



DOMAIN
DRIVEN
DESIGN
EUROPE

DEVS, LET'S RECLAIM DDD!

Arnaud THIEFAINE

Dorra BARTAGUIZ

WHY SHOULD WE LEARN DDD?

It's so stylish!

I want to do like
those who talk
about it.

For an awesome
resume!

It might be useful at
the end!

DDD
BUZZ WORD?



WHO ARE WE?

Arnaud THIEFAINE

Coach / trainer

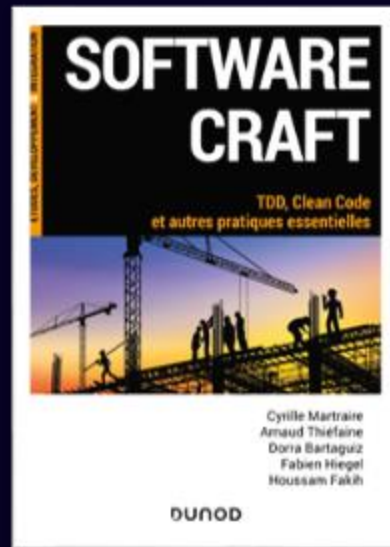
Co-author « Software Craft »



ArnaudThiefaine



athiefaine



Dorra BARTAGUIZ

VP Tech @Arolla

Co-author & illustrator « Software Craft »



DorraBartaguiz



iAmDorra

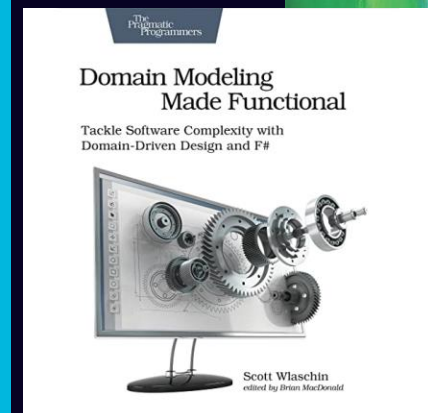
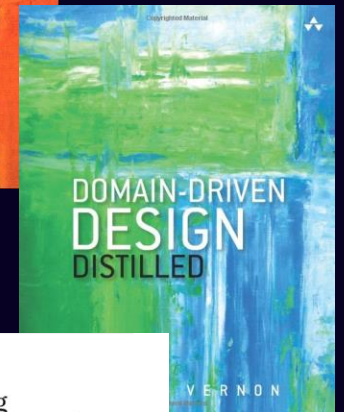
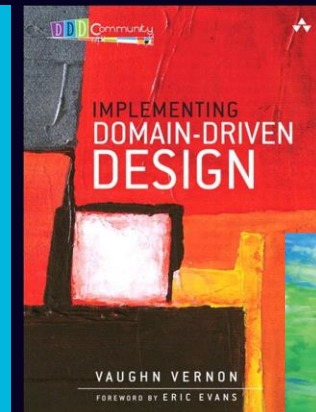
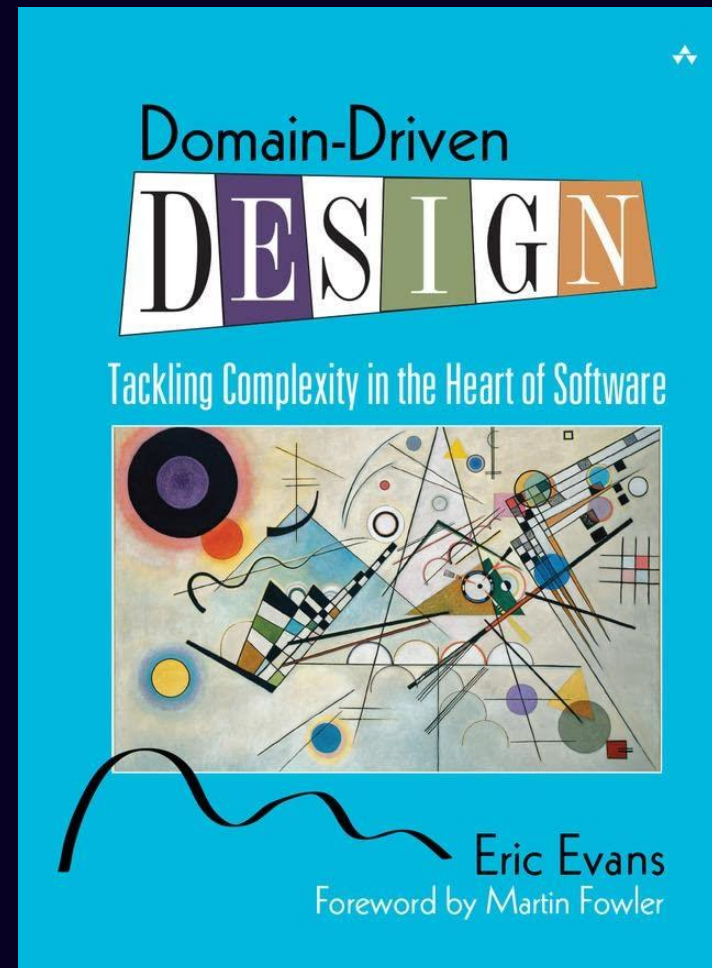


<https://www.arolla.fr/training/>

DOMAIN DRIVEN DESIGN



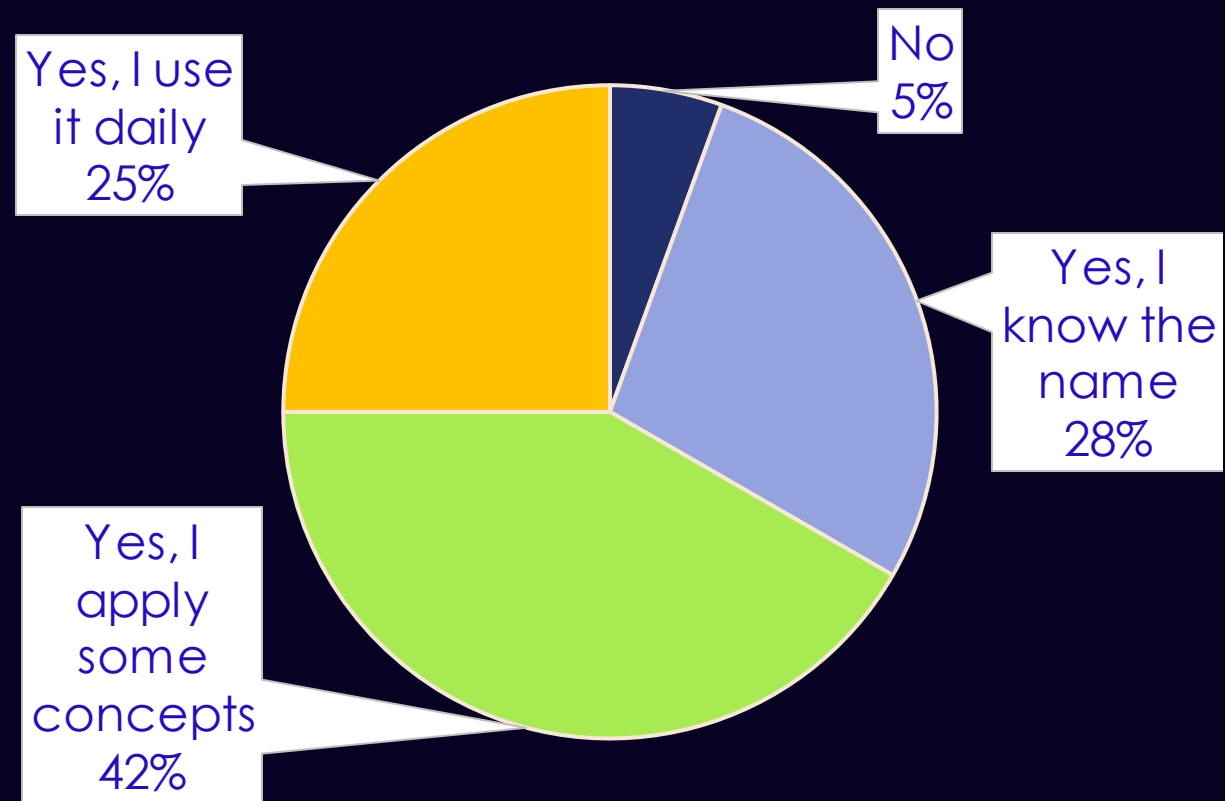
20 years



SURVEY



**Do you know
DDD?**





~56%

Appreciate DDD

~63%

Find it difficult to apply



Votre rapport à DDD - Domain Driven Design

Durée estimée d'exécution : 10 min

1. Est-ce que vous connaissez DDD - Domain Driven Design ?

- ☐ non
- ☐ oui, mais de nom
- ☐ oui, j'applique quelques concepts
- ☐ oui, je suis à fond dedans

2. A quel point vous appréciez DDD ?

☆ ☆ ☆ ☆

3. A quel point vous trouvez que c'est difficile à appliquer ?

☆ ☆ ☆ ☆

WHAT WE REPROACH THE DDD?

I am not an architect,
so I did not have the
opportunity to dig

I do not have access
to the organization's
strategic discussions

The organization
and the teams are
too hermetic to
make it

It is mainly used to
make bullshit
diagrams

It does not go back
to the code

SUMMARY



- Modelling Kata (Theater Kata)
- Value Object to rescue primitive-obsessions
- Business and technic separation
 - Sandwich pattern
 - Hexagonal architecture
- Bounded Contexts identification
- Protect invariants
 - Aggregate
 - Entity
 - Value Object
- Service & repository
- Separate Bounded Contexts

THEATER KATA



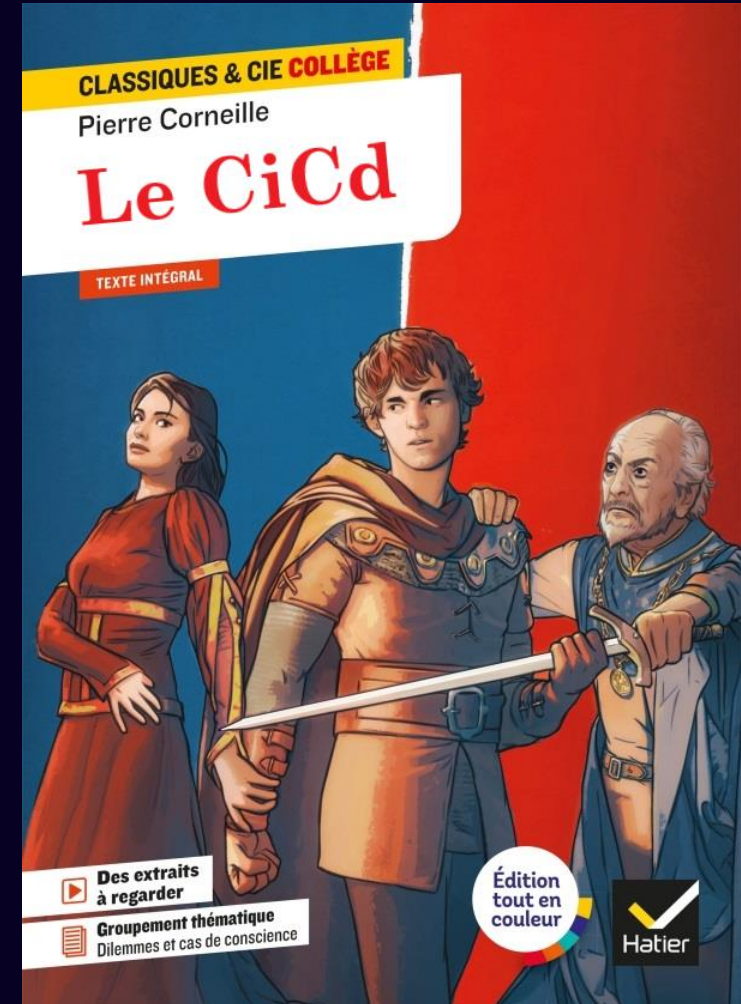
Beatriz « The CICD »



50% VIP



35€ standard category
+50% for premium





THEATER KATA

Alix books 4 seats 

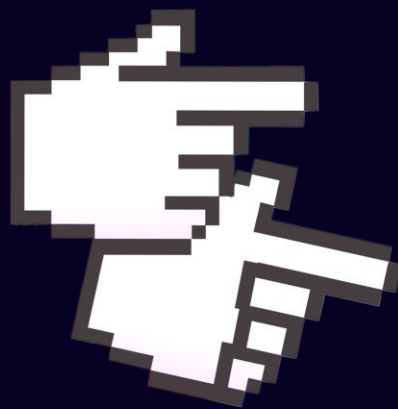
 Premium category

Side-by-side seats 

 Loyalty card

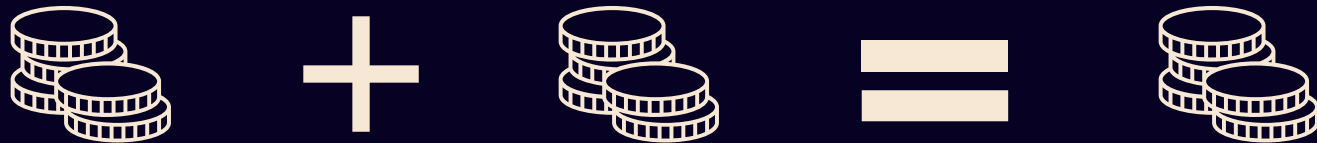
 Tickets

@ArnaudThiefaine @DorraBartaguiz



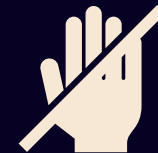
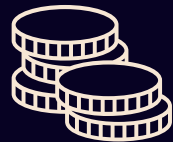
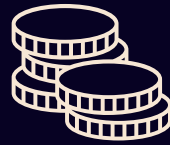
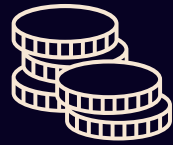
LET'S CODE

PRICING ARITHMETICS

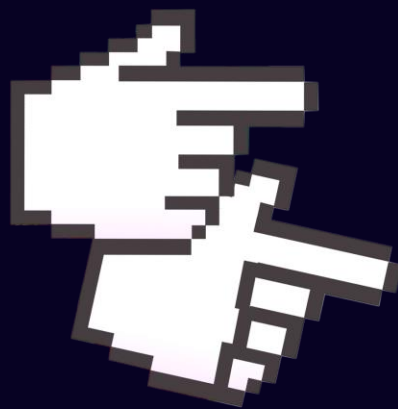


DSL

PRICING ARITHMETICS



@ArnaudThiefaine @DorraBartaguiz

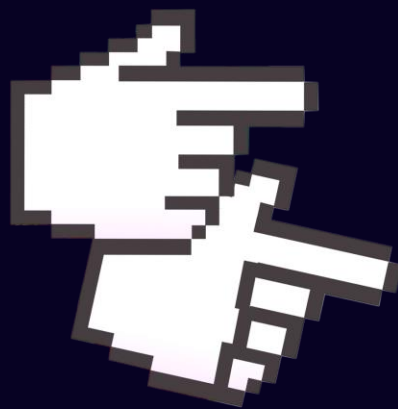


LET'S CODE

PATTERN SANDWICH



@ArnaudThiefaine @DorraBartaguiz



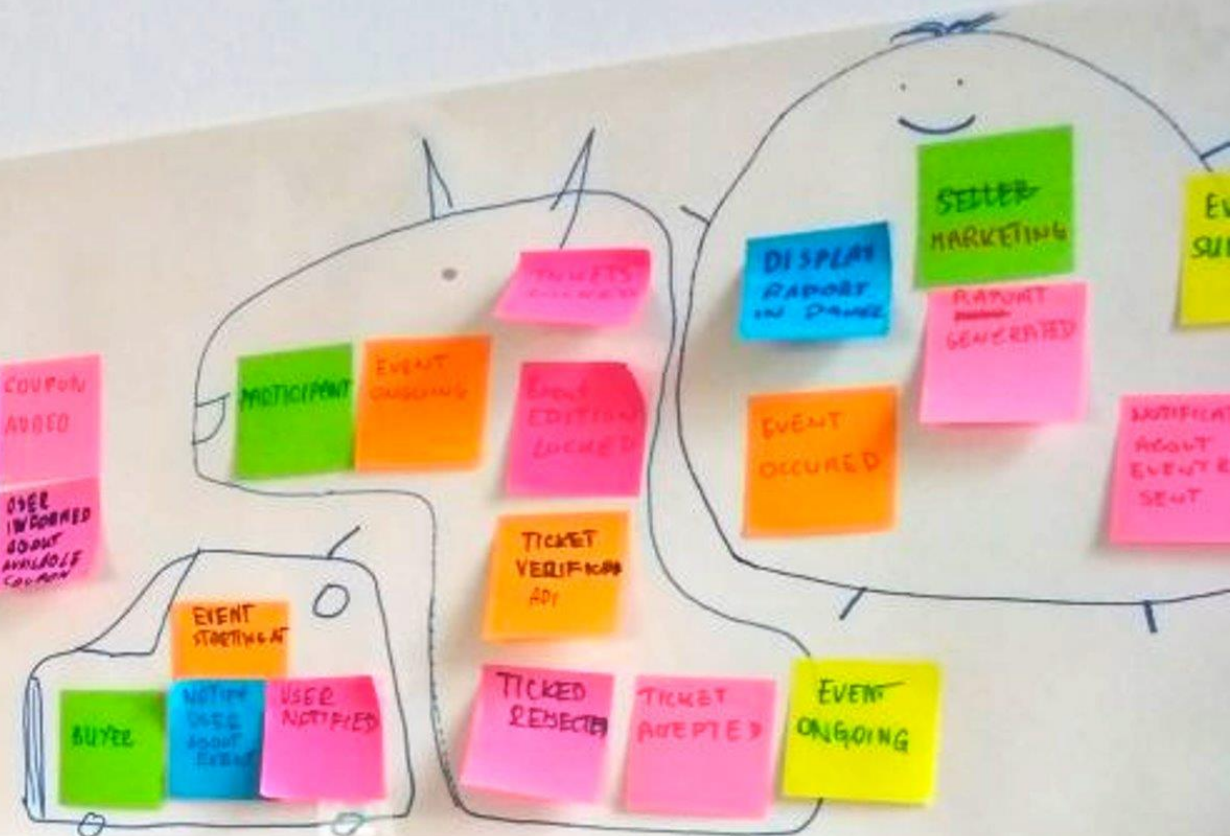
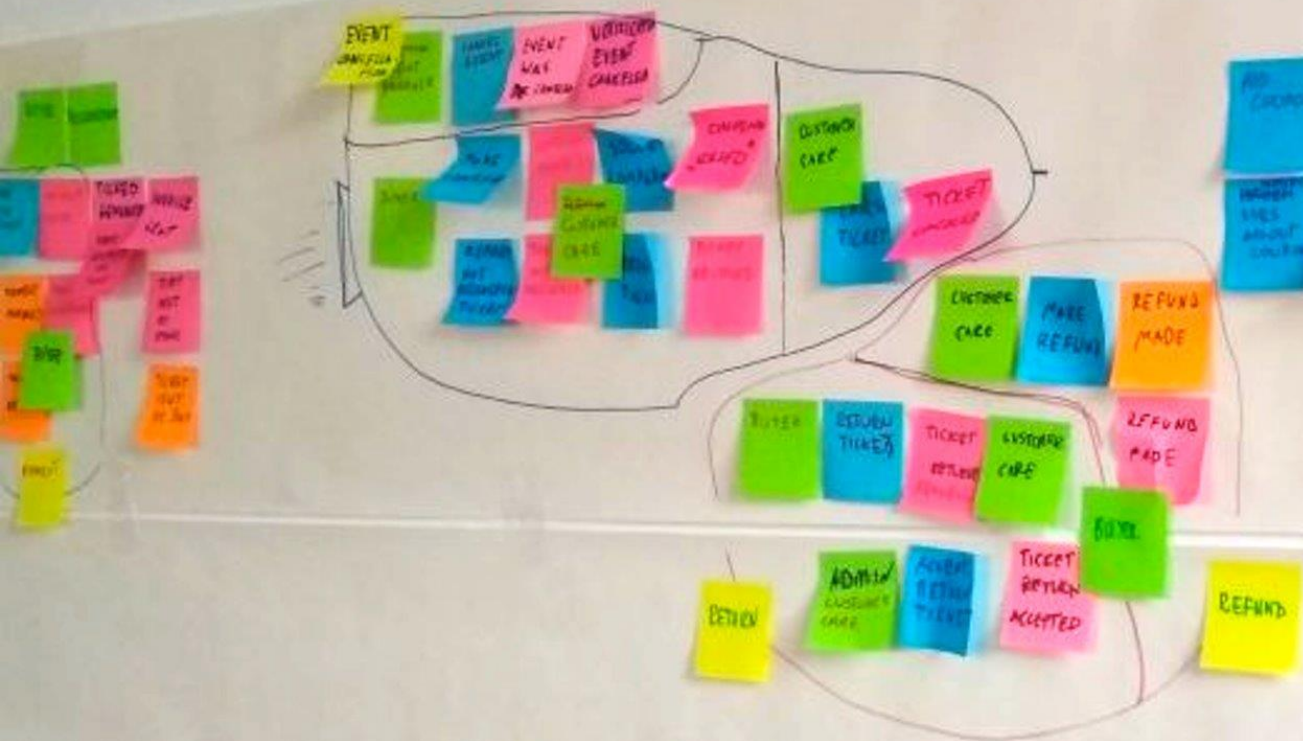
LET'S CODE

IDENTIFICATION OF DIFFERENTS BUSINESS/DOMAINS

Aka Bounded Contexts

Event Storming workshops

@ArnaudThiefaine @DorraBartaguiz





SJD & ING

Seat
reservation

Marketing

Performance
scheduling

Pricing

Ticketing



SJD & ING

Seat
reservation

Marketing

Performance
scheduling

Pricing

Ticketing



PERSONAE

Beatriz, actress

Julia,
marketing

Alix, client

Director

Terminal

SYNONYMS

Entry
reservation

Premium
armchair

Seat layout

Marketing

Performance
scheduling

Pricing

Ticket
printing



@romeu@mastodon.social

@malk_zameth

In the culinary bounded context: 🍅 is a vegetable

In the botanic bounded context: 🍅 is a fruit

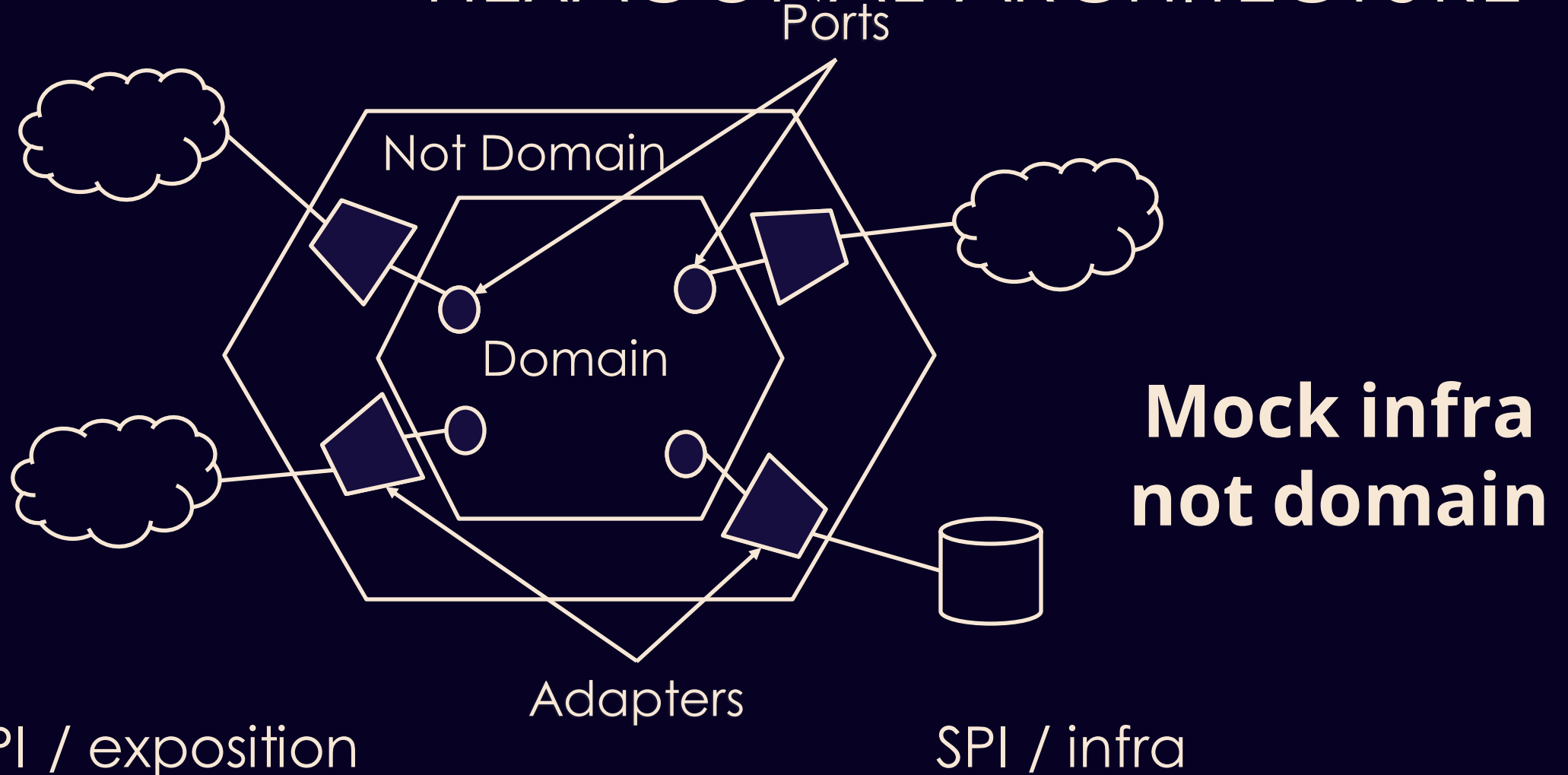
In 🤔 bounded context: 🍅 is feedback

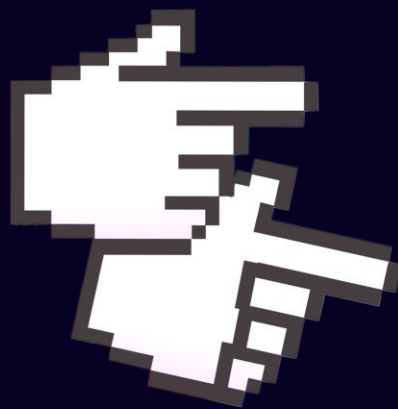
2:17 PM · 10 nov. 2017

BC IDENTIFICATION HINTS

- “/t/s/ion” sjõ
- “ing”
- Different personae
- Synonyms
- Words with multiple meanings

HEXAGONAL ARCHITECTURE

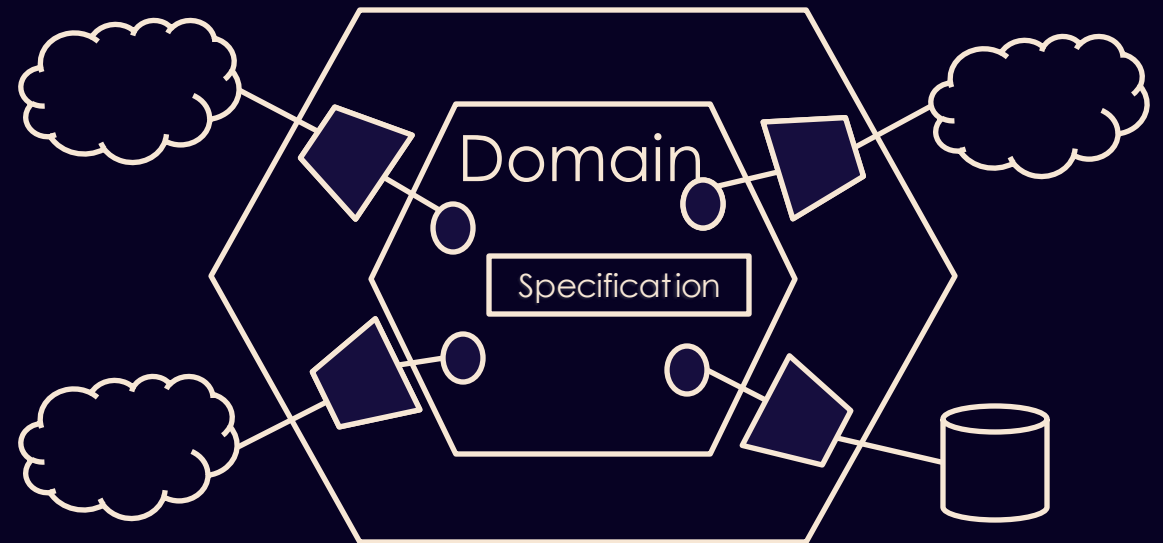




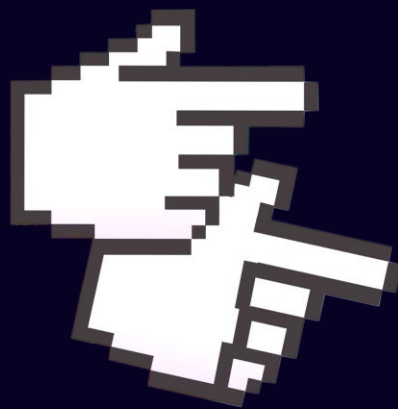
LET'S CODE

SPECIFICATION

- Domain Invariant : predicat
- Validate a domain objet using isSatisfiedBy()
- Model a rule
- Configurable



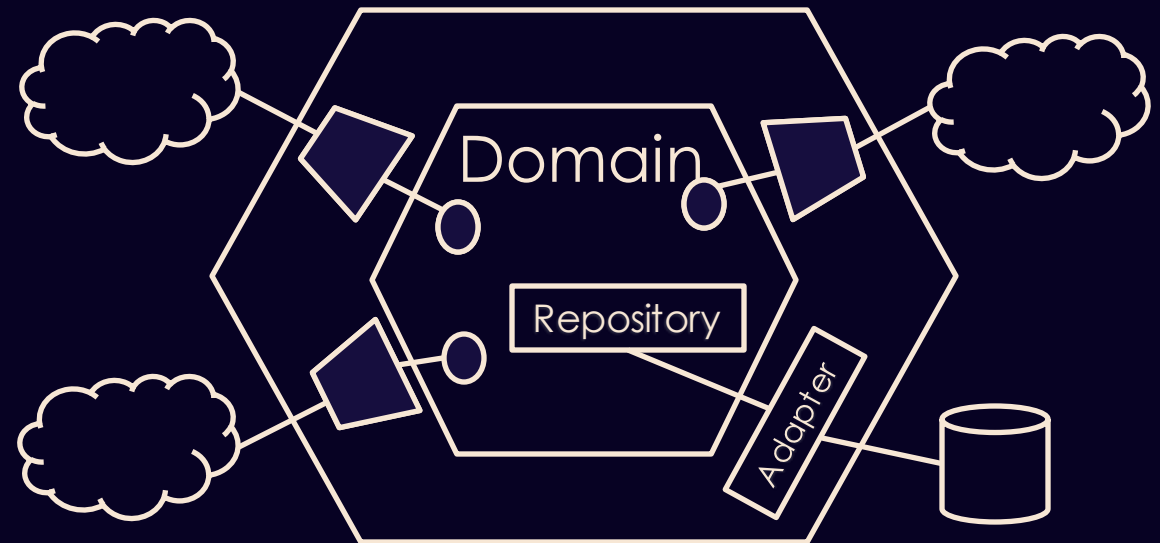
@ArnaudThiefaine @DorraBartaguiz



LET'S CODE

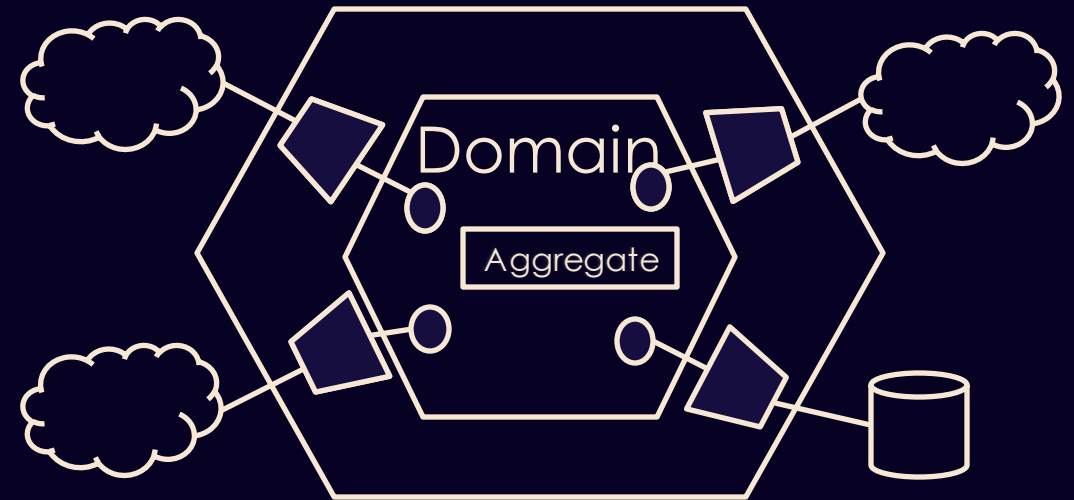
REPOSITORY

- Collection-oriented
- Manage transaction
 - in repo not the service
- Side Effect
- Injectable
- Ubiquitous language
 - For naming

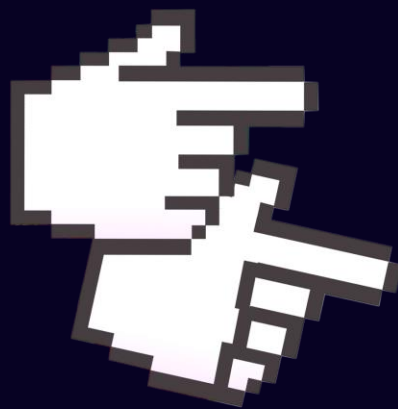


AGGREGATE

- Unique identity for the grape
- Aggregate Root
- Invariant to protect
- Consistence
 - validity
 - transactional (save/load)



@ArnaudThiefaine @DorraBartaguiz



LET'S CODE

Performance

Id

play

startDate

endDate

performanceNature

Topology

Reservation

Allocation

Performance

Id
play
startDate
endDate
performanceNature

TheaterSession

titre
startDate

Reservation

Topology

PerformanceNature

value

Allocation

Value object

No identity

Value equality

Immutable

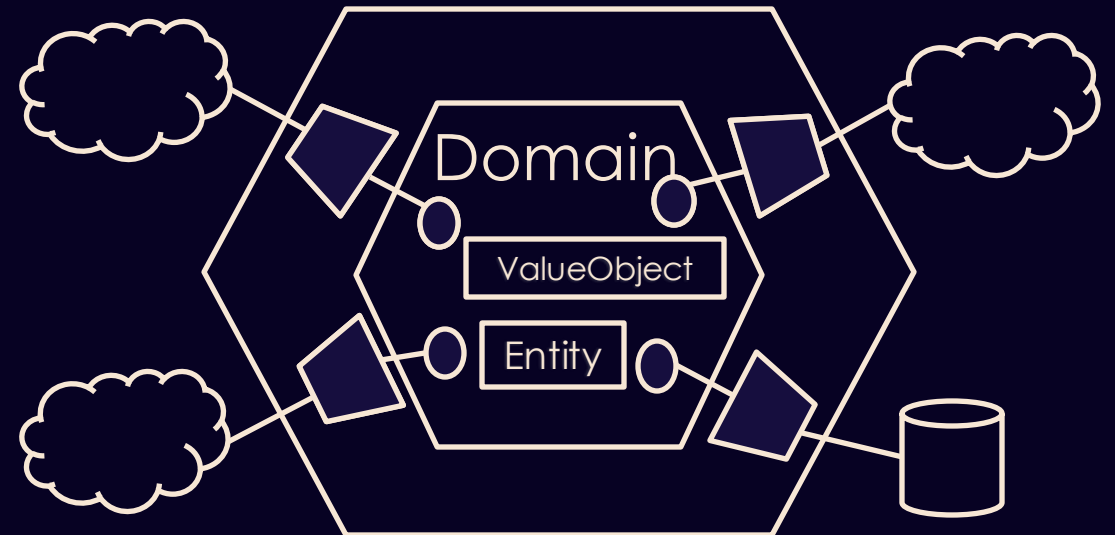
Parametrized Constructor
Factory method

Side-effect-free

Entity

Unique identity

Evolves over time



**Value
object**

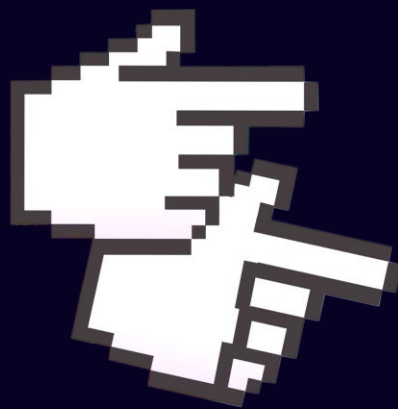


Entity



**Empty /
filled**

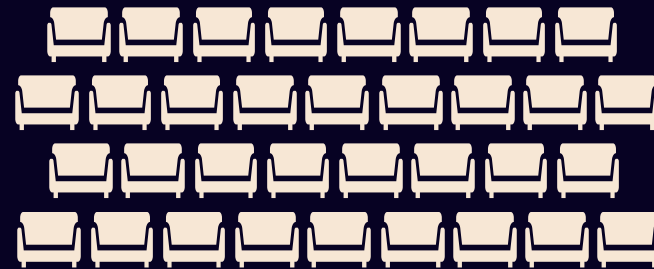
Identity



LET'S CODE

THEATER TOPOLOGY

Central zone
(standard)



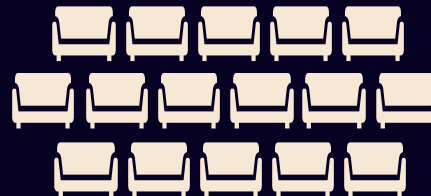
Row A

Row B

Row C

Row D

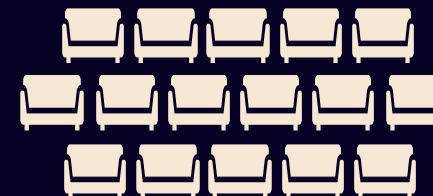
Row E



Row F

Row G

Balcony 1
(premium)



Row I

Row J

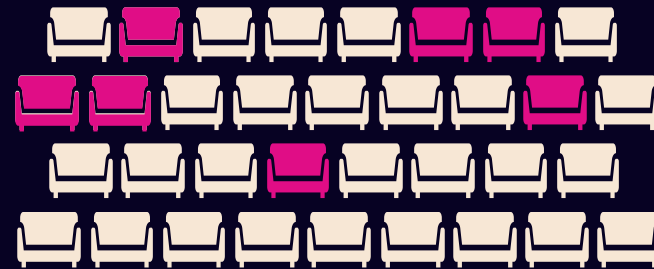
Row K

Balcony 2
(premium)

Lea needs to
reserve 4 seats

THEATER TOPOLOGY

Central zone
(standard)



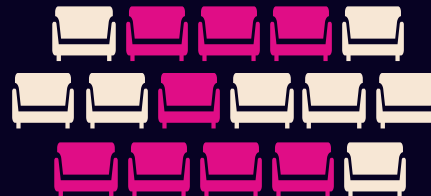
Row A

Row B

Row C

Row D

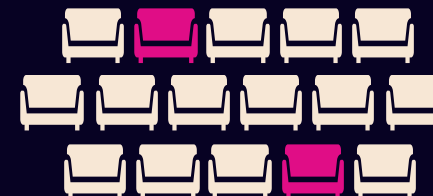
Row E



Row F

Row G

Balcony 1
(premium)



Row I

Row J

Row K

Balcony 2
(premium)

THEATER TOPOLOGY

Central zone
(standard)



Row A

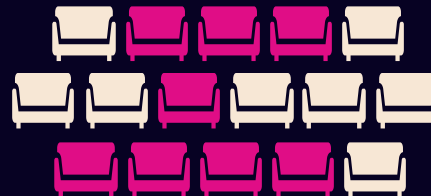
Row B

Row C

Row D

Lea gets seats
B3, B4, B5 and B6

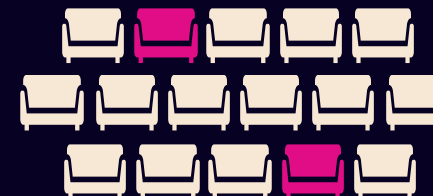
Row E



Row F

Row G

Balcony 1
(premium)



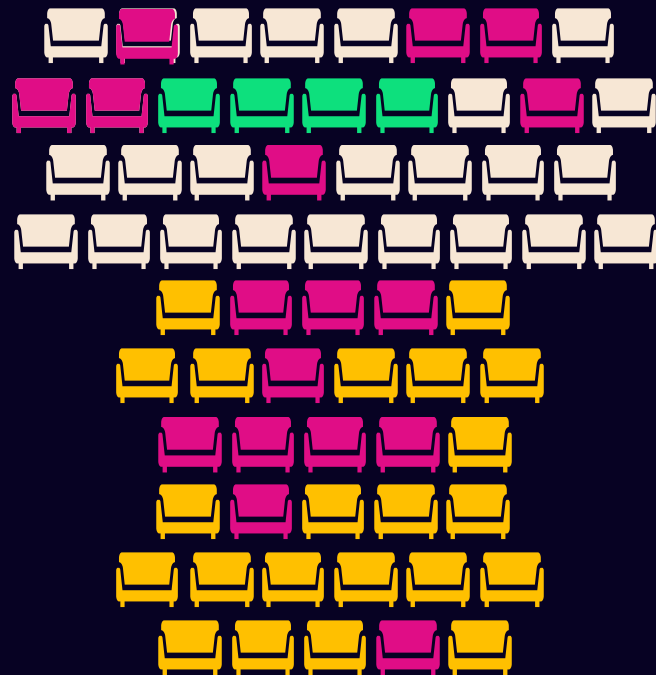
Row I

Row J

Row K

Balcony 2
(premium)

SEAT ALLOCATION



Row A

Row B

Row C

Row D

Row E

Row F

Row G

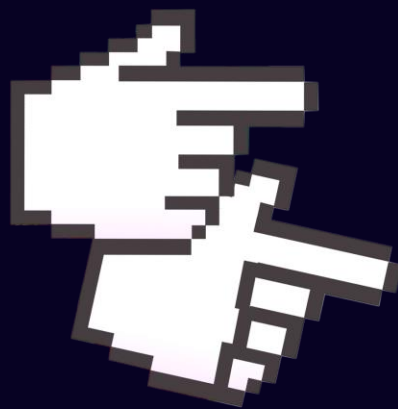
Row I

Row J

Row K

No more Zones

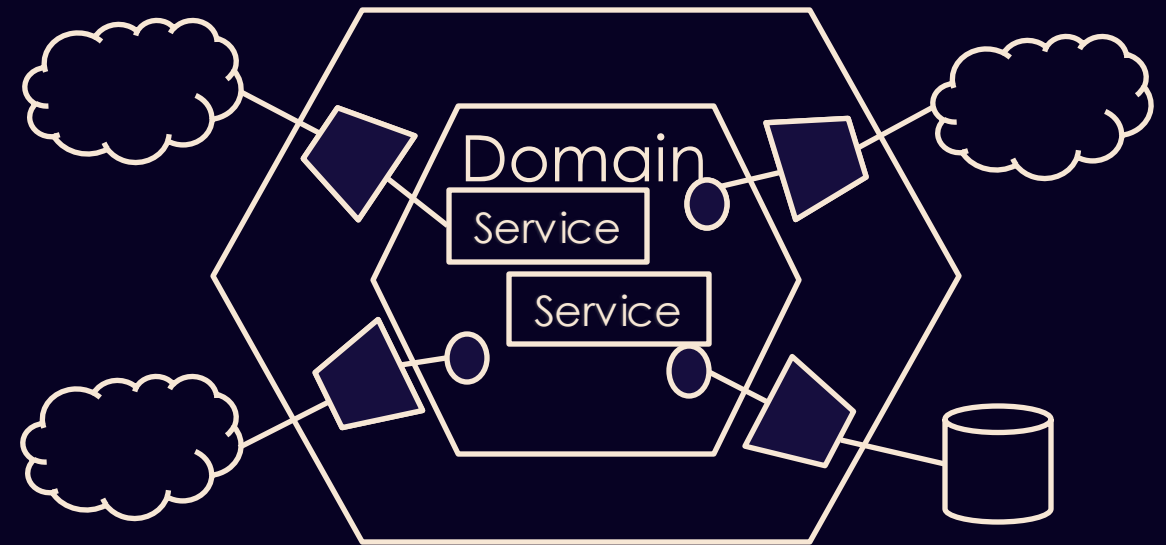
@ArnaudThiefaine @DorraBartaguiz




LET'S CODE

SERVICE

- Stateless
- Injectable
- Business logic
 - Neither in Value Object
 - Nor in Entity
- Ubiquitous language
 - For naming

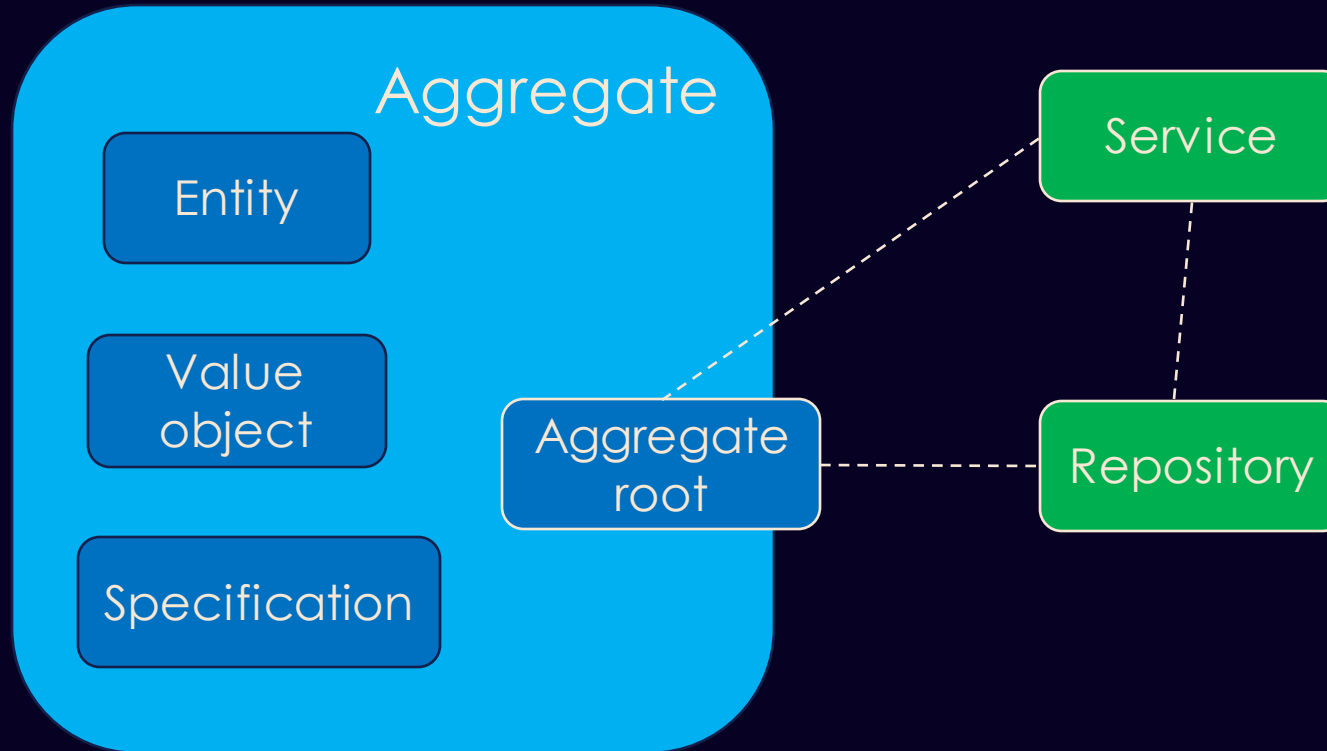




““ Make sure **you need** a service. [...] Using Services overzealously will usually result in the negative consequences of creating an **Anemic Domain Model**.

- Vaughn Vernon
Implementing Domain-Driven Design

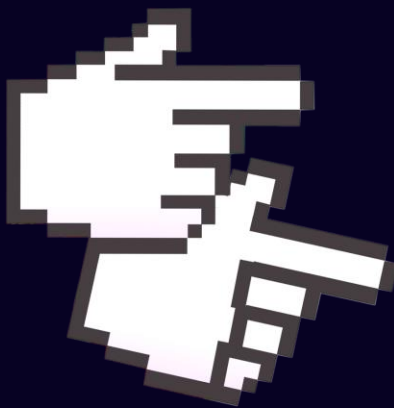
TO SUM UP



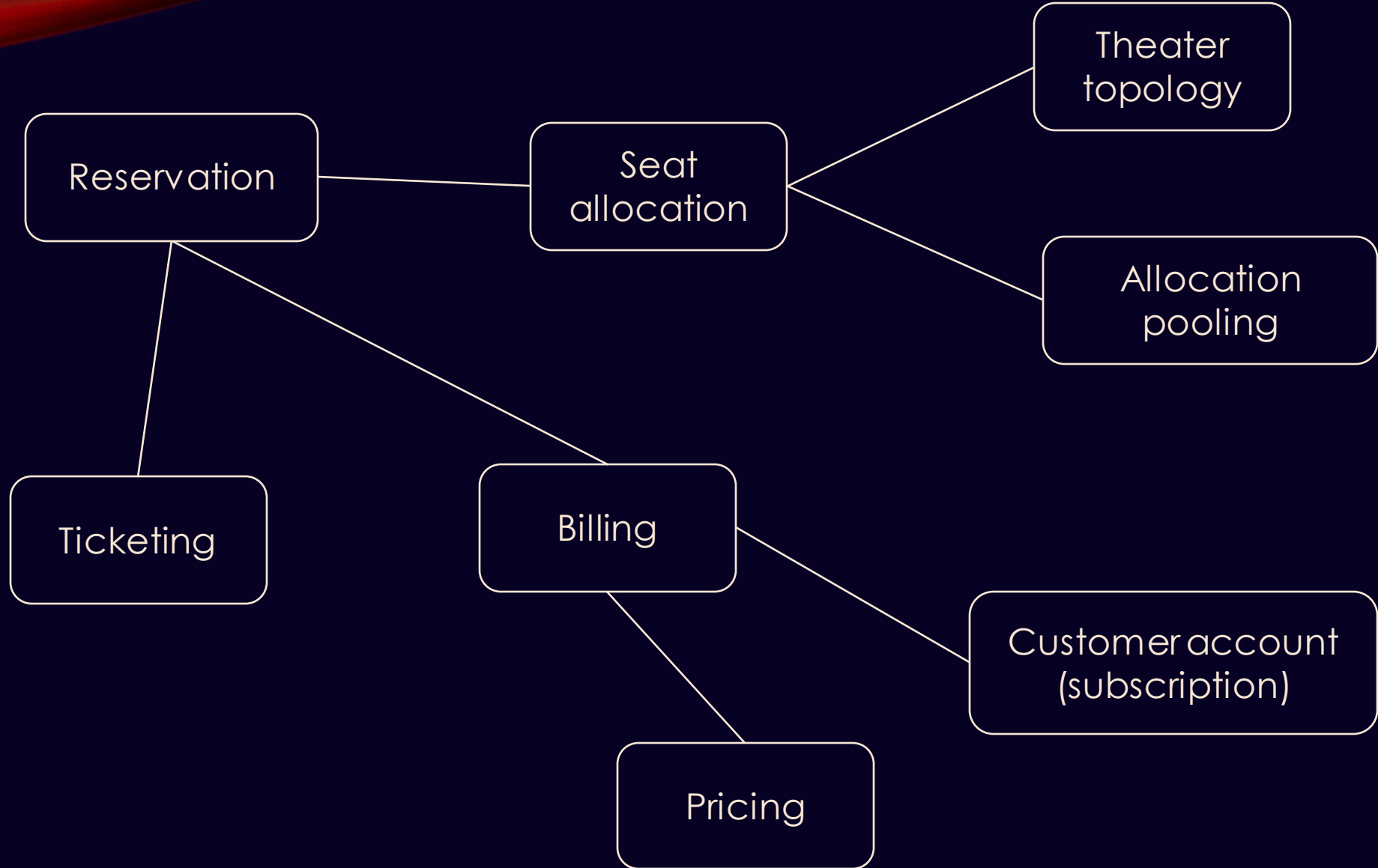
Domain model
(instantiables)

Components
(injectables)

@ArnaudThiefaine @DorraBartaguiz



LET'S CODE



UP TO YOU!



- Rely on test coverage
- Identify Bounded Contexts
- Refactor in baby steps
 - Extract Value Object, Entities,
 - Aggregates, Repositories, Services
- Promote naming



THANKS

@ARNAUDTHIEFAINE
@DORRABARTAGUIZ