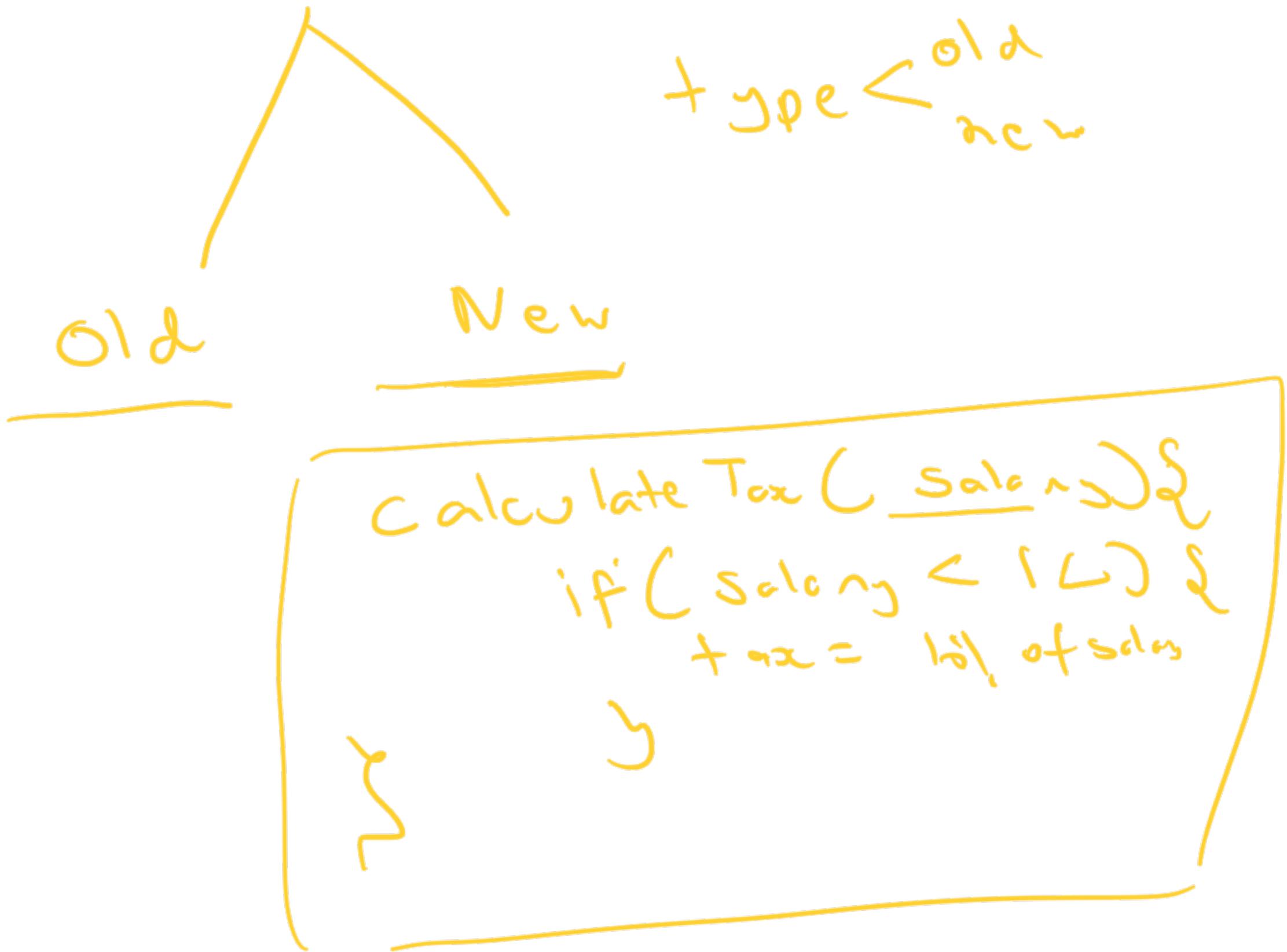


LLD - Adapter and flyweight

Tax Calculation



Factory - set of related classes



Old

New

Condition — regime

Condition — multiple subklasses

Abstract factory

- Sub classes
- family of sub class.



Family of products

→ factory of factory,

India, nev^{er} regime



India



Nevin India et al.

factory -

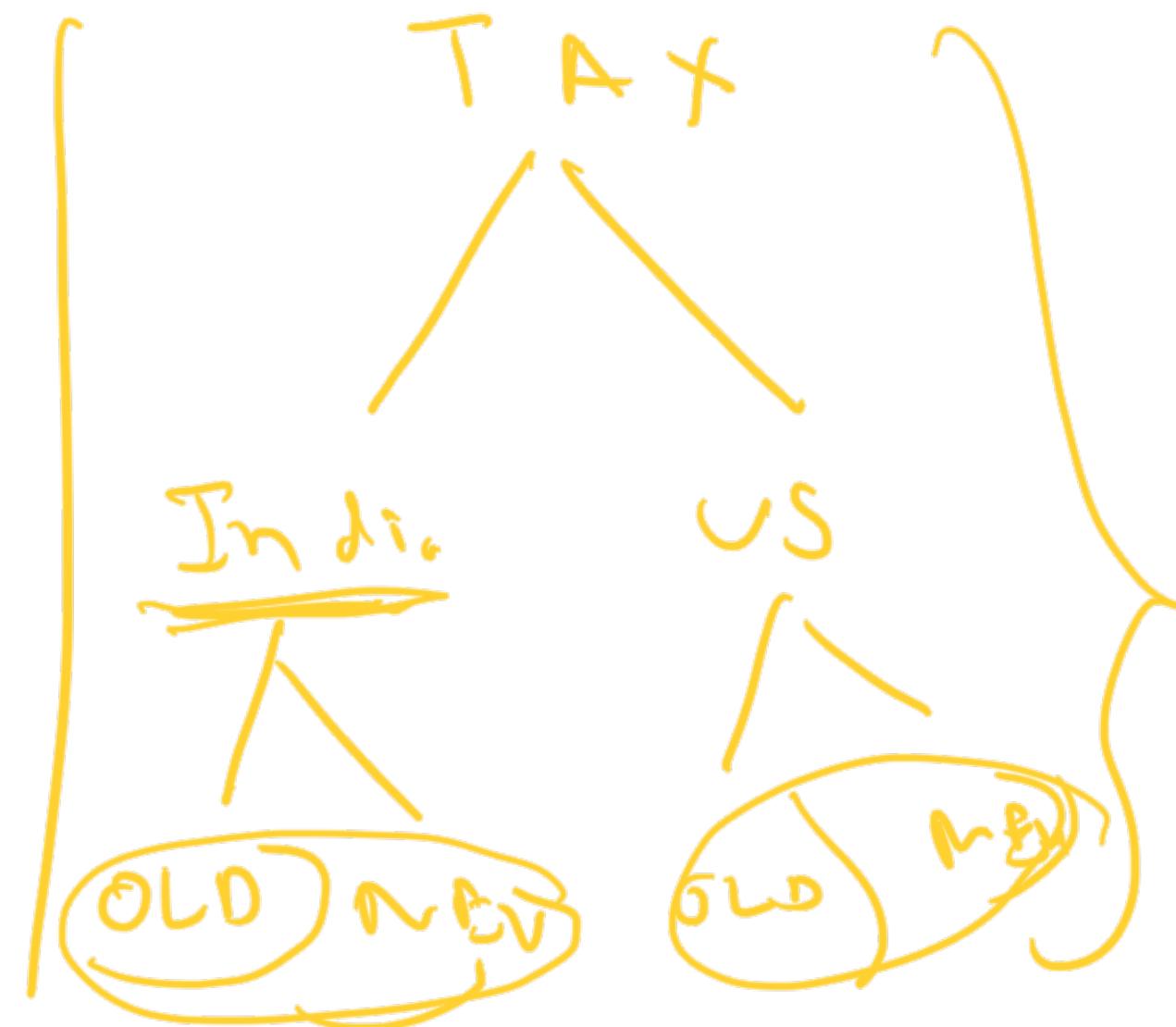
products | related class

- on basis of a revolution
you want to create

diff. institution

abstract \rightarrow family of products

factory \rightarrow hierarchy



A genda - Structural design p.

→ Adapter

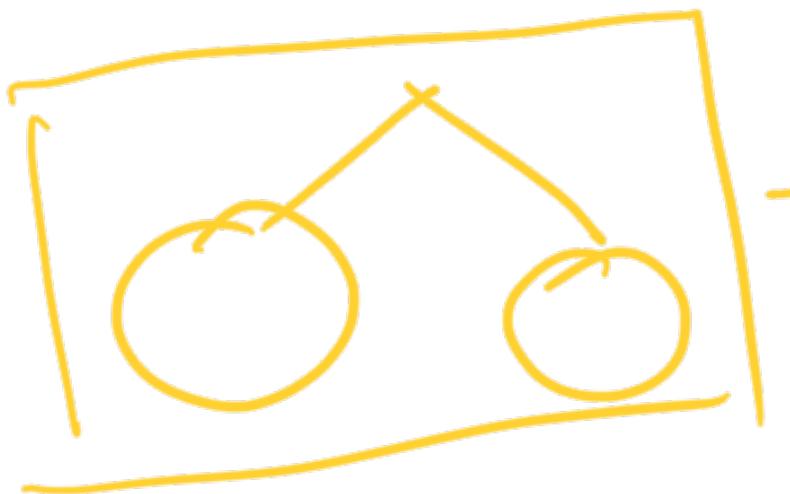
→ Flyweight

→ Decorator

→ Facade

Structural

- ↳ structure
- ↳ interact



SOLID

loose coupling

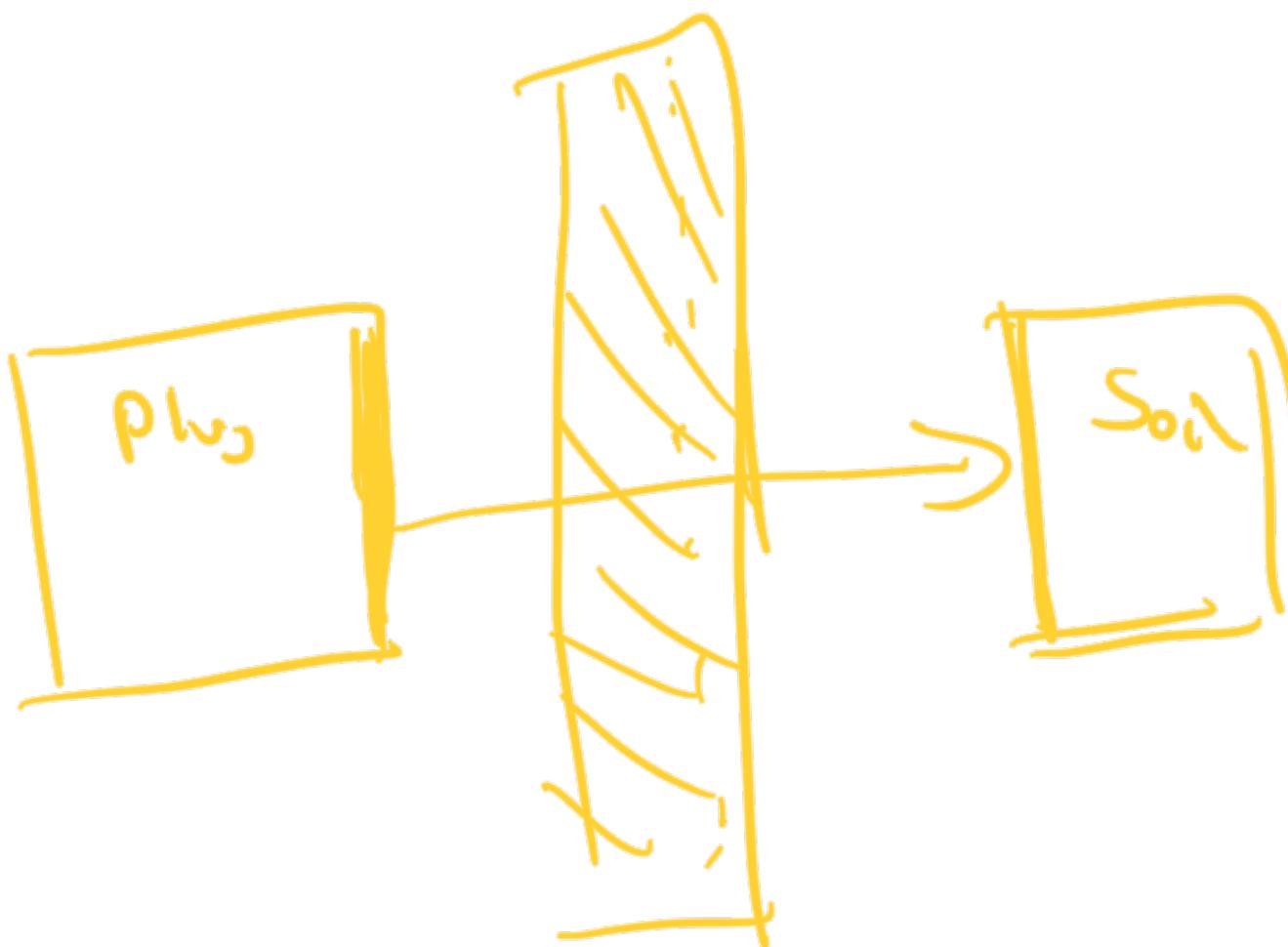
tight cohesion



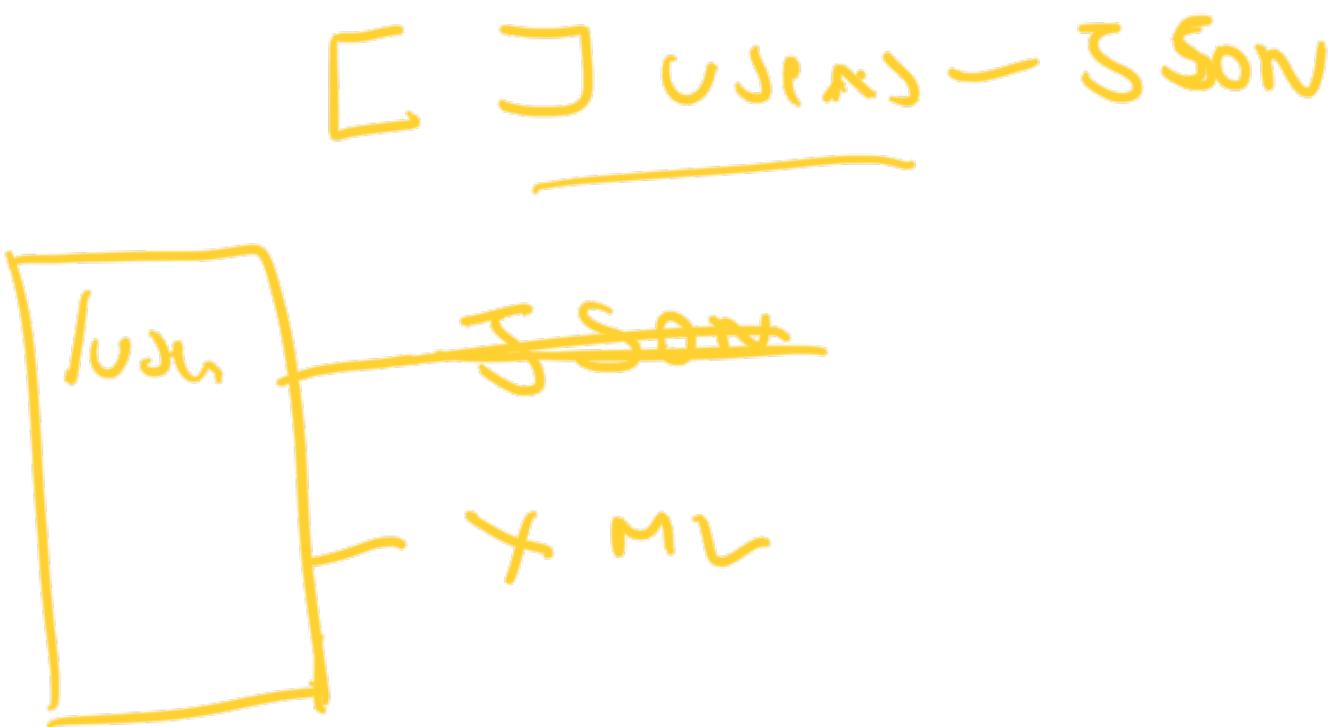
A & open



incompatible interf locs



adapter allows us to work
with incompatible interfaces.



- ① Change from JSON to XML? ~~X~~
- ② Create new API with XML content

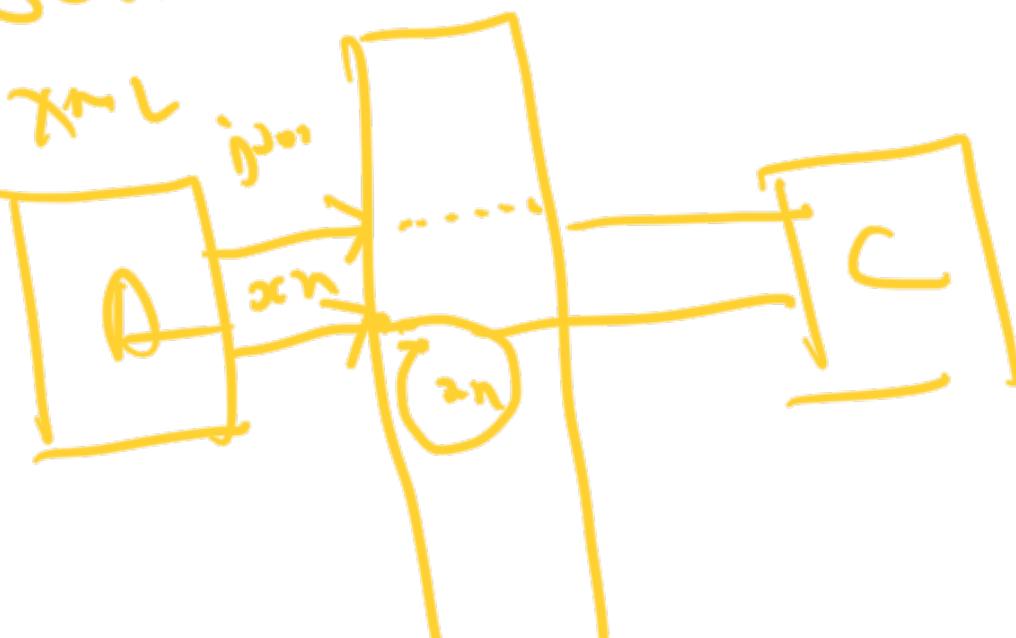
| users - 0m |
| users - 3m | X

③



Java → JSON → XML

Java → JSON
→ XML



Payment Gateway

↳ Royal Pay

↳ Cash Free

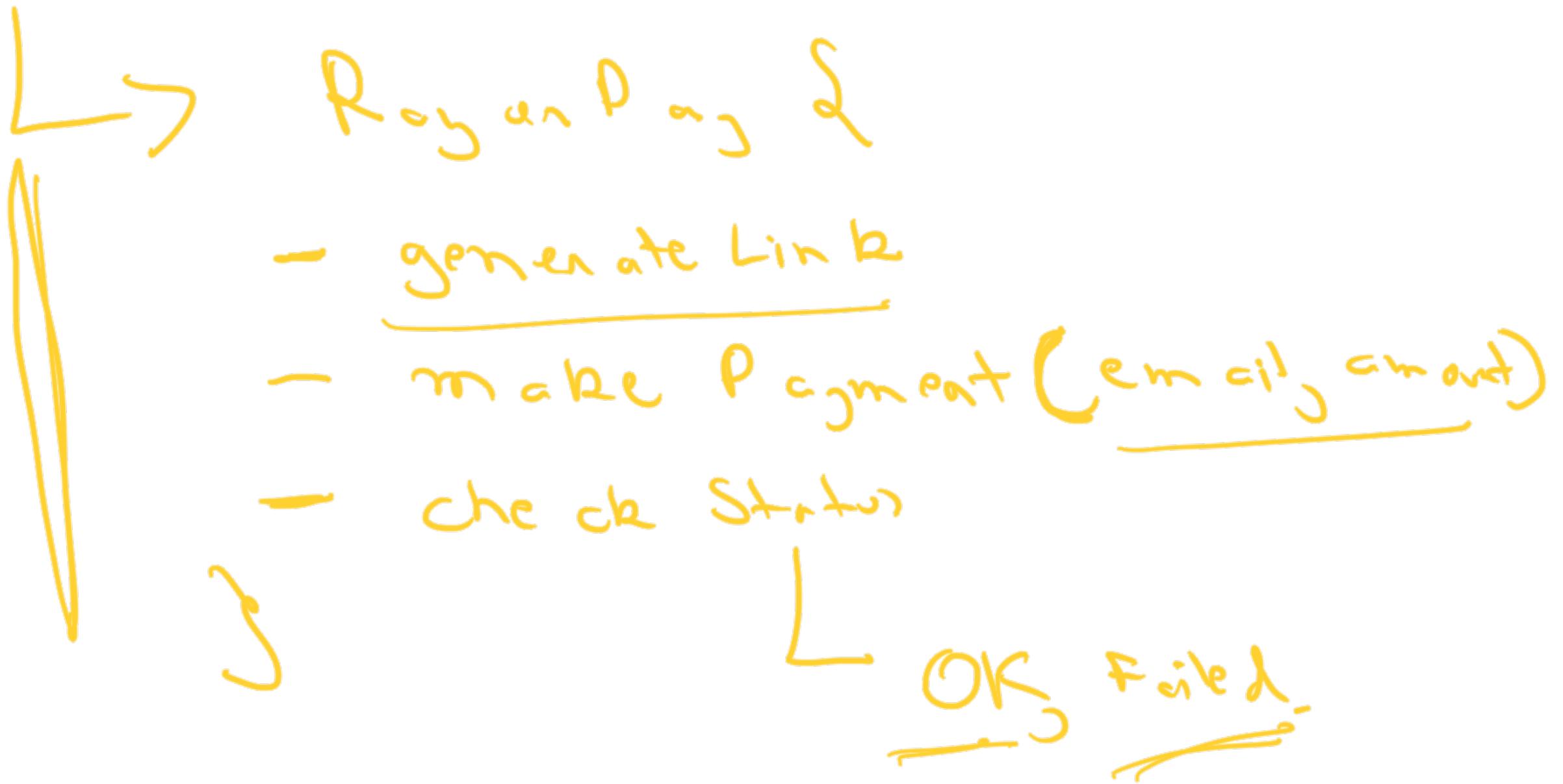
↳ Pay C

↳ Stripe

↳ P, P.I

adoption

RegenPay



make Order()

RegenPay API. makePayment ([email,

3

Page 2

→ createLink()

→ create fragment [email], am out, none)

→ checkStatus()

↳ SUCCESS,
FAILURE



(in out)



①

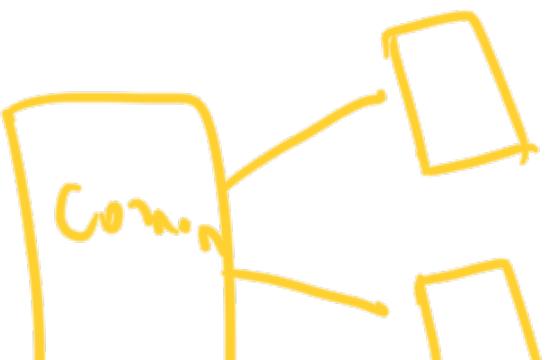
~~Different methods~~

②

~~Different parameters~~

③

~~Different return values~~





-
- Ray casting API
 - Polygon API



Create common interface - ad opter.

interface PaymentProvider {
 → Lin BCJ;
 → Da (Payment Request)

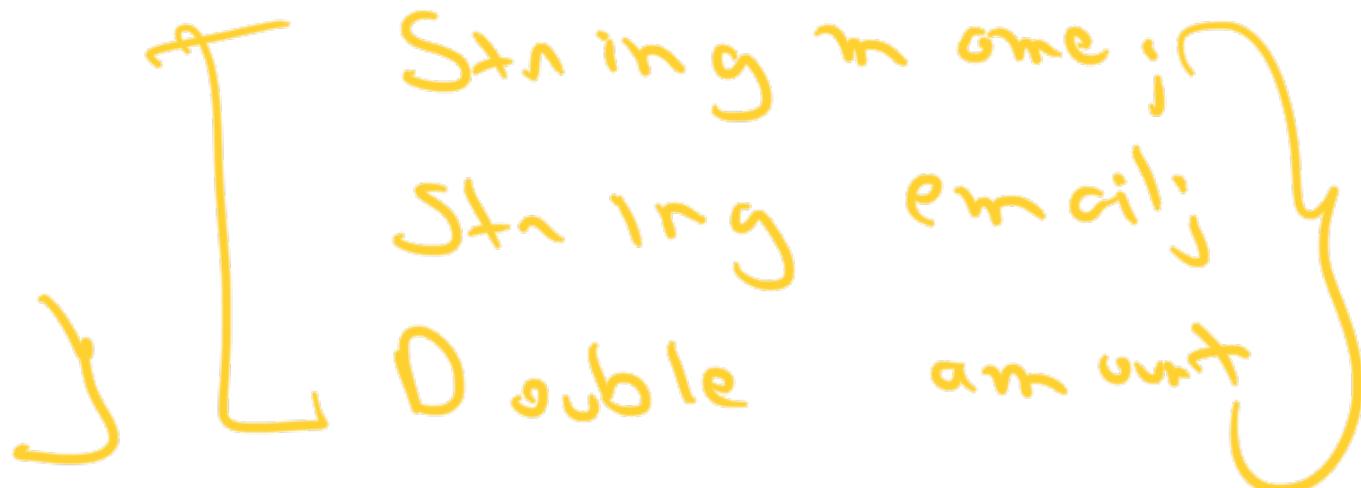
→ Logische Objekte

DTO - Builder

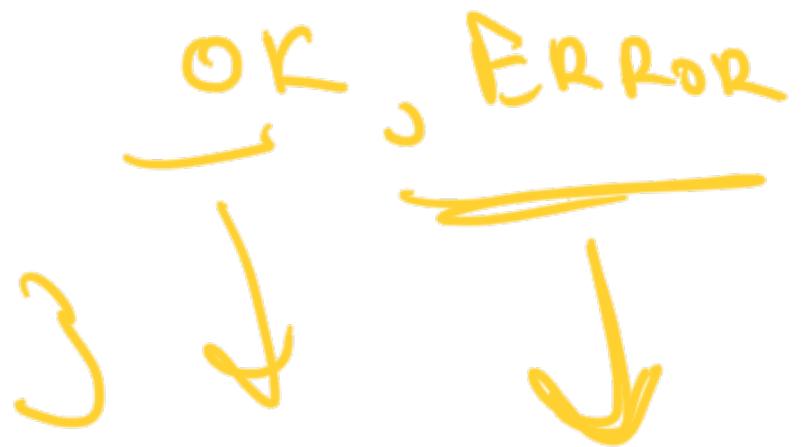


Common DTO

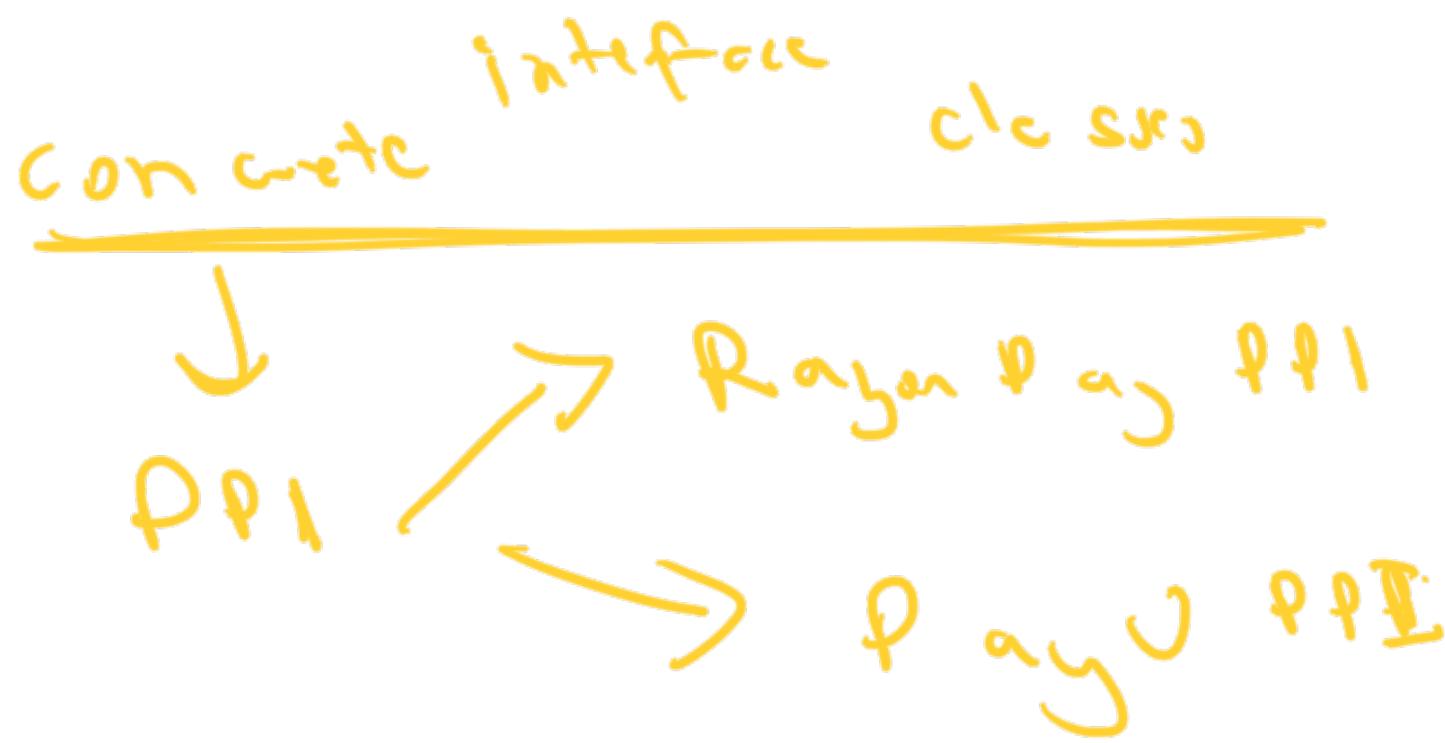
Augment Request



Common Payment Structure



②



Rayon Day ffi ↳

✓ think ↳
 return ready. ↳ delegate

Upcoming Requests

Pay (Payment Request) ↳
return $\tau p.$ → abe Payment
req. email ↳ req. and
req. phone

Check

Check ↳
return $\tau p.$ check status (id)

Request ↳
to RP error ↳ : Payment Status ↳
c. id

To RP error ↳ : Payment Status ↳
c. id

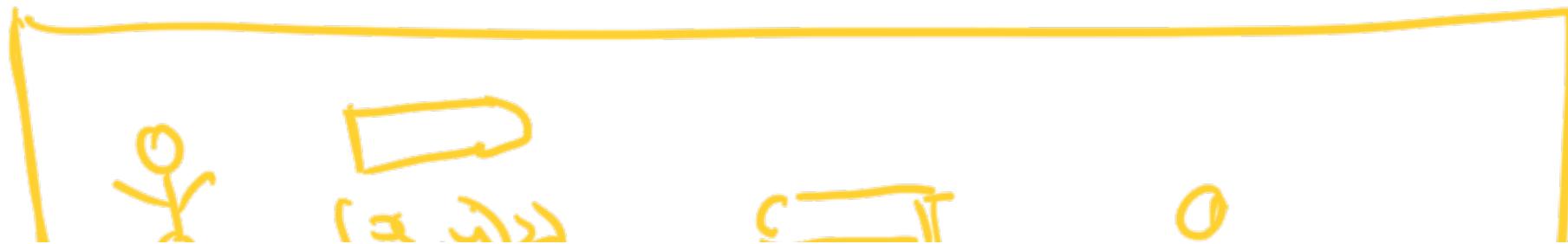
Flyweight

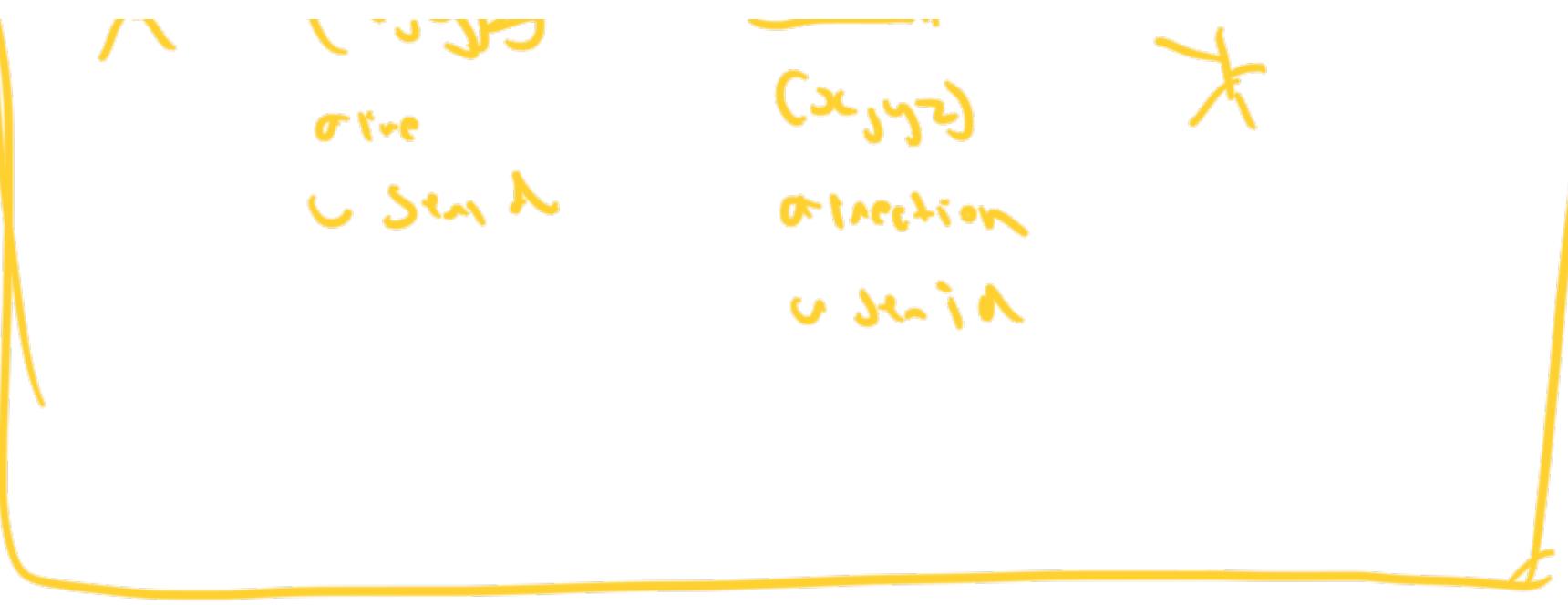
~~extensibility, maintainability~~

→ game development

→ Scalability → RAM

PUBT





- Player
- Map
- Terrain
- Trees
- bullets

bullet
~~Double $x_j = e$~~
~~Double $y_j = e$~~
~~Double $z_j = e$~~
 Double weight; -int
 String colour; -int
Byte[] image; -int

.09 .55 .78
Bullet Type
Type - in

$$4 \times 8 + 8 + 8 + 1 \kappa_B$$

$$= \frac{32}{1} + 1 \kappa_B$$

$$\approx 1 \kappa_B = 1 \text{ b cllct}$$

$$200 \times 200 + 1$$

$$40000 40 \kappa_B$$

$$= 2000 + 200 \times 1$$



fly weight
-
extreme
→ charge

→ on →
→ behavior.
intrinsic
→ does not change



Bullet

(Flyweight)

- image
- weight
- type

Flyingbullet

- x_0, y_0, z
- speed

Incl.



Create fly weight

Bullet {
image }

→ intrinsic
weight
type
color

②

Create extrinsic state object

Flying Bullet ↴
~~x₀, y₀, v₀~~

- Speed
- Bullet bullet

③

Compose bullet & FB

Flying Bullet ↴
....

Bullet bullet

U

