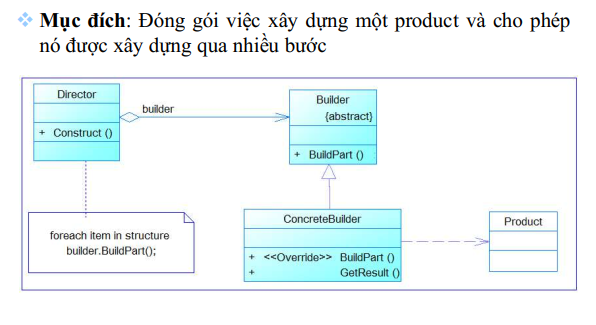
- Builder pattern: xây dựng đối tượng nhìu thành phần cần khởi tạo

Chia nhỏ contracter

Cách viết



Sử dụng builder có nhiều contructor gây khó hiểu, set, get

Ví dụ bánh Piza hàm dẫn: size, phô mai, … nhiều tham số

Khách hàng muốn một số thứ, không cần đủ hết

Director; sử dụng Builder( build các đối tượng mình cần thiết)

Có cỏcreateBuilder kế thừa builder

Class product: đối tượng xe, pizza

Xe oto tự động, con

----------- Ôn tập kiểm tra và thi -------------------------

**- Stragery:**

+ Trường hợp sử dụng: khi mà 1 đối tượng có cùng kiểu hành vi giống nhau thì ta đưa kiểu hành vi vào giao diện và các đối tượng gọi lại kiểu hành cụ thể thông qua 1 giao diện chung

+ Mục đích sử dụng: Cho phép ta định nghĩa 1 tập hợp các thuật toán chung (phương thức chung), và mỗi lớp rieeng biệt sẽ implement lại giao diện đó, mà không làm thay đổi cấu trúc của giao diện đó.

- **teamplate method:**

+ Mục đích: định nghĩa 1 khung sườn của 1 thuật chung bao gồm nhiều bước trong 1 phương thức( tức là công thức chung của 1 đối tượng được đưa vào khung sườn, và các hành vi khác nhau gọi lại công thức chung đó và sử dụng chúng)

+ Mục đích sử dụng: Cho phép ta định nghĩa 1 tập hợp các thuật toán chung (phương thức chung), và mỗi lớp rieeng biệt sẽ kế thừa lại lớp trừu tượng đó, mà không làm thay đổi cấu trúc của lớp trừu tượng đó.

- **So sánh giữa template và stragery:**

Điểm khác:

+Templated Method: Mỗi lớp con sẽ cài đặt 1 thuật toán cụ thể khác nhau bằng cách viết lại body của các phương thức mà nó Override từ lớp trừu tượng.

+ Stragery: Mỗi đối tượng sẽ được chọn 1 hành vi cụ thể trong 1 giao diện nào đó, mà không cần phải implement toàn bộ hành vi cụ thể của 1 giao diện đó.

**--- Observer ---**

+ Mục đích: Định nghĩa 1 phụ thuộc one to many giữa các đối tượng sao cho khi 1 đối tượng (one) thay đổi trạng thái, thì các đối tượng còn lại (many) sẽ được nhận thông báo và cập nhật trạng thái 1 cách tự động

vd: Trong lớp học thầy giáo (one) còn sinh viên(many). Thầy giáo đưa ra bài tập về nhà yêu cầu hoàn thành nó thì các đối tượng phụ thuộc vào thầy giáo đó sẽ nhận thông báo và cập nhật trạng thái(đã hoàn thành hay chưa hoàn thành BT).

+ Trả lời câu hỏi "Giải thích vai trò của hàm Update()": trong IObserver chứa danh sách các thành viên đã đăng ký từ trước. Khi người quản lý update 1 thông tin mới thì các thành viên trong danh sách đều đc cập nhật lại thông tin mới.

+ Cấu trúc: Subject là người quản lý hoặc tỉ giá

ConcreateObserver là thành viên hoặc nhà đầu tư

foreach o in Observer o.update() có nghĩa là cứ mỗi lần người quản lý cập nhật thông tin mới thì thông tin đó sẽ dc cập nhật cho từng thành viên. Tại Subject có 1 mũi tên hướng đến IObserver vì nó cần lấy thông tin từ IObserver

**------ Iterator ----------**

+ Mục đích: Cung cấp 1 cách truy cập các phần tử của 1 tập hợp 1 cách tuần tự mà không cần biết cấu trúc của tập hợp đó

+ Dùng khi ta có 1 mảng 1 chiều (hoặc mảng 2 chiếu, mảng số nguyên, mảng kí tự,...) gồm tên của các xe hơi. Mẫu này sẽ giúp ta truy cập tên xe hơi đầu tiên(First) hoặc tên xe hơi tiếp(Next) theo hoặc tên xe đã bán (isDOne).

SKIP TẠM THỜI

---- **CHAIN -OF RESPONSSIBLITY -----------**

+ Mục Đích: Tách rời người gởi và người xử lý request

+ Trường hợp sử dụng: Ví dụ bài xét duyệt danh hiệu thành viên (vàng bạc kim đương) theo điểm.

+ Ý tưởng:

Kết nối các đối tượng thực hiện request thành một chuỗi

Chuyển request dọc theo chuỗi cho đến khi gặp được đối tượng có

khả năng xử lý được nó.

----------------------------Nhóm mẫu khởi tạo ---------------------------------

**----------Singleton**

+ Mục đích: Đảm bảo chỉ có 1 thể hiện (instance) của 1 lớp (Class) được tạo (thông qua void creatInstance...), và cho phép sự truy cập toàn cục (Chỉ instance khai báo từ class) đến đối tượng đó (instance).

+ Việc tạo đối tượng của 1 lớp phải do chính lớp đó đảm nhận.

-----------Xem mã lệnh trong sách để hiểu cấu trúc và trường hợp sử dụng

**--------------Factory Method -----------------------**

+ Mục đích: Tạo 1 giao diện (interface) để tạo đối tượng (Thay cho lệnh new ấy). Nhưng để cho lớp con. Tách phần mã lệnh thương xuyên thay đổi và phần mã lệnh cố định. Nhằm đảm bảo

việc nâng cấp chương trình dễ dàng hơn. Việc sử dụng giao diện giúp các lớp nó implement lên

nó thực thi dễ dàng hơn.

+ Câu hỏi nâng cao,Factorry Method dùng để làm gì?:đẻ tách rời phần khởi tạo thể hiện (instance) đối tượng của chương trình với các phần còn lại của chương trình.

+ Trường hợp sử dụng": Tạo đối tượng mà không chỉ ra lớp cụ thể của nó.

vd instance: Duck duck\_instance = new Duck(); thì cái duck\_instance chính là 1 thể hiện

**-------------Builder pattern:**

+ Mục Đích : Đóng gói việc xây dựng máy tính (Product) và cho phép nó được xây dựng

(lắp ráp) qua nhiều bước(addCPU, addRAM,...).

+ Ưu điểm : Đóng gói cách xây dựng 1 đối tượng phức tạp.

Cho phép 1 đối tượng được xây dựng qua nhiều bước và tiến trình khác nhau.

Sự thực thi 1 product (Computer) có thể được hoán chuyển in,out bởi vì client chỉ thấy 1 giao diện trừu tượng.

------**Compossite pattern:**

Mục đích:  Sắp xếp các đối tượng vào các cấu trúc cây để biểu diễn các

phân cấp part-whole giữa các đối tượng

 Đối xử với các đối tượng riêng lẻ và nhóm các đối tượng theo

các cách giống nhau

**--------Decorator**

Mục đích: Cho phép thêm mới các trạng thái và hành vi vào 1 đối tượng lúc run-time bằng cách sử dụng kỹ thuật subclassing để mở rộng các chức năng của lớp

**----------Facede**

Mục đích:

- Cung cấp 1 interface hợp nhất cho 1 tập các interface trong 1 subsystem

- Định nghĩa 1 interface ở mức cao làm cho việc sử dụng subsystem trở nên dễ dàng hơn