SOLID PageObjectS

Remigiusz Dudek

Introduction

Remigiusz Dudek

Senior QA Engineer at Klika Tech co.

- Test Warez 2019: Architecture driven QA
- Test Warez 2018: Testing hexagonal Architecture
- Test Warez 2017: Consumer Driven Contracts
- Test Warez 2016: From CI to CD
- Test Warez 2015: 7 Cardinal Sins of automted testing

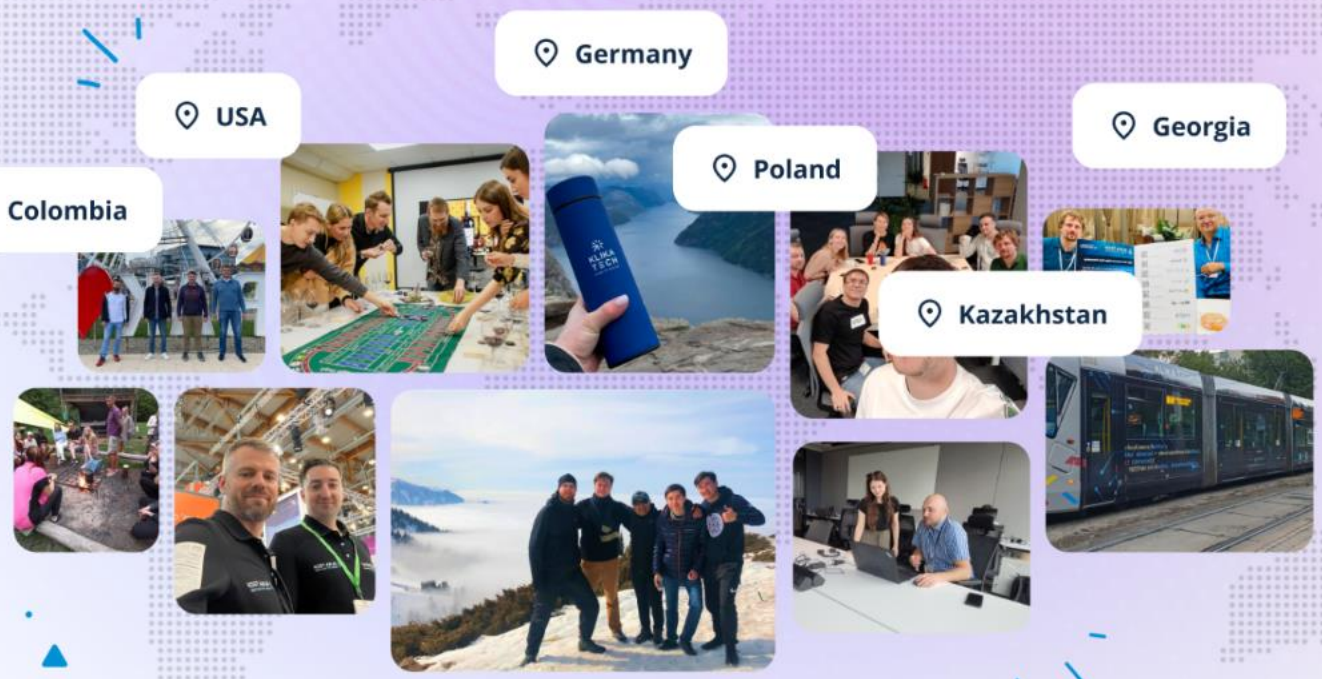
WHO WE ARE

Klika Tech is an **IoT & cloud** product and solutions development company with 335 employees in 8 countries.

aws partner network

Advanced
Consulting Partner

- IoT Competency
- Digital CX Services Competency
- DevOps Competency
- AWS IoT Core
- Amazon API Gateway
- AWS CloudFormation
- AWS IoT GreenGrass
- AWS Lambda



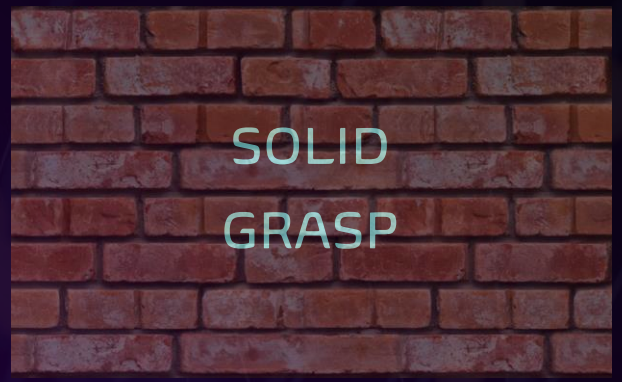
Rules of Engagement

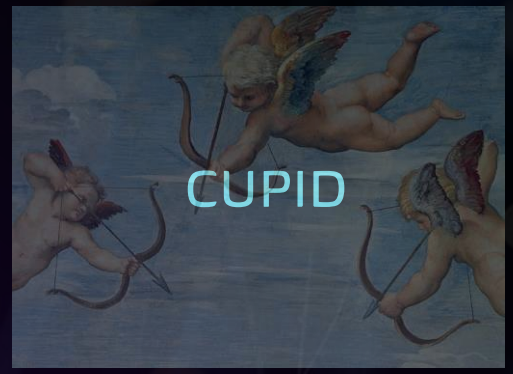
- Mutual respect
- Constructive critique
- Solution questioning
- Safe environment for trying new things

Overview

Given OO paradigm how to create a SOLID solution you'd LOVE to work with.

OOP/OOD





OOP

„making code easy to read, makes it easier to write“ by R. C. Martin

Domain Driven Rules

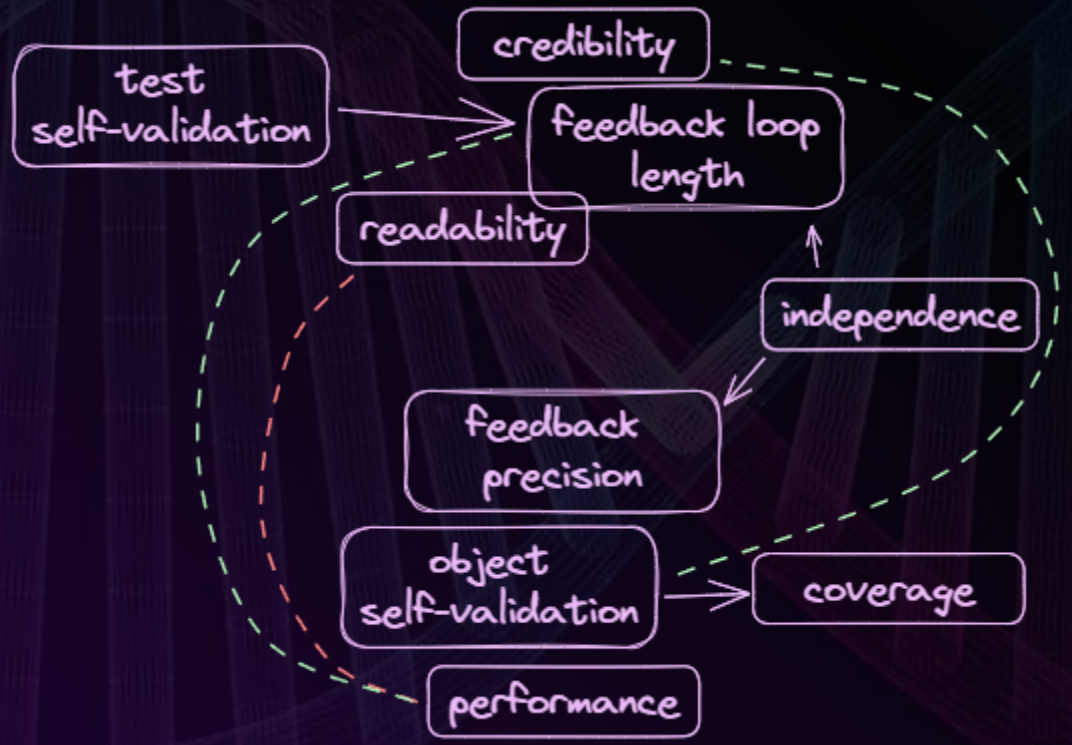
- Always optimize code for readability
- Use Domain laguage
- API should tell the story, should lead you by your hand
- Idempotency

OOP

„when you focus on what's important, everything else falls into place“

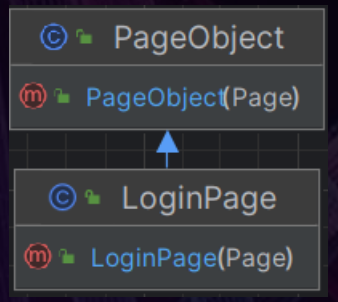
OOP

„when you focus on what's important, everything else falls into place“



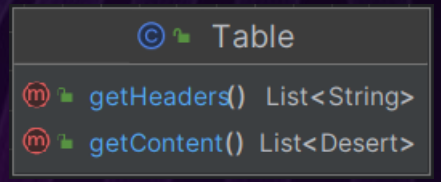
OOP

„Any fool can write code that a computer can understand. Good programmers write code that humans can understand“ by M. Fowler



OOP

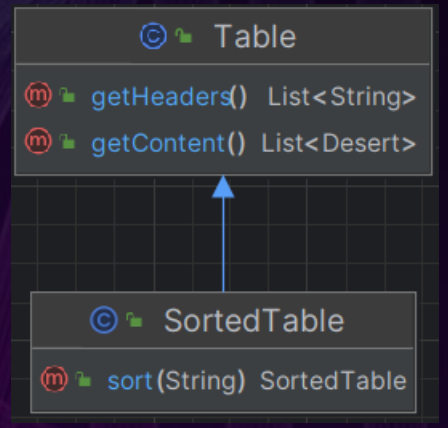
„You can‘t build a great building on a weak foundation“



Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Gingerbread	356	16	49	3.9

OOP

„You can‘t build a great building on a weak foundation“



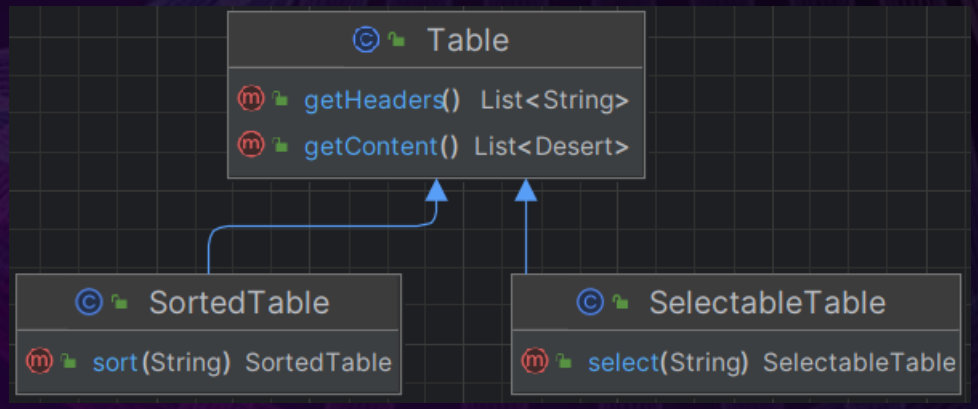
Nutrition

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Marshmallow	318	0	81	2

Rows per page: 5 1–5 of 13

OOP

„You can‘t build a great building on a weak foundation“



Nutrition

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Marshmallow	318	0	81	2

Rows per page: 5 1–5 of 13

OOP

„You can‘t build a great building on a weak foundation“

Nutrition

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Marshmallow	318	0	81	2

Rows per page: 5 1–5 of 13

OOP

„You can‘t build a great building on a weak foundation“

Nutrition

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Marshmallow	318	0	81	2

Rows per page: 5 1–5 of 13

OOP

„You can‘t build a great building on a weak foundation“

Composition over inheritance

- Inheritance = A is B
- Composition = A is composed of B

Nutrition

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Marshmallow	318	0	81	2

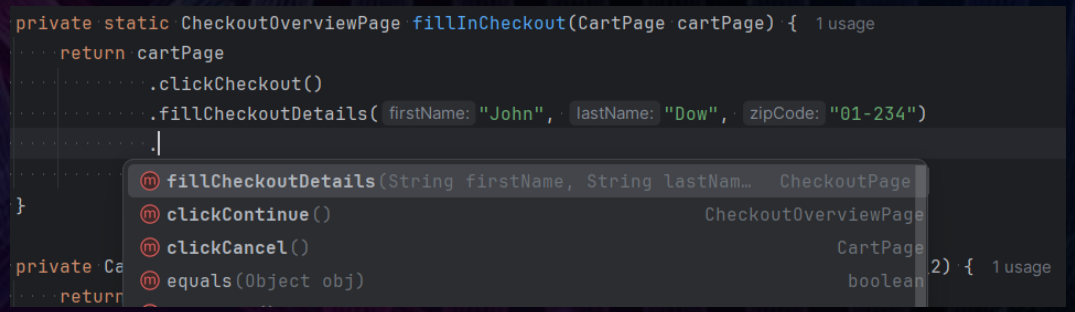
Rows per page: 5 1–5 of 13

OOP

„You can‘t build a great building on a weak foundation“

Fluent API

- Pros
  - Intuitiveness – design leads by hand
  - Encapsulation – protected from implementation details
- Caveats
  - Directions of flow
  - Flow breakers – getters – convention needed



SOLID

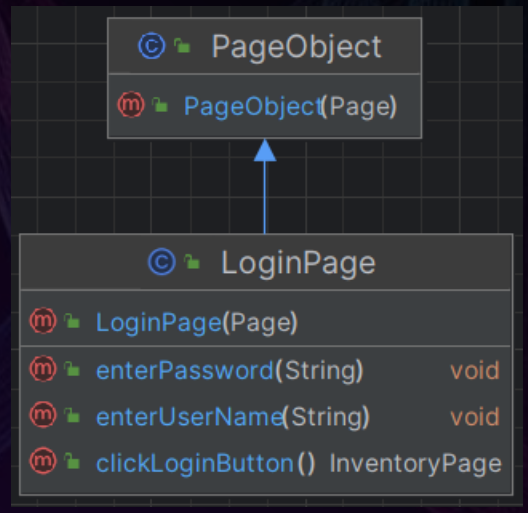
„The price of greatness is responsibility“ by W. Churchill

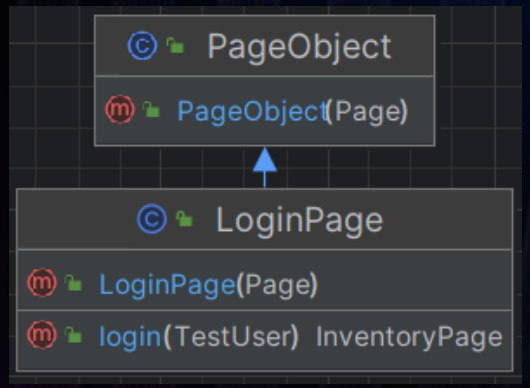
Single responsibility

- Translates business interface to object interface

.VS.

- Translates GUI widgets into object interface





SOLID

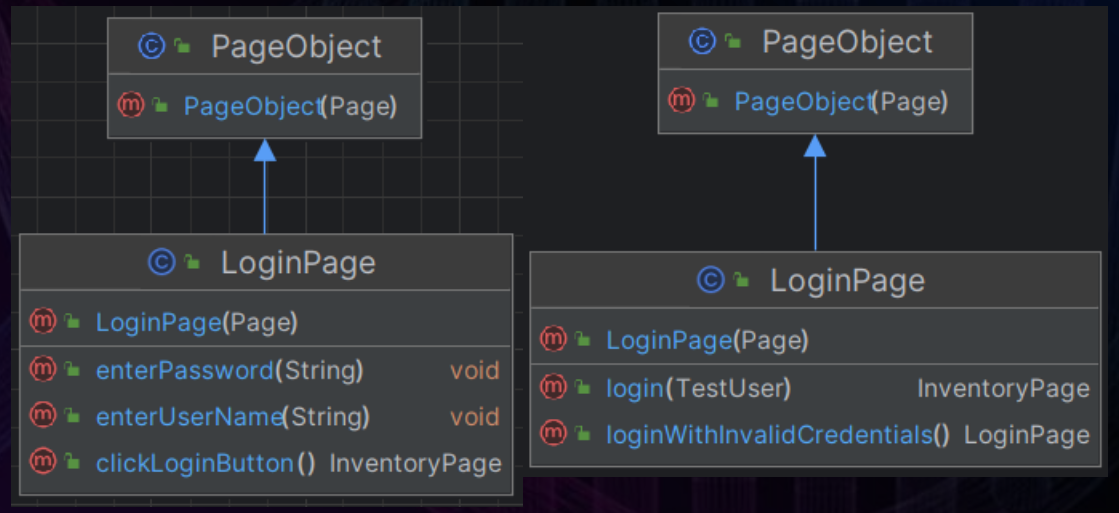
„The price of greatness is responsibility“
by W. Churchill

Single responsibility

- Translates business interface to object interface

.VS.

- Translates GUI widgets into object interface

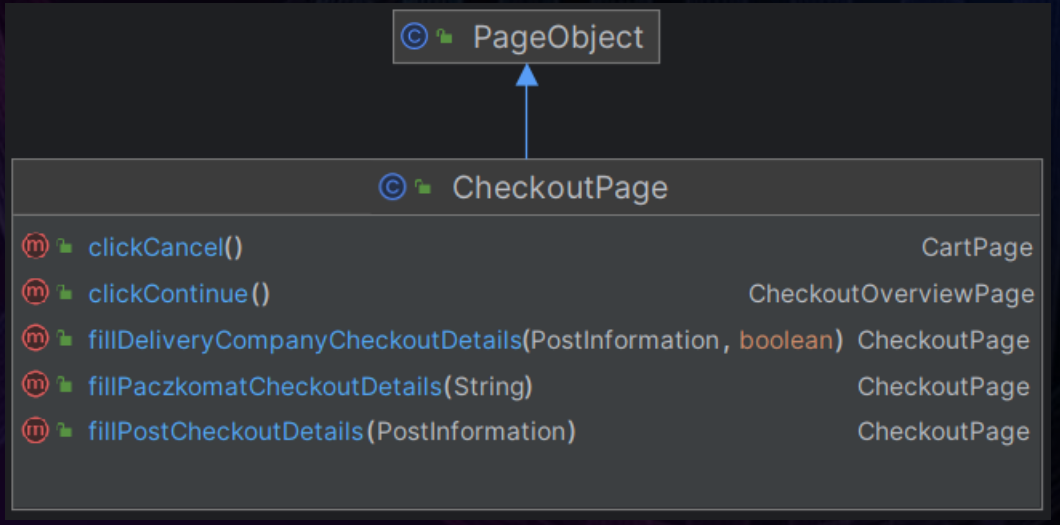


SOLID

„Close your eyes, it helps openning your mind"
by Unknown

Open Close principle

- The goal of the design is to protect us from variability of the world

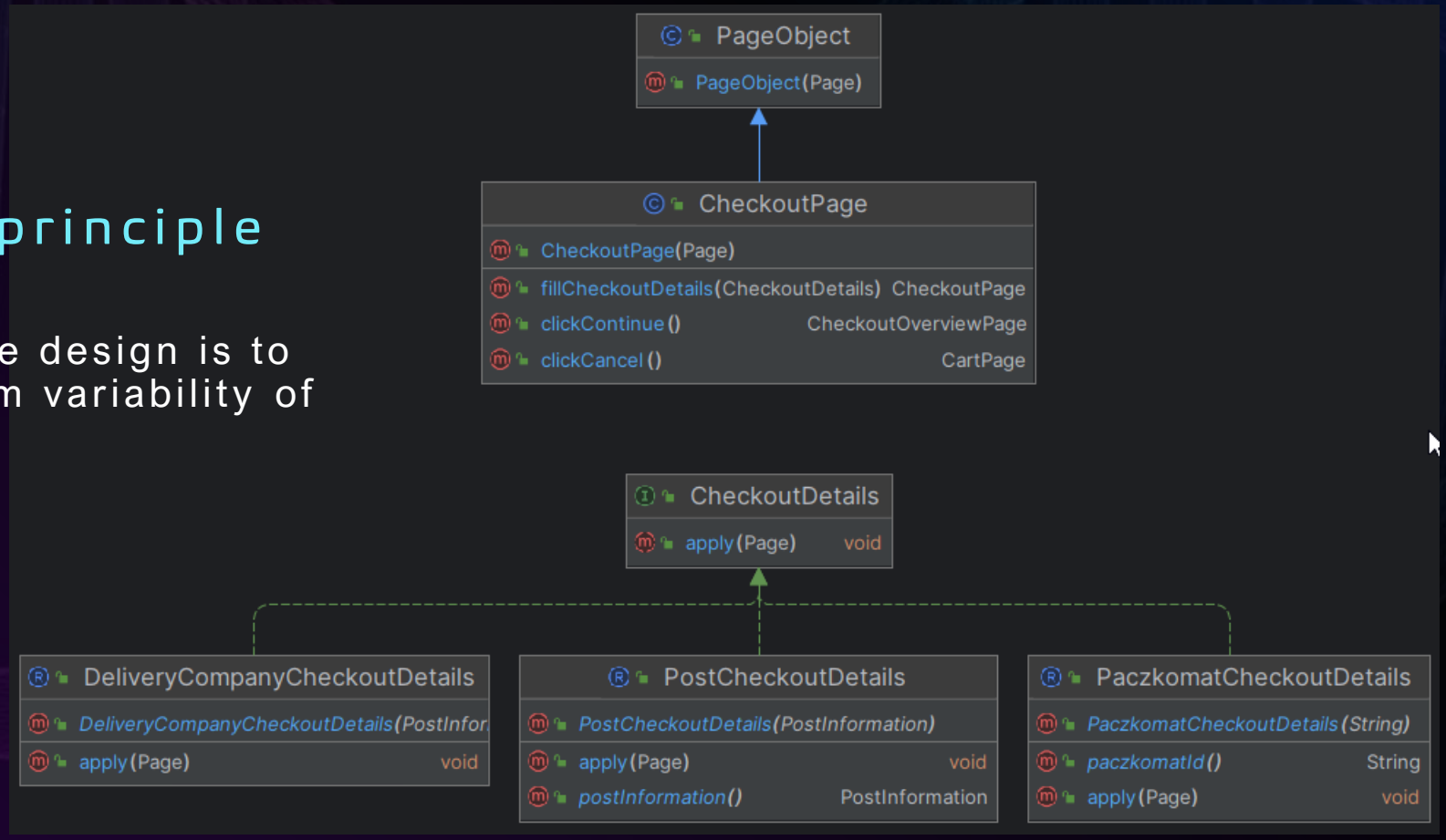


SOLID

„Close your eyes, it helps openning your mind"
by Unknown

Open Close principle

- The goal of the design is to protect us from variability of the world

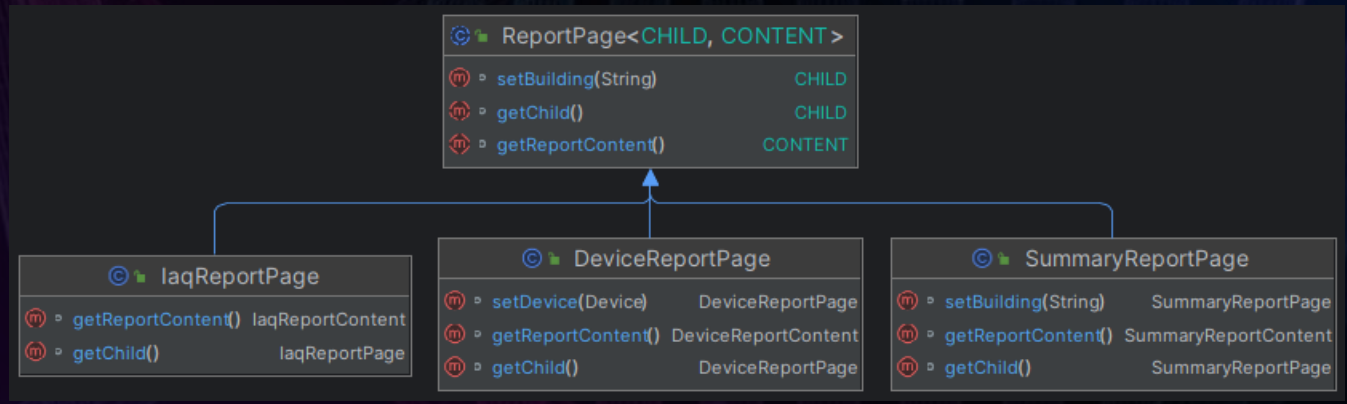


SOLID

„Trap is only a trap if you don't know about it, if you know about it, it's a challange"

Liskov substitution

- Principle of no setting traps for yourself

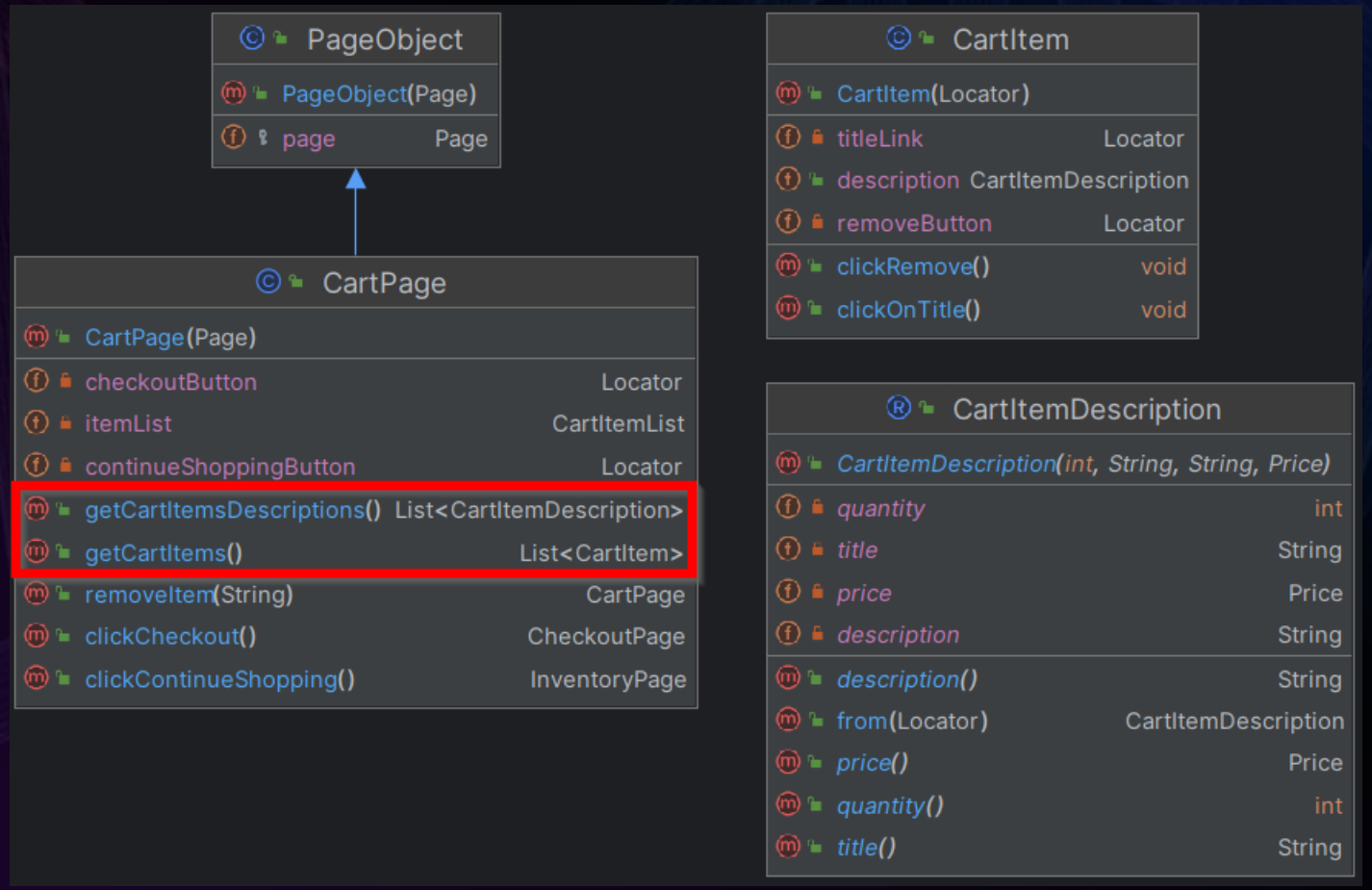


SOLID

„Everything is possible, but everything is not necessary“
by A. Part

Inteface segregation

- Expose only what is required

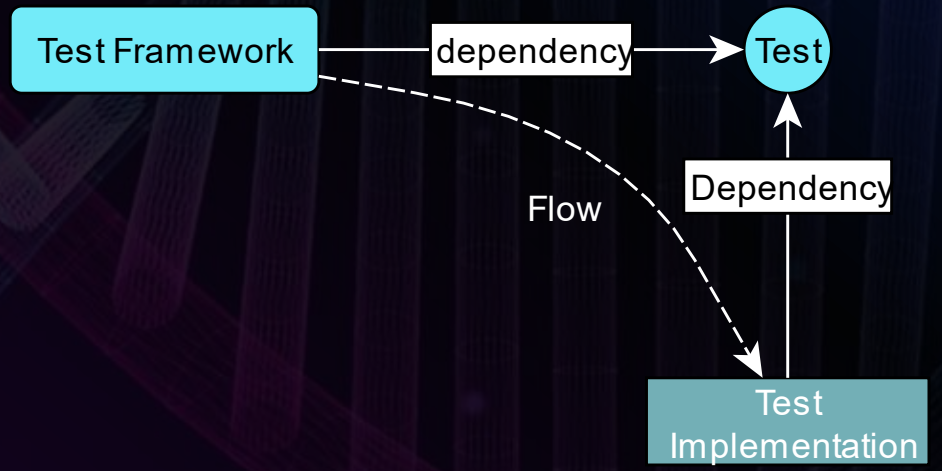


SOLID

„The happiness is to enjoy the present, without anxious dependence upon the future“ by Seneca

Dependency Inversion

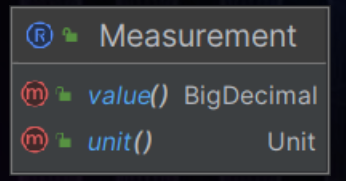
- Don’t depend on concretes, depend on abstraction
- High-level module should not depend on low-level module



GRASP

„We are drowning in information but starved for knowledge“
by John Naisbitt

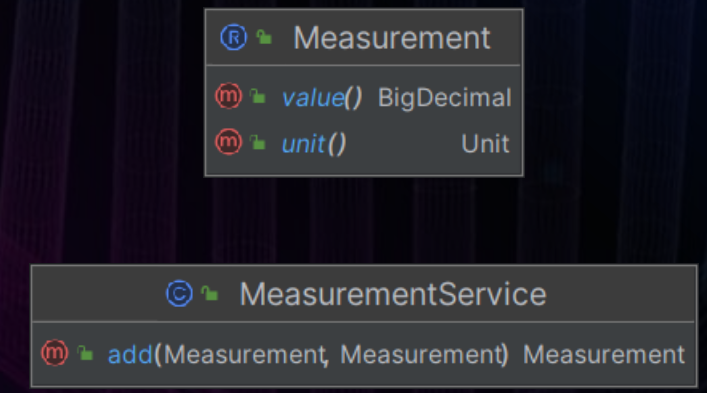
- We need to implement adding functionality



GRASP

„We are drowning in information but starved for knowledge“
by John Naisbitt

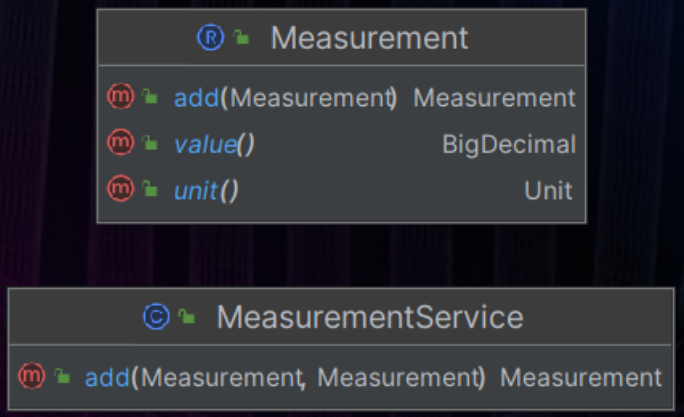
- We need to implement adding functionality
  - Separate service



GRASP

„We are drowning in information but starved for knowledge“
by John Naisbitt

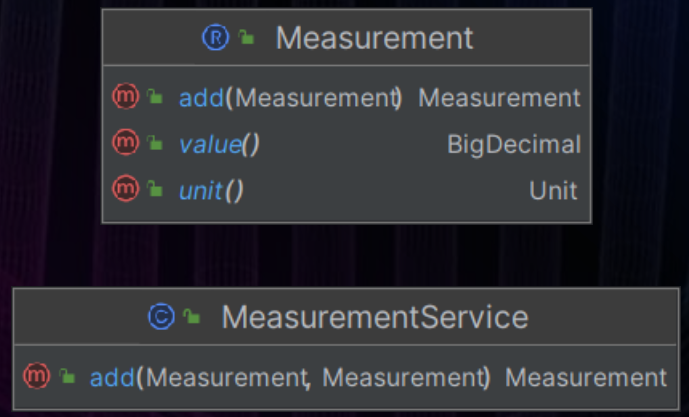
- We need to implement adding functionality
  - Separate service
  - `add()` method in `Measurement`



GRASP

„We are drowning in information but starved for knowledge“
by John Naisbitt

- We need to implement adding functionality
  - Separate service
    - Flexibility in terms of adding new complex operations (e.g. unit conversion)
    - Requires additional conventions to quickly find available operations
  - `add()` method in `Measurement`
    - fluent API – leads you by hand
    - `DTO` hell

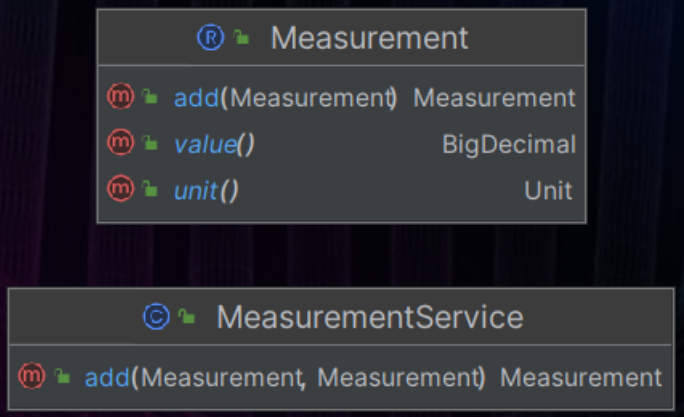


GRASP

„We are drowning in information but starved for knowledge“
by John Naisbitt

Information expert

- How to assign responsibilities?
- Assign it to the class that has all the information to fulfill it



GRASP

„I'm the creator of my reality"
by Annette Funicello

Creator

- Who should create object `Desert`?

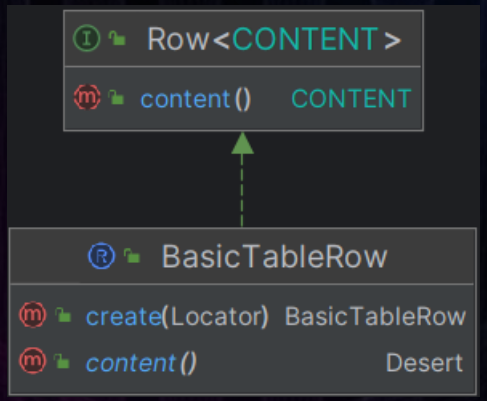
Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)
Frozen yoghurt	159	6	24	4
Ice cream sandwich	237	9	37	4.3
Eclair	262	16	24	6
Cupcake	305	3.7	67	4.3
Gingerbread	356	16	49	3.9

GRASP

„I'm the creator of my reality"
by Annette Funicello

Creator

- Who should create object A?
- The one who:
  - Compositely aggregates objects of type A
  - Closely uses objects of type A
  - Have initializing information for type A



GRASP

„In all fighting (...) indirect methods will be neeeded in order to secure victory“
by Sun Tzu

Indirection

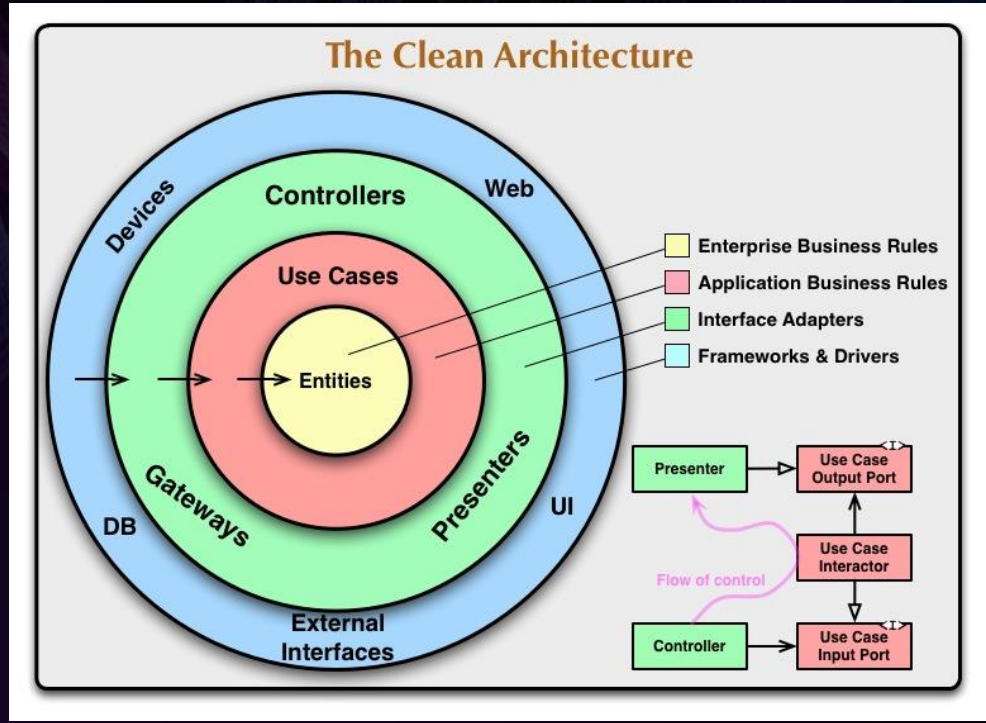
- How not to couple objects?
- Design object responsible for handling interactions between other objects

GRASP

„What consumes your mind, controls your life“
by Anonymous

Controller

- Who should be responsible for handling an input system event?



GRASP

„Who plays together stays together“
by Anonymous

Low Coupling

- How much two components <x> each other
  - are connected to
  - rely on
  - know about
  - depend on

- Coupling levels
  - **Content coupling (e.g. I know your internals)**
  - **Temporal coupling (e.g. order matters)**
  - **Global data (e.g. static)**
  - **Control (e.g. pass boolean flag to change behavior)**
  - **Stamp (e.g. entire data structure is shared)**
  - **Data (e.g. share data throu parameters)**
  - **Message (e.g. pub-sub)**
  - **No coupling**

GRASP

„Who plays together stays together“
by Anonymous

Low Coupling

$$C = 1 - \frac{1}{d_i + 2 \times c_i + d_o + 2 \times c_o + g_d + 2 \times g_c + w + r}$$

- d_i - #input params
- c_i - #input control params
- d_o - #output params
- c_o - #output params
- g_d - #global data vars
- g_c - #global control vars
- w - #fan-out
- r - #fan-in

- Coupling levels
  - **Content coupling (e.g. I know your internals)**
  - **Temporal coupling (e.g. order matters)**
  - **Global data (e.g. static)**
  - **Control (e.g. pass boolean flag to change behavior)**
  - **Stamp (e.g. entire data structure is shared)**
  - **Data (e.g. share data throu parameters)**
  - **Message (e.g. pub-sub)**
  - **No coupling**

GRASP

„Who plays together stays together“
by Anonymous

High cohesion

- How much elements inside module belong together?
- How much methods belong to fields?

- Coupling levels
- **Coincidental (e.g. utils class)**
- **Logical (e.g. all controllers in one dir)**
- **Temporal (e.g. ex handling)**
- **Procedural (e.g. there is a certain procedure to follow)**
- **Informational (e.g. operate on the same data)**
- **Sequentional (e.g. output -> input)**
- **Functional (e.g. all is required to achieve goal)**
- **Perfect (atomic)**

GRASP

„Everything morphs, nothing stands still“
by Heraclitus (almost)

Polymorphism

- Variations in behaviou should be placed in child classes

GRASP

„Protection ans security are only valuable if they don‘t cramp life excessively“
by Carl Jung

Protected variations

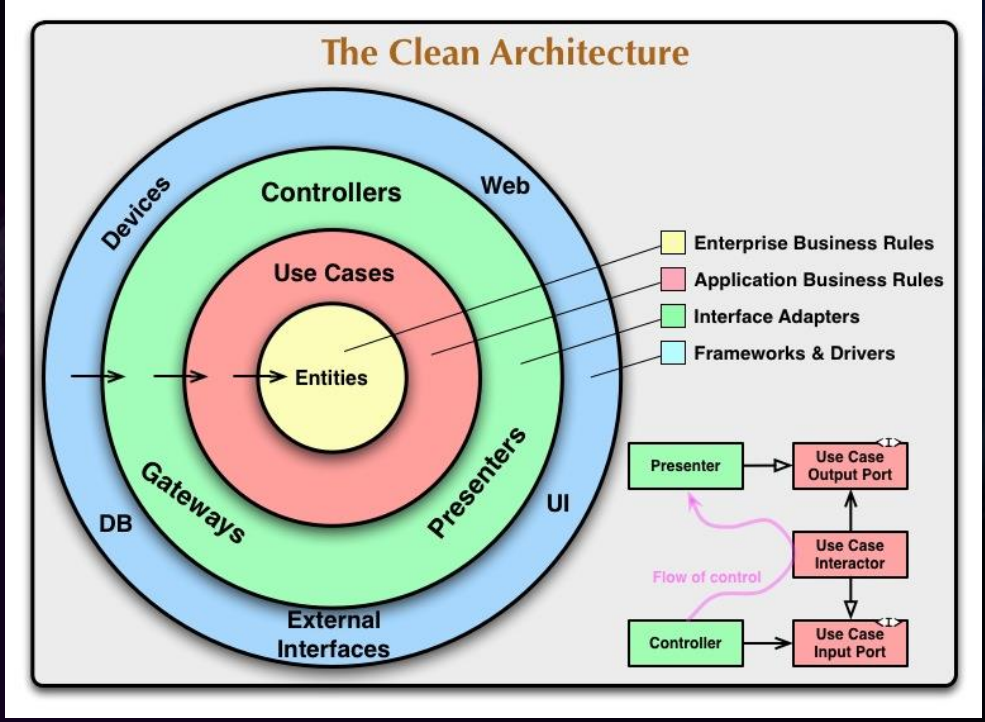
- Wrap instability of the world into an interface and use polymorphism

GRASP

„Don't blame the world. Find a solution."
by Sri Chinmoy

Pure fabrication

- A class „fabricated" for the sake of the solution, not derived from problem space



CUPID

„Forever is composed of nows“ by Emily Dickinson

Composable

- Small surface area
- Intention revealing
- Minimal dependencies
- „Fits in your head”

CUPID

„Efforts and courage are not enough without purpose and direction“
by J.F Kennedy

UNIX philosophy

- Do one thing and do it well
- Single purpose vs responsibility

CUPID

„The best way to predict the future is to create it“
by A. Lincoln

Predictable

- The principle of least astonishment
- No side effects
- Repeatability / Reliability
- Observable

CUPID

„There should be one – and preferably only one – obvious way to do it“
by The Zen of Python

Idiomatic

- Empathy
- Code format / Dir structure
- „There should be one (...) obvious way to do it”

CUPID

„The biggest problem in communication is the illusion that it has taken place“
by G. B. Shaw

Domain based

- Ubiquituous language
- Domain driven dir structure (Clean architecture)

Thank you

Remigiusz Dudek

remigiusz.dudek@gmail.com