TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN – CHƯƠNG TRÌNH CHẤT LƯỢNG CAO

BRIDGE DESIGN PATTERN

Seminar

Lớp: *21CLC08*
Giảng viên: *Đỗ Nguyên Kha*

Thành viên:

- 21127386 – Nguyễn Thị Cẩm Nhung
- 21127398 – Bành Minh Phương
- 21127466 – Hoàng Anh Tú
- 21127598 – Phạm Tiến Đức

Thành phố Hồ Chí Minh, Tháng 12, 2022

MỤC LỤC

I. Giới thiệu về design pattern

Khi lập trình các dự án, đôi lúc chúng ta sẽ gặp phải những vấn đề mà chúng ta đã gặp phải trước đây. Khi giải quyết những vấn đề ấy, chúng ta thường sử dụng lại mẫu thiết kế cũ trước đó, việc sử dụng lại mẫu thiết kế này là một kỹ thuật trong lập trình hướng đối tượng.

Với từng vấn đề cụ thể khác nhau, chúng ta sẽ có mẫu thiết kế cụ thể để giải quyết vấn đề ấy một cách tối ưu nhất. Vì vậy, có rất nhiều mẫu thiết kế khác nhau, tuy nhiên chúng thường được dựa trên những quy tắc sau đây về hướng đối tượng:

+ Lập trình cho interface chứ không phải để implement interface đó.(?)
+ Ưu tiên object composition hơn là thừa kế.

Các mẫu thiết kế thường được chia thành 3 nhóm gồm:

+ Nhóm khởi tạo (Creational pattern)
+ Nhóm cấu trúc (Structural pattern)
+ Nhóm tương tác (Behavioral pattern)

II. Bridge Pattern

1. Vấn đề

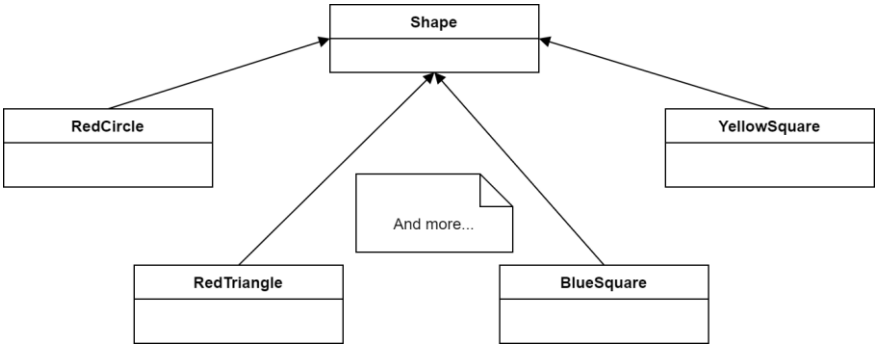
Giả sử ta có một vấn đề như sau:

+ Có các khối hình vuông, hình tròn, hình tam giác.
+ Có các màu đỏ, vàng, xanh.
+ Với mỗi khối ta có thể lựa chọn một màu.

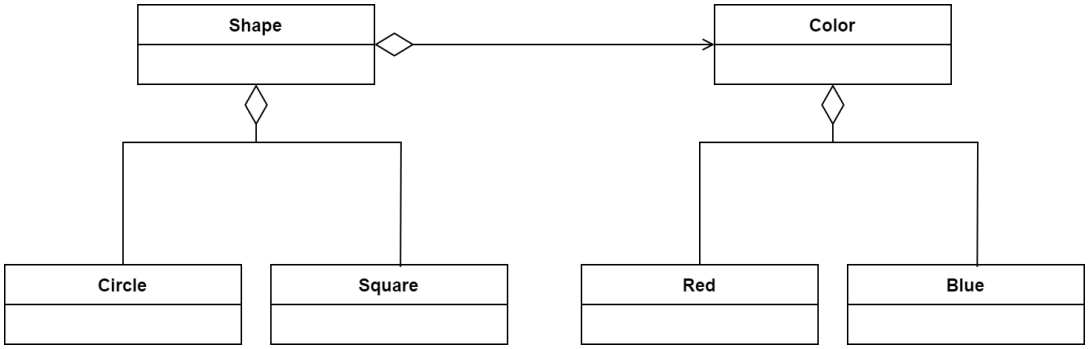
Với vấn đề nêu trên, nếu như chúng ta muốn tạo ra tất cả các tổ hợp của khối và màu thì sẽ phải cần đến 9 lớp đối tượng như là RedCircle, RedTriangle, BlueSquare, YellowSquare, …

2. Giải quyết

Với hướng giải quyết như trên, mỗi khi chúng ta muốn thêm một khối hình dạng, chúng ta sẽ phải tạo ra thêm nhiều lớp hình dạng đó tương ứng với những màu chúng ta có. Và nếu như chúng ta muốn thêm nhiều hơn một khối hình, thì việc liên tục tạo ra các lớp đối tượng như vậy sẽ khiến code của chúng ta trở nên rất thiếu tổ chức và sẽ rất khó để có thể bảo trì sau này.



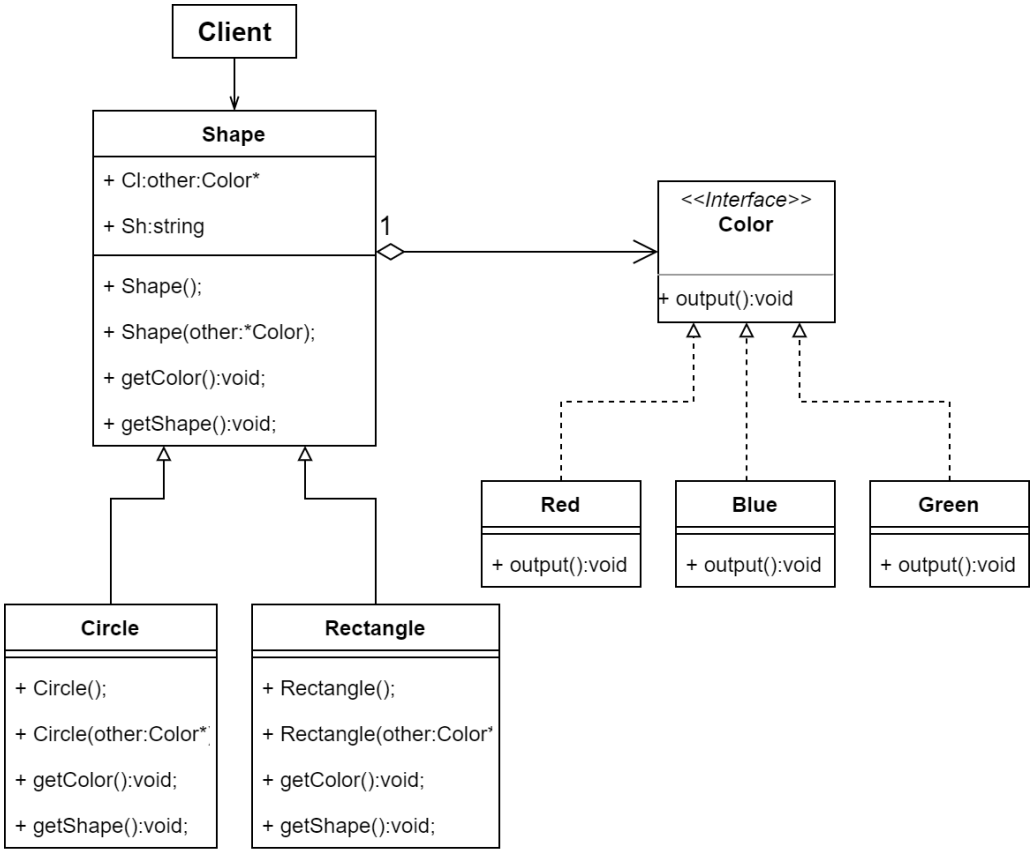
Để giải quyết vấn đề trên, chúng ta có thể sử dụng đến mẫu thiết kế Bridge. Mẫu thiết kế Bridge là một trong các mẫu thiết kế thuộc nhóm cấu trúc. Ý tưởng của mẫu Bridge là tách tính trừu tượng ra khỏi tính hiện thực, từ đó có thể dễ dàng chỉnh sửa hoặc thay thế mà không làm ảnh hưởng đến những nơi có sử dụng lớp ban đầu.



Trong ví dụ của chúng ta, các khối sẽ được xem như là các *Abstract*, còn các màu sắc chính là phần *Implementation* của chúng. Và hiện tại chúng ta đang muốn mở rộng chúng một cách độc lập. Bridge giải quyết vấn đề này bằng cách tạo ra một cầu nối giữa phần *Abstract* và phần *Implementation* để khi mở rộng, thay thế hay chỉnh sửa một trong 2 phần ấy sẽ không ảnh hưởng đến chương trình hiện tại.

3. Cài đặt

Để sử mẫu thiết kế Bridge cho ví dụ nêu trên, dưới đây là sơ đồ lớp cài đặt:



Với sơ đồ trên, các lớp *Red, Blue, Green* được kế thừa từ lớp *Color*. Còn với các lớp *Circle, Rectangle* sẽ được kế thừa từ lớp *Shape*. Lớp *Color* sẽ được xem như là một thuộc tính của lớp *Shape*.

4. Sử dụng khi nào

- Khi muốn tách sự ràng buộc giữa *Abstraction* và *Implementation*, để có thể dễ dàng mở rộng độc lập nhau.
- Khi cả *Abstraction* và *Implementation* đều nên được mở rộng bằng những subclass.
- Thay đổi trong thành phần được bổ sung thêm của một *Abstraction* mà không ảnh hưởng đối với các Client.

5. Các ưu và nhược điểm

Ưu điểm:

+ Giảm số lượng lớp con không cần thiết.
+ Code sẽ gọn gàng và kích thước ứng dụng sẽ nhỏ hơn.
+ Dễ dàng bảo trì và phát triển về sau.
+ Giảm sự phụ thuộc giữa *Abstraction* và *Implementation*.

Nhược điểm:

+ Có thể trở nên phức tạp hơn khi áp dụng cho những lớp cần có tính liên kết, phụ thuộc giữa *Abstraction* và *Implementation* cao.

+ Dễ bị lạm dụng quá nhiều dẫn đến việc phức tạp hóa các vấn đề đơn giản một cách không cần thiết.

III. Nguồn tham khảo

1. refactoring.guru, *Bridge:*
https://refactoring.guru/design-patterns/bridge
2. viblo.asia, *Bridge Design Pattern - Trợ thủ đắc lực của Developers (Tháng 11 21, 2021):*
https://viblo.asia/p/bridge-design-pattern-tro-thu-dac-luc-cua-developers-gDVK2oG2ZLj?fbclid=IwAR3xtbRR5DMXCEXImgc2jU_i4gBddEdgmxfL73i0a425el5tbSw0ApDV3QA