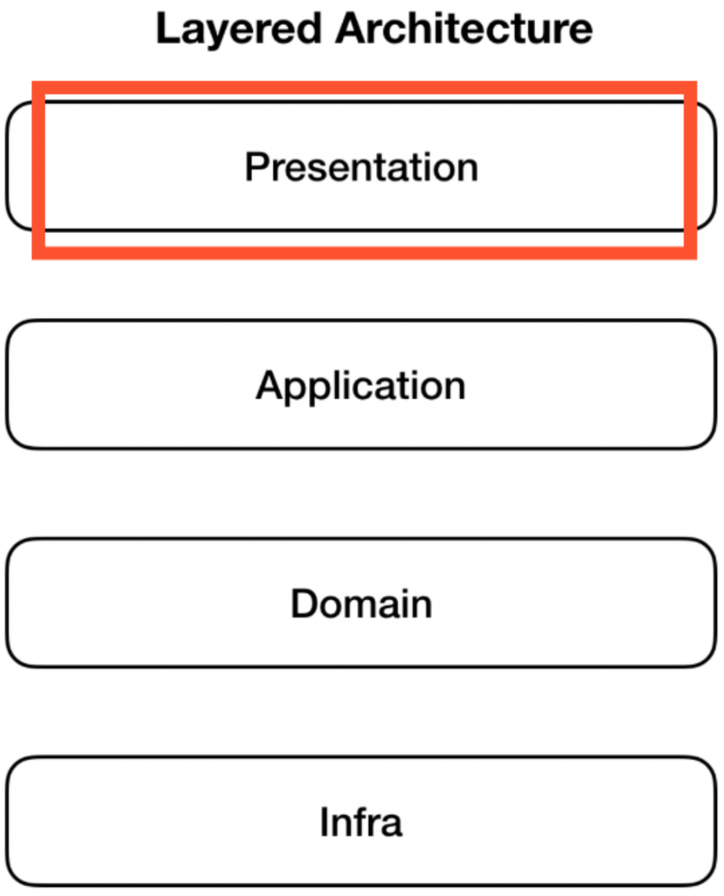
- 초기비용과 시간

  초기에 도메인 모델을 개발하고 도메인 전문가와의 협업을 강조하는 방법론으로
  도메인의 복잡성이 큰 경우에는 초기 분석과 설계에 많은 시간과 노력이 필요할 수 있다.

- 과도한 추상화

  과도한 추상화는 코드의 이해를 어렵게하고 개발자들이 복잡한 구조를 해석하고 유지보수 하는 데
  어려움을 줄 수 있다.

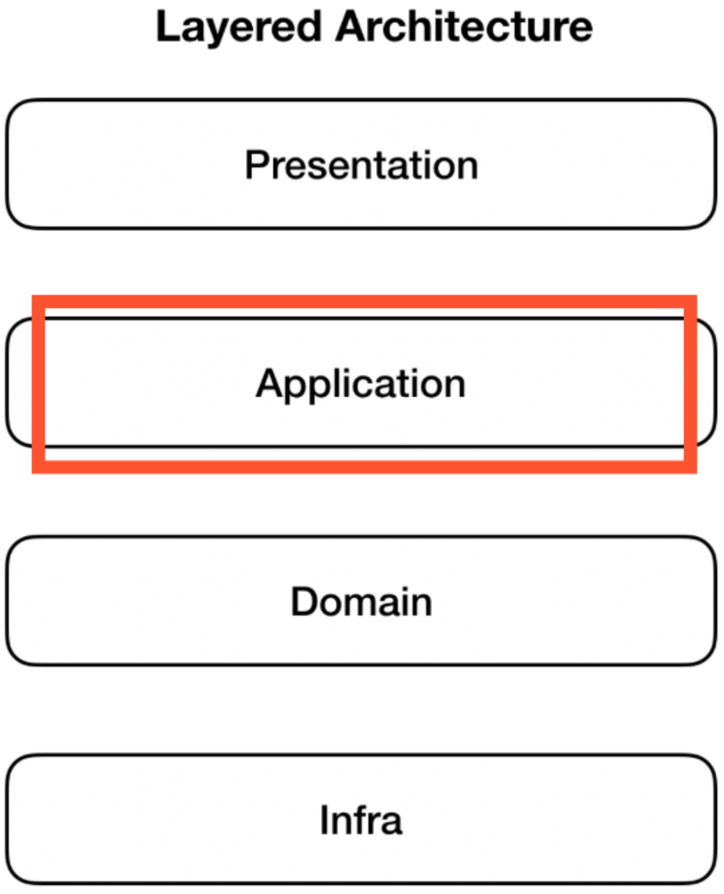
01 DDD



Presentation 계층 - Controller

사용자의 요청을 처리하는 Layer
사용자에게 UI나 클라이언트에게 Response를 보내는 역할을 하는
모든 클래스를 포함
Client로부터 request를 받고 response를 하는 모든 API를 담는다.

01 DDD



Application 계층 - Service

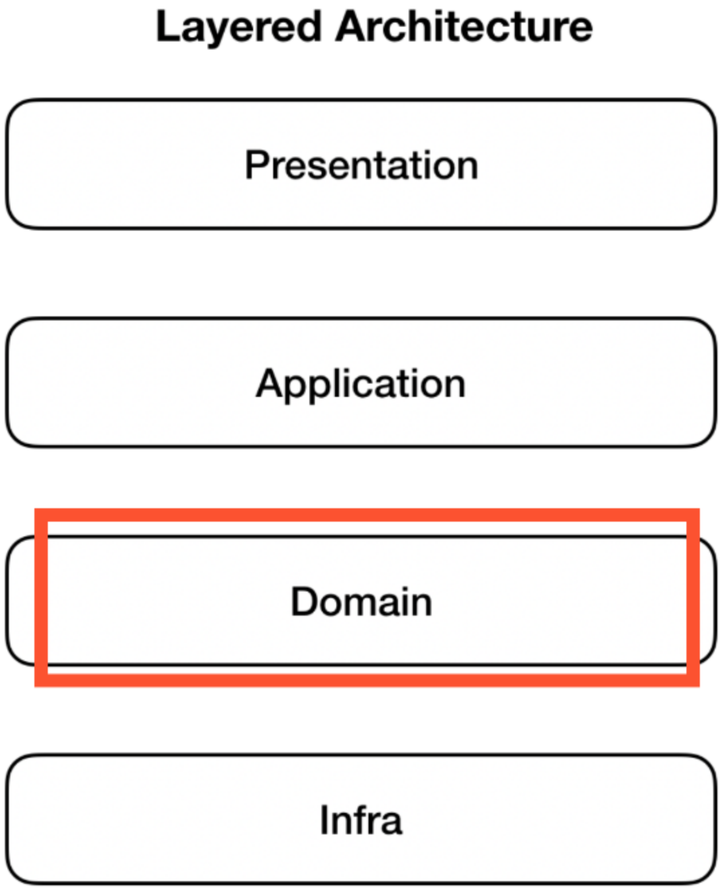
비즈니스 로직을 정의하는 Layer

Domain Layer와 Infrastructure 계층을 연결

많은 정보를 갖고 있지 않도록 유지하는 것이 좋음

트랜잭션 단위, DTO 변환, 엔티티 조회 및 저장이 포함

01 DDD



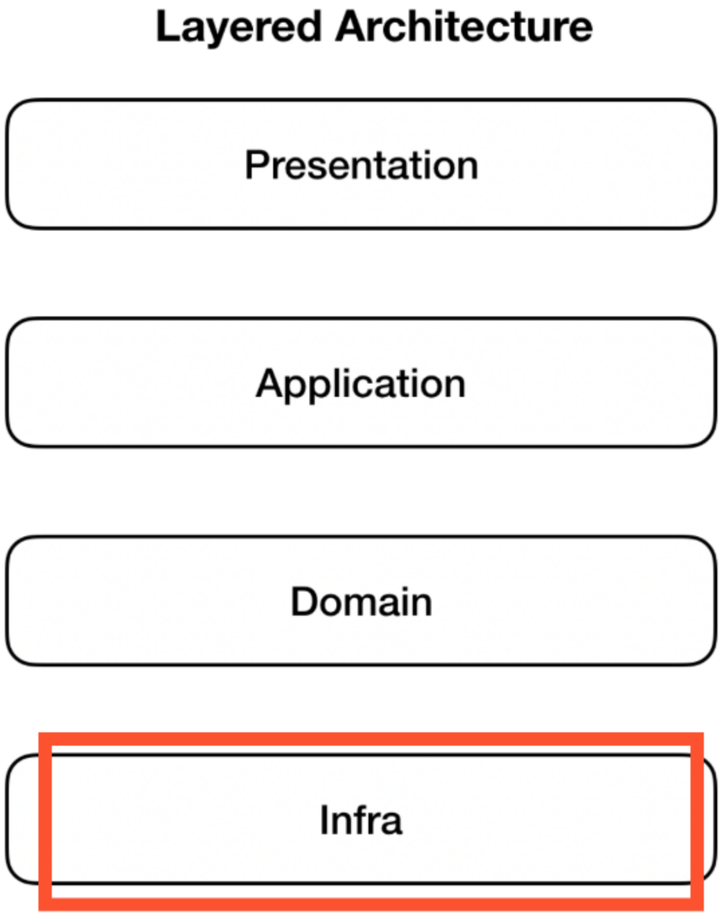
Domain - Model

비즈니스 규칙, 정보에 대한 실질적인 도메인의 정보를 가지고 있다.

Entity를 활용해서 도메인 로직이 실행된다.

주로 상태 제어 역할을 한다.

01 DDD



Infrastructure - Respository

외부와의 통신 (DB, 메세징 시스템)을 담당한다.
해당 계층에서 얻어온 정보를 Application Layer나 Domain Layer에 전달하는 역할을 한다.

01 DDD

DDD와 유비쿼터스

개발은 혼자 하는 것이 아닌 팀원들과 다같이 하는 팀 프로젝트이기 때문에 모두가 이해할 수 있는
언어로 의사소통하고 이러한 언어로 소통하는 것이 중요하다!

➡ 이러한 중요성을 잘 나타내는 것이 DDD 패턴

DDD에서는 유비쿼터스 언어를 모든 프로젝트 구성원이 사용하는 비즈니스 및 도메인 언어라고 정의함

01 DDD

유비쿼터스 언어 만들기

용어사전 예시)

한글명	영문명	설명
상품	product	메뉴를 관리하는 기준이 되는 데이터
메뉴 그룹	menu group	메뉴 묶음, 분류
메뉴	menu	메뉴 그룹에 속하는 실제 주문 가능 단위
메뉴 상품	menu product	메뉴에 속하는 수량이 있는 상품
금액	amount	가격 * 수량
주문 테이블	order table	매장에서 주문이 발생하는 영역
빈 테이블	empty table	주문을 등록할 수 없는 주문 테이블
주문	order	매장에서 발생하는 주문
주문 상태	order status	주문은 조리 → 식사 → 계산 완료 순서로 진행된다.
방문한 손님 수	number of guests	필수 사항은 아니며 주문은 0명으로 등록할 수 있다.
단체 지정	table group	통합 계산을 위해 개별 주문 테이블을 그룹화하는 기능
주문 항목	order line item	주문에 속하는 수량이 있는 메뉴
매장 식사	eat in	포장하지 않고 매장에서 식사하는 것

- 팀내에서 사용하는 언어와 비즈니스 언어를 일치 시켜서 소프트웨어 모델에 반영하며 모든 팀원이 동일한 언어로 소통 할 수 있다.
- 개발자들이 비즈니스 전문가와 의사소통하며 비즈니스를 이해하는 속도가 향상된다.
- 유비쿼터스 언어로 생성한 도메인 모델을 통한 추상화로 복잡한 도메인을 더 이해하기 쉽게 한다.

02 출처

https://youwjune.tistory.com/38
https://happycloud-lee.tistory.com/94
https://tech1.tistory.com/93
https://medium.com/@jnso5072/thinking-%EA%B0%9C%EB%B0%9C%EC%9E%90%EB%8A%94-%EC%9C%A0%EB%B9%84%EC%BF%BC%ED%84%B0%EC%8A%A4-%EC%96%B8%EC%96%B4%EB%A1%9C-%EB%8C%80%ED%99%94%ED%95%98%EA%B3%A0-%EC%BD%94%EB%93%9C%EC%97%90-%EB%85%B9%EC%97%AC%EB%82%B4%EC%95%BC-%ED%95%9C%EB%8B%A4-f50126e45fc2
https://backendbrew.com/docs/ddd/strategic/ubiquitous