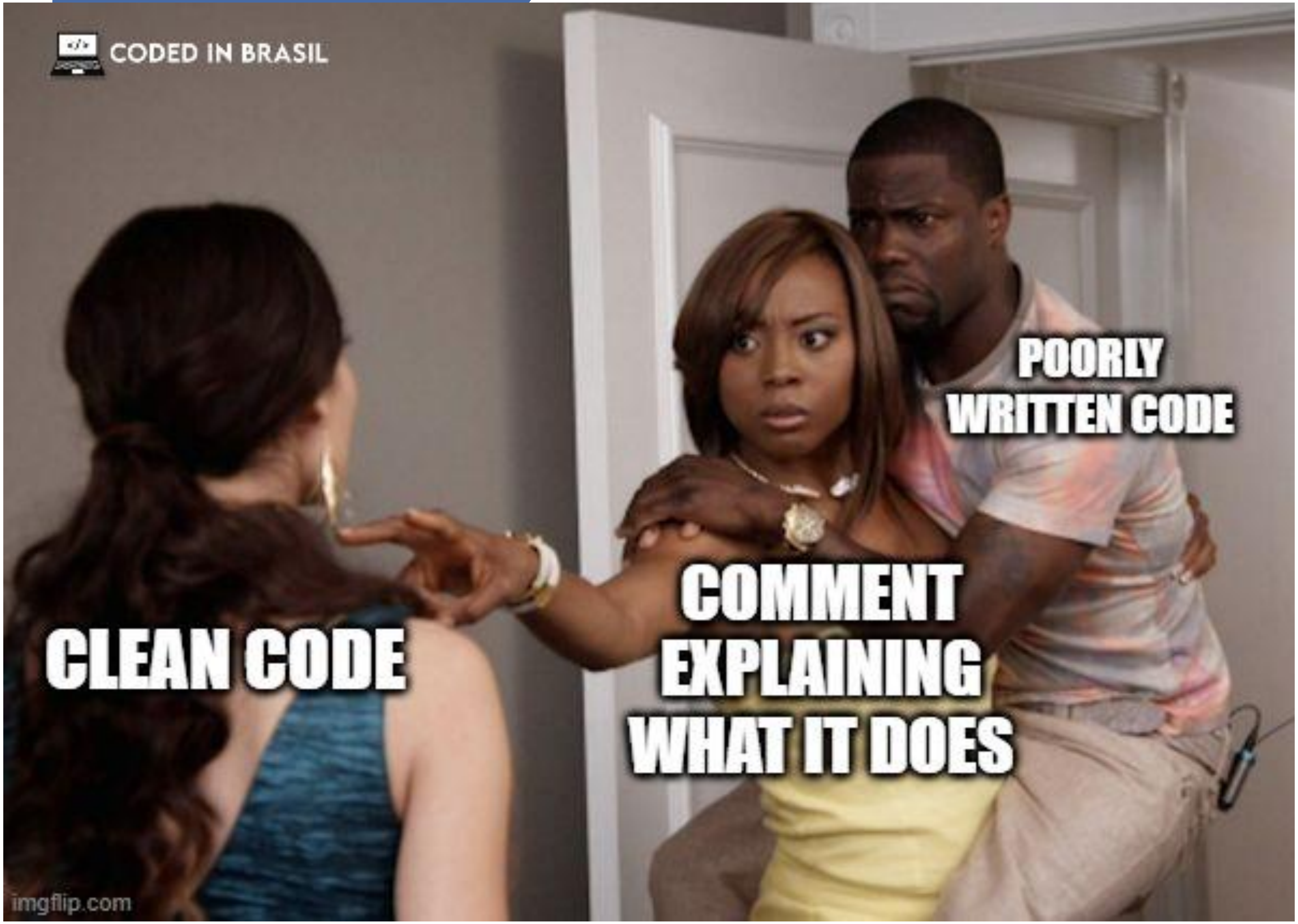
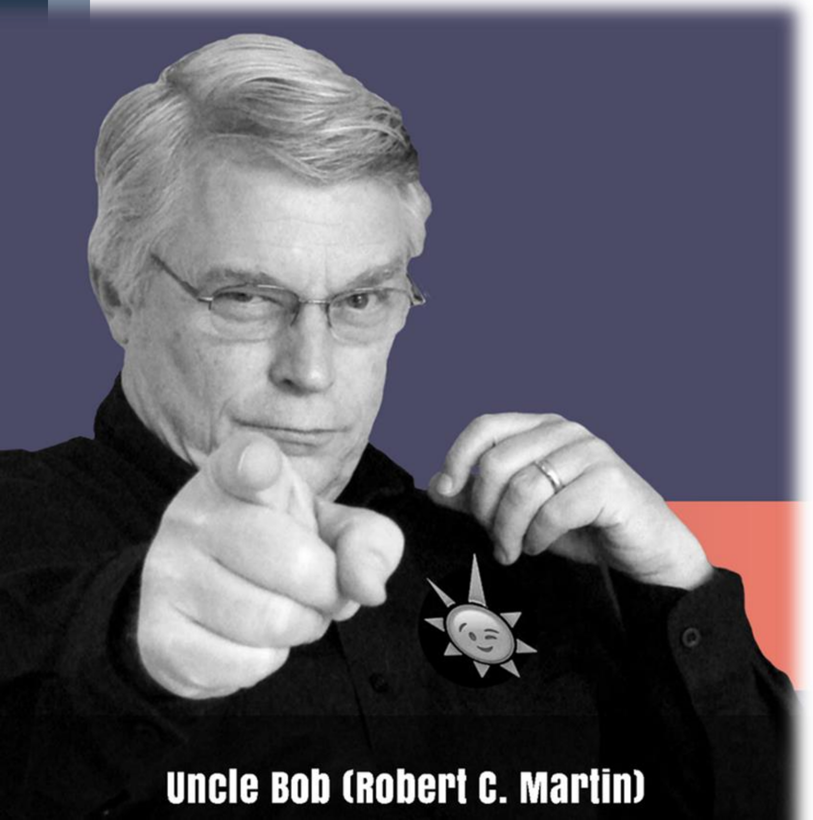
CLEAN architecture

Paolo Burgio
paolo.burgio@unimore.it


CODED IN BRASIL
POORLY WRITTEN CODE
COMMENT EXPLAINING WHAT IT DOES
CLEAN CODE
imgflip.com

What is it?

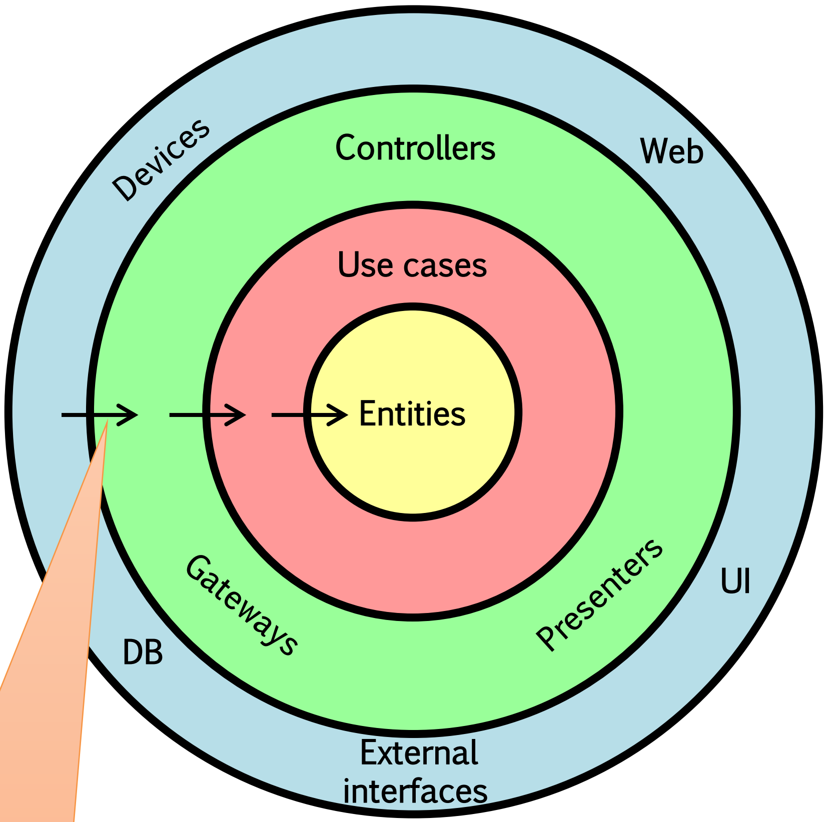
- A structure that enables building software that is more scalable, testable, maintainable
- Built upon/heavily relies on good coding practices (e.g., SOLID, design patterns..)
- Disclaimer: +15-20% dev time overhead



- “Uncle Bob” started his blog in 2011

As simple as this

- Aka: “Onion Architecture”



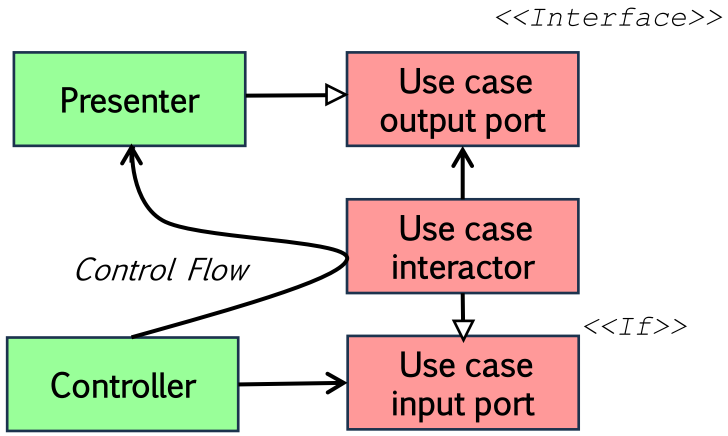
Dependencies go from “out” to “in”

Enterprise business rule

Application business rule

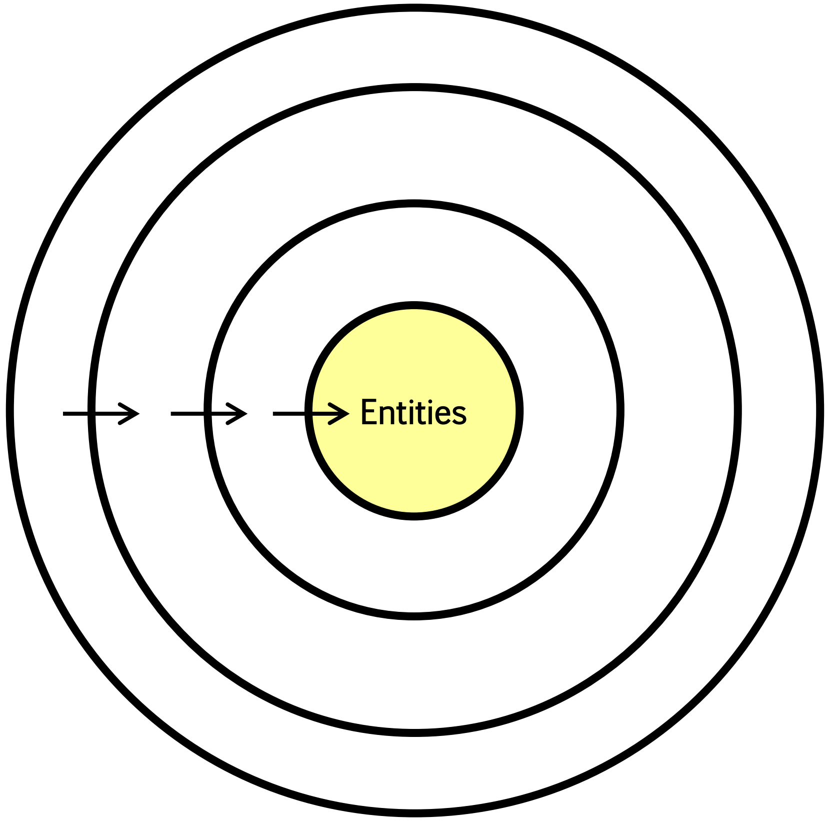
Interface Adapters

Frameworks & Drivers



The Model

› Our view of the world: just field, and basic operations (get, set..)



Enterprise business rule

› Everything depends on them/includes them, they do not depend on anything

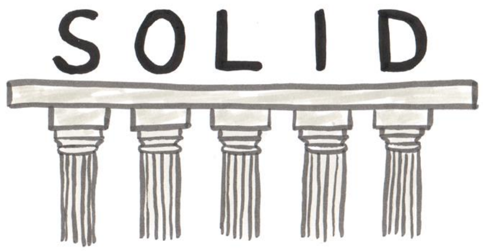
› Why is this so important?

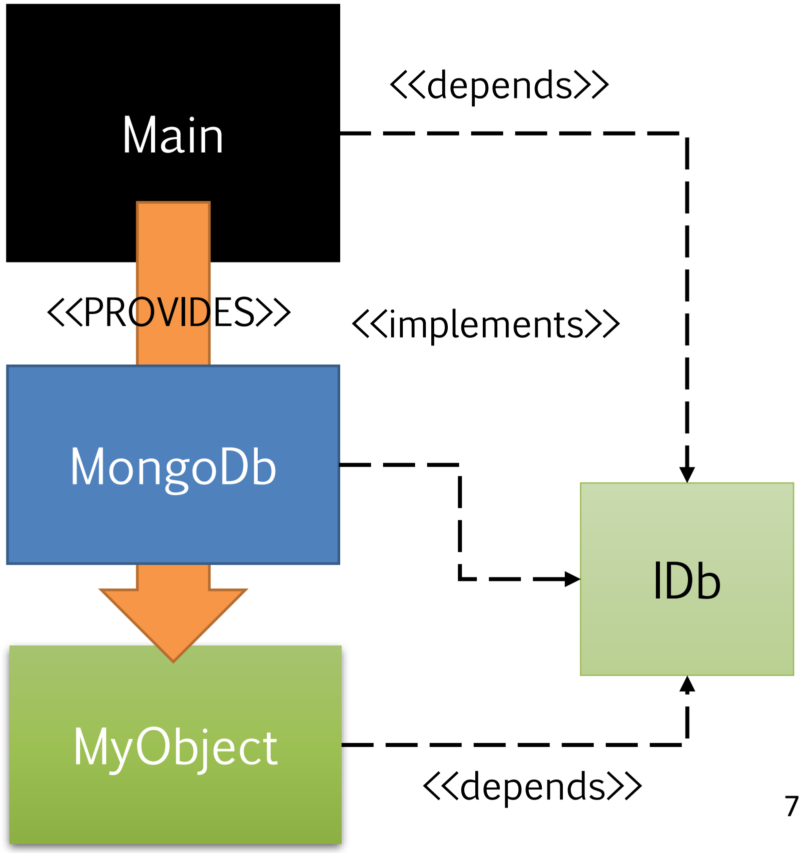
Dependency Inversion

- Reduce coupling
  - Avoids unnecessary dependencies that ultimately make the code hard to modify
- Enables fast testing and debugging
- Wraps functionalities (Interface Segregation)

(Only one issue)

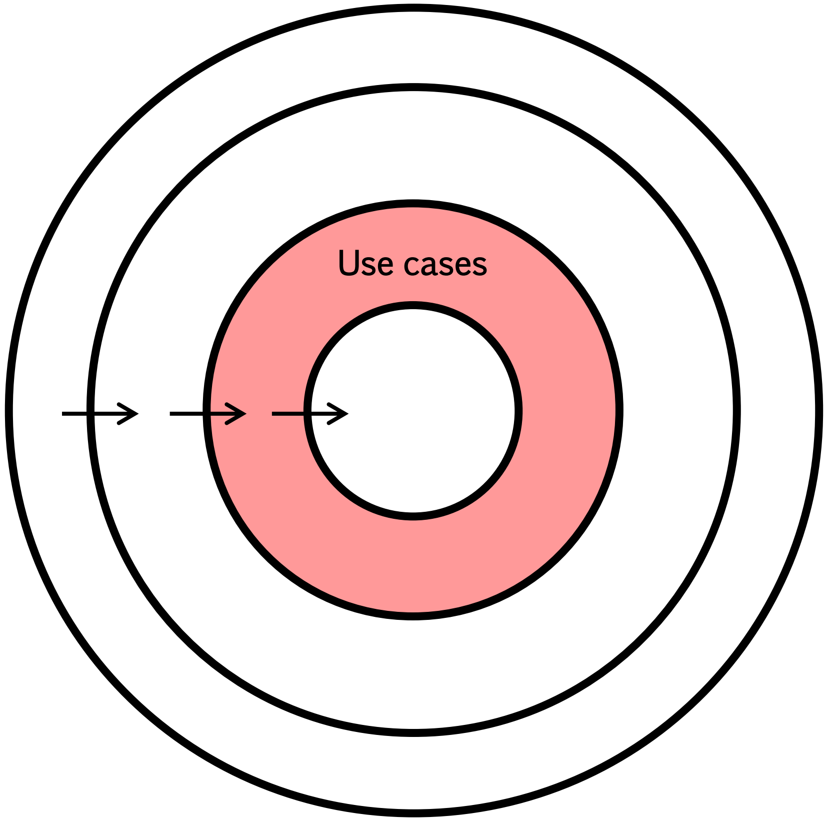
- You need to find a (elegant) way to provide the required services
- Dependency Injection!





Straight from requirements

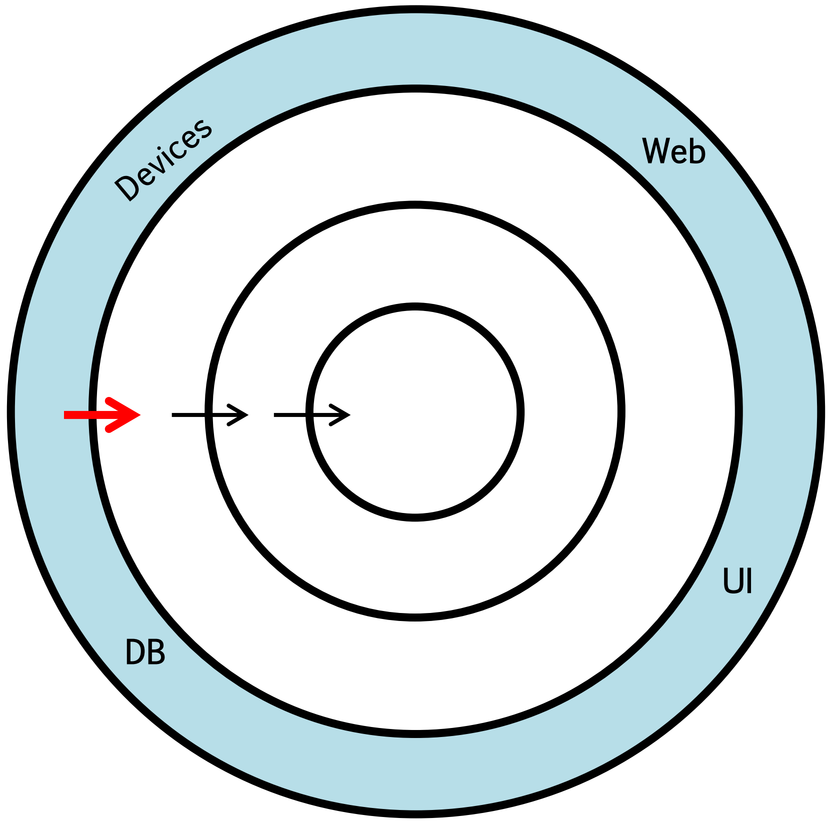
› Application specific logics: functionalities



Application business rule

"The bad world"

- This layer represents, and wraps, "external" dependencies, e.g., DTOs, MongoDb...

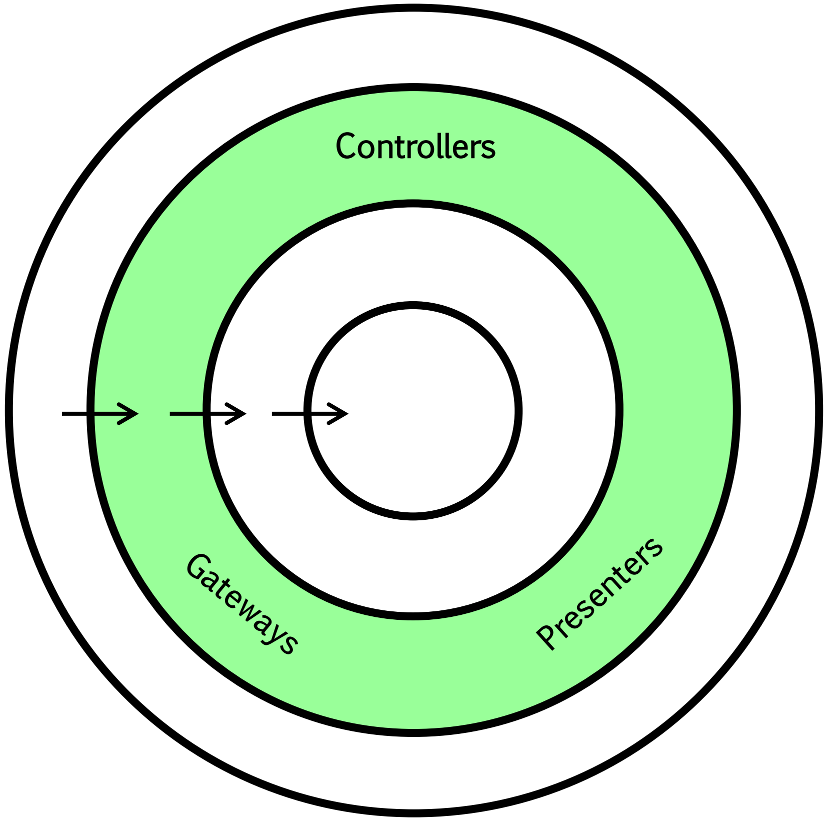


Frameworks & Drivers

- How do we implement the dependency?

Our good old frient

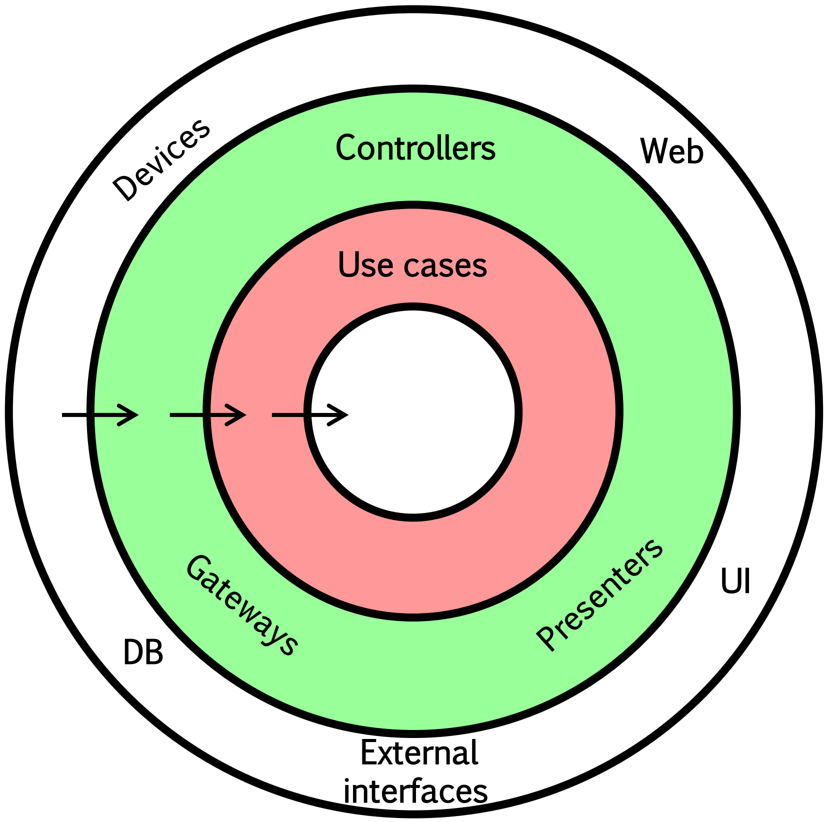
- Aka: “Onion Architecture”



Interface Adapters

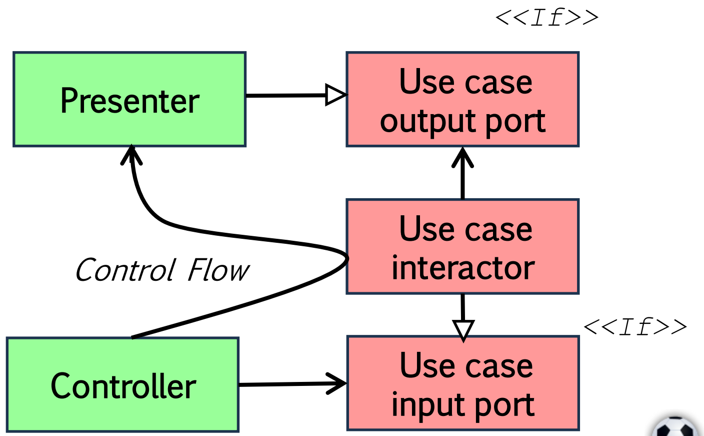
Control flow, and class diagram

› Note how we use Interfaces, and (consequently) Dependency Injection



Application business rule

Interface Adapters



CLEAN, in practice

Take any “basic” application, and refactor it following the clean architecture

1. Refactor the basic example of C# WebApi

```
$ dotnet new webapi --use-controllers [-o MyApi]
```

Use dependentcy injection with builder.Services.Add in "

2. Take the basic example of a Java webapi app, and refactor it

3. Take any code that you wrote

Dependency Injection in dotNet

Example: WebApp

- We build and run the actual program, explicitly, in `Program.cs`
- `WebApplicationBuilder` is the class that performs (Web)Application startup
- It has features to inject services

```
// 'Scoped' means that you create a new instance every time
// it is injected
builder.Services.AddTransient<IService, ConcreteImplementation>();

// 'Transient' services are created only once for every HTTP request
// we are serving (hence, useful for keeping states within a request
builder.Services.AddScoped<IService, ConcreteImplementation>();

// ...
builder.Services.AddSingleton<IService, ConcreteImplementation>();
```

Dependency Injection in Java

- TODO

References

Course website

- http://hipert.unimore.it/people/paolob/pub/ProgSW/index.html

Uncle Bob

- https://blog.cleancoder.com/uncle-bob/2011/11/22/Clean-Architecture.html

My contacts

- paolo.burgio@unimore.it
- http://hipert.mat.unimore.it/people/paolob/