Clean Architecture

A Clean Architecture é um modelo que tem como objetivo o desacoplamento entre as regras de negócio, ou domínio, da aplicação e os recursos externos como frameworks e banco de dados

FrontPage

Not Secure | mail.fitnesse.org/FrontPage

FitNesse Features Download Plug-ins User Guide

FrontPage

The fully integrated standalone wiki and acceptance testing framework

Be-monster not thy feature, wer't my fitnesse
-- Shakespeare, King Lear

Download FitNesse

It's a Collaboration tool

Since FitNesse is a wiki web server, it has a very low entry and learning curve, which makes it an excellent tool to collaborate with, for example, business stakeholders.
Read more...

It's a Test tool

The wiki pages created in FitNesse are run as tests. The specifications can be tested against the application itself, resulting in a roundtrip between specifications and implementation.
Read more...

It's Open

FitNesse is an open source project. The code base is not owned by any company. A lot of information is shared by the FitNesse community. It's extremely adaptable and is used in areas ranging from Web/GUI tests to testing electronic components.
Read more...

FitNesse v20201213
Front Page - User Guide - Features - Download - Plugins - Fixtures - Slack Community

"The center of your application is not the database, nor is it one or more of the frameworks you may be using. The center of your application is the use cases of your application"

Robert Martin

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

Os use cases expõe o comportamento demandado pelos drivers (atores), e orquestram as entidades e os recursos externos como banco de dados, APIs, Filas

API
Queue
CLI
Signup
GetAccount
RequestRide
AcceptRide
StartRide
UpdatePosition
FinishRide
GetRide
application/use cases layer

Caso de uso é diferente de CRUD?

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

Entidades são responsáveis por abstrair as regras de negócio independentes, que podem ser desde um objeto com métodos até mesmo um conjunto de funções

domain/entity layer

O que são regras de negócio <u>independentes</u>?

- A placa do carro é válida?
- Qual é a distância entre duas coordenadas?
- Quanto é a tarifa da corrida?
- Como é a mudança de status de uma corrida?

Essas entidades não são as mesmas que utilizamos em um ORM

Domain Objects vs. ORM Objects

O problema do domínio anêmico

The Clean Architecture
Devices
Web
Controllers
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

API
Queue
RequestRide
Account
a
b
c
Ride
a
b
c
domain/entity layer
infra/interface adapters layer
application/use case layer

Os interface adapters fazem a ponte entre os casos de uso e os recursos externos

Tratamento de requisições e respostas HTTP, lidando com parâmetros

Acesso ao banco de dados, todo o código SQL pertence à esta camada

Integração com uma API externa

Escrita e leitura no sistema de arquivos

Conversão de dados para formatos específicos como CSV e PDF

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

infra/interface adapters layer
API
RequestRide
Account
a
b
c
Ride
a
b
c
domain/entity layer
HttpServer
application/use case layer
infra/frameworks and drivers layer

Por fim, os frameworks and drivers são o nível mais baixo de abstração, é a interação com a tecnologia, com os componentes que realizam a conexão com o banco de dados, as requisições HTTP, a interação com o sistema de arquivos ou o acessa aos recursos do sistema operacional

The Clean Architecture
Devices
Controllers
Web
Use Cases
Entities
Gateways
Presenters
DB
UI
External
Interfaces
Enterprise Business Rules
Application Business Rules
Interface Adapters
Frameworks & Drivers

Quem está dentro não conhece quem está fora, mas quem está fora conhece quem está dentro, as entidades não conhecem os use cases e esses não conhecem a implementação dos interface adapters, que não conhecem a implementação dos frameworks and drivers

São somente 4 camadas?

Uma camada é uma fronteira lógica entre um conjunto de componentes, que tem uma responsabilidade bem definida

As camadas não tem necessariamente
relação física com as pastas

Como incializamos a aplicação?

O main é o ponto de entrada da aplicação (HTTP, CLI, UI, Testes), é lá que as fábricas e estratégias são inicializadas e as injeções de dependência são realizadas durante a inicialização

When composing an application from many loosely coupled classes, **the composition should take place as close to the application's entry point as possible**. The Main method is the entry point for most application types. The Composition Root composes the object graph, which subsequently performs the actual work of the application

Entry point
Composition Root