

6.3 Mẫu thiết kế FACADE

6.3.1 Mở đầu

Trong bài này, chúng ta sẽ bàn luận về một mẫu cấu trúc khác, tức là mẫu Façade (về bên ngoài). Nhưng trước hết ta sẽ bàn luận về bài toán mà sẽ được giải quyết bởi mẫu này.

Công ty của bạn là công ty chế tạo sản phẩm và sẽ đưa sản phẩm ra thị trường, tên là Máy chủ lập lịch (Schedule Server). Nó là một kiểu máy chủ được dùng để quản lý các công việc. Các công việc có thể có kiểu bất kỳ như danh sách thư điện tử email, tin nhắn sms, đọc hoặc viết các file từ một đích nào đó, hoặc chỉ đơn giản là truyền file từ nguồn đến đích. Sản phẩm được sử dụng bởi người phát triển để quản lý các kiểu công việc như vậy và có khả năng tập trung hướng tới mục tiêu kinh doanh của họ. Máy chủ thực hiện mỗi công việc tại mỗi thời điểm yêu cầu của họ và cũng quản lý mọi vấn đề bên dưới khác như vấn đề song song và an ninh bởi chính nó. Như một nhà phát triển, người ta chỉ cần code các yêu cầu kinh doanh liên quan và một số các lời gọi API sẽ được cung cấp để xếp lịch cho công việc theo yêu cầu của họ.

Mọi việc tiến triển tốt, cho đến khi khách hàng bắt đầu than phiền về các tiến trình bắt đầu start, kết thúc stop của máy chủ. Họ nói mặc dù máy chủ làm việc tốt, các tiến trình khởi động và dừng còn rất phức tạp và họ muốn có cách nào khác để làm điều đó. Máy chủ đưa ra giao diện phức tạp cho khách hàng mà trông hơi đáng sợ đối với họ.

Chúng ta cần cung cấp cách dễ dàng hơn để khởi động và tắt máy chủ.

Một giao diện phức tạp đối với khách hàng được xem xét như một lỗi thiết kế của hệ thống hiện tại. Nhưng may mắn hoặc không may mắn, chúng ta không thể bắt đầu thiết kế lại và coding từ bản nháp. Chúng ta cần một cách để giải quyết vấn đề này và làm cho giao diện dễ dàng truy cập hơn.

Mẫu Façade có thể giúp chúng ta giải quyết bài toán thiết kế này. Nhưng trước hết, chúng ta tìm hiểu về mẫu Façade.

6.3.2 Mẫu Façade là gì

Mẫu Façade làm cho giao diện phức tạp dễ dàng sử dụng hơn. Mẫu Façade cung cấp giao diện thống nhất cho tập các giao diện trong một hệ thống con. Façade định nghĩa một giao diện mức độ cao mà làm cho hệ thống con đó dễ sử dụng hơn.

Façade thống nhất các giao diện mức thấp phức tạp của hệ thống con để cung cấp cách đơn giản truy cập đến giao diện đó. Nó chỉ cung cấp một mức cho các giao diện phức tạp của hệ thống con mà làm cho nó dễ sử dụng hơn.

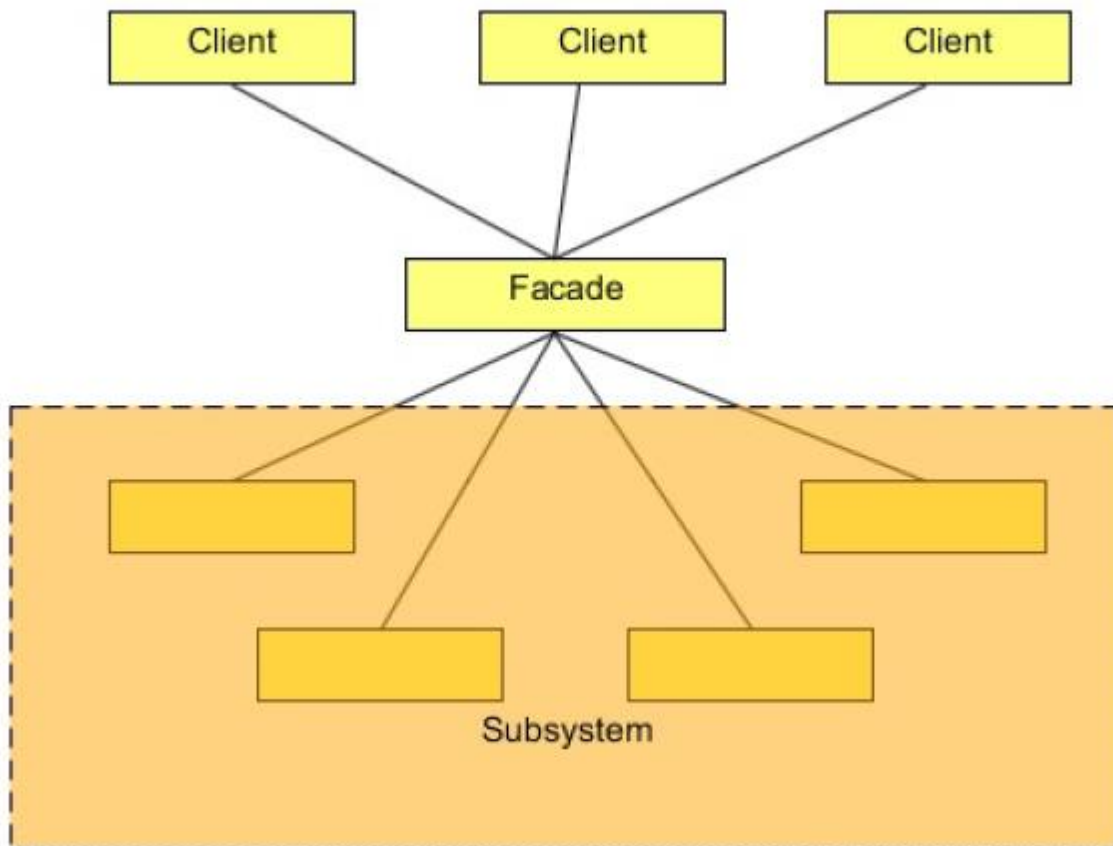
Façade không đóng gói các lớp hoặc giao diện của hệ thống con, nó chỉ cung cấp giao diện đơn giản hóa cho chức năng của chúng. Khách hàng có thể truy cập trực tiếp các lớp này. Nó vẫn phơi bày ra chức năng đầy đủ của hệ thống cho các khách hàng, những người có thể cần nó.

Façade không chỉ có thể đơn giản hóa giao diện, nhưng nó cũng tách gắn kết giữa khách hàng và hệ thống. Nó bám chặt vào “nguyên lý biết ít nhất” mà tránh gắn chặt giữa khách hàng và

hệ thống con. Nó cung cấp sự linh hoạt: giả sử trong hệ thống trên, một công ty muốn bổ sung một số bước để khởi động hoặc tắt máy chủ lập lịch, mà có giao diện khác nhau của riêng mình. Nếu bạn đã lập trình code khách hàng của bạn cho Façade thay vì cho hệ thống con, code client của bạn không cần phải thay đổi, chỉ Façade được yêu cầu thay đổi, mà có thể phân bổ như phiên bản mới cho client.

Các khách hàng trao đổi với hệ thống con bằng cách gửi yêu cầu đến Façade, mà nó sẽ truyền tiếp chúng cho các đối tượng hệ thống con phù hợp. Mặc dù các đối tượng hệ thống con thực hiện các công việc thực tiễn, facade có thể làm công việc của nó là truyền giao diện của nó cho giao diện của hệ thống con. Client mà sử dụng Façade không cần truy cập hệ thống con trực tiếp.

Cần lưu ý rằng, Façade cũng như Adapter có thể bao bọc nhiều lớp, nhưng Façade được sử dụng cho một giao diện để đơn giản hóa việc sử dụng các giao diện phức tạp, trong khi Adapter được dùng để chuyển đổi một giao diện sang giao diện mà client mong muốn.



6.3.3 Giải pháp cho bài toán

Vấn đề đối mặt với khách hàng trong việc sử dụng Máy chủ Lập lịch là tính phức tạp của máy chủ khi khởi động và tắt các dịch vụ. Khách hàng muốn cách đơn giản để làm điều đó. Sau đây là đoạn code mà client được yêu cầu để khởi động và tắt máy chủ:

```
ScheduleServer scheduleServer = new ScheduleServer();
```

Để khởi động máy chủ, client cần tạo đối tượng của ScheduleServer và sau đó cần gọi các phương thức sau theo trình tự đó để bắt đầu khởi động và khởi tạo server.

```
scheduleServer.startBooting();  
scheduleServer.readSystemConfigFile();  
scheduleServer.init();  
scheduleServer.initializeContext();  
scheduleServer.initializeListeners();  
scheduleServer.createSystemObjects();  
  
System.out.println("Start working.....");  
System.out.println("After work done.....");
```

Để tắt máy chủ, client cần gọi các phương thức sau theo cùng một thứ tự

```
scheduleServer.releaseProcesses();  
scheduleServer.destory();  
  
scheduleServer.destroySystemObjects();  
scheduleServer.destoryListeners();  
scheduleServer.destoryContext();  
scheduleServer.shutdown();
```

Điều này trông nặng nề đối với họ, họ không quan tâm đến việc thực hiện các thủ tục này và tại sao chúng cần. Ngay cả khi điều này làm cho một số khách hàng quan tâm họ phải quan tâm giao diện mức thấp của hệ thống, nói chung hầu hết khách hàng không thích điều đó.

Để giải quyết điều này, chúng ta cần tạo ra lớp Façade mà sẽ bao bọc đối tượng máy chủ. Lớp này sẽ cung cấp giao diện (phương thức) đơn giản cho khách hàng. Các giao diện này bên trong sẽ gọi các phương thức trên đối tượng máy chủ. Trước hết ta xem code và sau đó bàn luận về nó.

```
package com.javacodegeeks.patterns.facadepattern;

public class ScheduleServerFacade {

    private final ScheduleServer scheduleServer;

    public ScheduleServerFacade(ScheduleServer scheduleServer){
        this.scheduleServer = scheduleServer;
    }

    public void startServer(){

        scheduleServer.startBooting();
        scheduleServer.readSystemConfigFile();
        scheduleServer.init();
        scheduleServer.initializeContext();
        scheduleServer.initializeListeners();
        scheduleServer.createSystemObjects();

    }

    public void stopServer(){

        scheduleServer.releaseProcesses();
        scheduleServer.destory();
        scheduleServer.destroySystemObjects();
        scheduleServer.destoryListeners();
        scheduleServer.destoryContext();
        scheduleServer.shutdown();

    }

}
```

Lớp ScheduleServerFacade là lớp Façade, mà bao bọc đối tượng ScheduleServer, nó khởi tạo đối tượng server qua hàm tạo của nó, và có hai phương thức đơn giản, *startServer()*, và *stopServer()*. Các phương thức này bên trong thực hiện khởi động và tắt máy chủ. Client chỉ cần gọi các phương thức đơn giản này. Bây giờ, không cần gọi mọi phương thức vòng đời và hủy, chỉ cần phương thức đơn giản và phần còn lại của tiến trình sẽ do Façade thực hiện.

Đoạn code sau chỉ ra cách Façade biến giao diện phức tạp thành đơn giản để sử dụng.

```
package com.javacodegeeks.patterns.facadepattern;

public class TestFacade {

    public static void main(String[] args) {

        ScheduleServer scheduleServer = new ScheduleServer();
        ScheduleServerFacade facadeServer = new ScheduleServerFacade(scheduleServer ↵
        );
        facadeServer.startServer();

    }

}
```

```
        System.out.println("Start working.....");  
        System.out.println("After work done.....");  
  
        facadeServer.stopServer();  
    }  
}
```

Cũng phải lưu ý rằng, mặc dù lớp Façade cung cấp giao diện đơn giản cho hệ thống con phức tạp, nó không đóng gói hệ thống con. Client vẫn còn có thể truy cập giao diện mức thấp của hệ thống con. Vì vậy, Façade cung cấp một mức bổ sung, giao diện đơn giản cho hệ thống con phức tạp, nhưng nó không hoàn toàn che giấu truy cập trực tiếp đến giao diện mức thấp của hệ thống con phức tạp.

6.3.4 Sử dụng mẫu Façade

Sử dụng mẫu Façade khi

- Bạn muốn cung cấp giao diện đơn giản cho hệ thống con phức tạp. Nó làm cho hệ thống con dễ tái sử dụng hơn và dễ tùy biến, nhưng nó có thể khó sử dụng hơn đối với khách hàng cần phải tùy biến theo yêu cầu. Façade cung cấp cái nhìn mặc định đơn giản về hệ thống con mà đủ tốt với hầu hết khách hàng. Chỉ khách hàng cần tùy biến sâu hơn sẽ cần xem xét kỹ hơn về Façade.
- Có nhiều phụ thuộc giữa client và các lớp cài đặt của trừu tượng. Đưa ra Façade để tách gắn kết hệ thống con khỏi khách hàng và các hệ thống khác, khi đó tăng sự độc lập và chuyển mang của hệ thống con.
- Bạn cần tạo mức cho các hệ thống con của bạn. Sử dụng Façade để xác định điểm vào mức từng hệ thống con. Nếu các hệ thống con là phụ thuộc, thì bạn có thể đơn giản sự phụ thuộc giữa chúng bằng cách làm cho chúng trao đổi với nhau duy nhất qua các Façade của chúng.