LLD - Abstract Factory and Adapter patterns

Agenda

- Abstract factory
  - Simple factory
  - factory method
  - Impl.
- Structural DP
  - Adapter

1.5

1hrs

---

① Simple factory

- not a design DP
- do not depend on subclasses

= maintenance nightmare

① Product interface

→ Button

→ background Object

→ Database

② Concrete product

Button → PrimaryButton

→ RoundButton

Database → MySQL
→ MongoDB

③ Class - Factory

ButtonFactory

DatabaseFactory

3.5 Static method

type → Product interface

ButtonType

DbType

conditionally create instance of

Btn

static ^ create Btn (BtnType) {

→ switch (type) {

→ PRIMARY:

return new PBtn();

} → R.

}

Adv → subclass

→ simple

Downsides:

→ SRP ✗

→ common place

for construction

Factory method

Product ⟷ Factory

Round ⟷ RoundButtonFactory

PB ⟷ PBFactory

① Product

② Concrete product

③ Factory interface

interface ButtonFactory {
  Btn createButton();
}

→ Factory method
→ virtual ctor.

④ Concrete factories

Btn

BtnFactory

(5) client code

PBFactory → btnFactory factory
= new PBFS

btn instance
→ factory.createBtn()

Btn , Primary

- getFactory(Primary)

↓

PBfactory

- getFactory(type)

↳ factory

Advantages | Disadvantages

① SRP ✓

② OCP ✓

② too many classes

Database

database = Database.create(

url

)

MySQLDatabase

mysql://

mongo://

Abstract factory

Chair

Family of products

Victorian chair + Victorian table

Victorian

Abstract factory

→ related family of products

Themes

Light theme

→ Light Btn

→ Light Checkbox

Dark theme

→ Dark Btn

→ Dark Checkbox

related family — Theme

|

→ related set of products

→ ensure that only products of some type can be provided to the user

→ factory of factories

1 Abstract Factory

PB factory RBF DCF LCF

S

→ btn factory . create btn()

→ C factory . create Checkbox()

y

: Dark Checkbox + Light Button

Fail → [Light, Light]

→ [Dark, Dark]

ThemeFactory
Dark Theme
Light Theme
create Dark
create DarkCheckbox
ThemeFactory

Create Btn

Create Checkbox

— ①

Darktheme

Lightthem

Create Btn

- Light Btn

Button [illegible]

Dark

Checkbox

Dark

---

① Product interface →

② Product concrete class →

③ Abstract factory interface →

→ Create btn

→ Create Checkbox

4

Concrete factory classes

→ Dark Theme factory

→ Light Theme factory

Creational

→ Singleton → Single instance

→ Builder — ✓

→ Prototype → object creation performance + registry

→ Factory → ?

→ different types of objects

→ ctor — lo variable

— builder

— Builder.

— beautiful

→ minimal, self documenting

→ not duplicated

Prototype — clone one an object that exists

prototype

- Chain table) Fun

Type → prototype.

prototype + factory

register
.Clone

creating

Singleton

Break

6:09 — 6:15

10:45

Theory

↓

Builder DP

LLD

→ usecase

→ Design TicTacToe

Bo ah

Buil.

Cpx

Sop → Adapter

→ Decorator

→ Flyweight

BOP → Observer
→ Strategy

Case Studies

- Real world - Pen
- Game - TicTacToe
- MS - parking
  - email
  - Bing
  - splitwise

Structural Design pattern

Adapter

Software Engineering

API

{ users: [ ]

}

XML

→

CSV

/api-json

/api-xml

Adapter Layer

/ users

/ users ? format = csv

Payment Gateway:

Razorpay

Stripe

→ PayU

→ Cashfree

Query tuning

→ monitoring

→ APM – application monitoring

– new relic, Sentry

→ SQL – slow queries

2S

→ EXPLAIN/ANALYZE

→ query execution plan

→ Scan?

→ Table scan

→ In dex Scan

→ Rows?

→ index covering

→ cost