Tìm hiểu về Proxy Pattern

1. Proxy Pattern là gì

Proxy Pattern là một trong những Pattern thuộc nhóm cấu trúc (Structural Pattern). Proxy có nghĩa là “ủy quyền” hay “đại diện”. Mục đích xây dựng Proxy Pattern cũng chính vì muốn tạo ra một đối tượng sẽ ủy quyền, thay thế cho một đối tượng khác.

Proxy Pattern là mẫu thiết kế mà ở đó tất cả các truy cập trực tiếp đến một đối tượng nào đó sẽ được chuyển hướng vào một đối tượng trung gian (Proxy Class). Mẫu Proxy (người đại diện) đại diện cho một đối tượng kahcs thực thi các phương thức, phương thức đó có thể được định nghĩa lại cho phù hợp với mục đích sử dụng.

2. Phân loại Proxy

* Virtual Proxy: Virtual Proxy tạo ra một đối tượng trung gian mỗi khi có yêu cầu tại thời điểm thực thi ứng dụng, nhờ đó làm tăng hiệu suất ứng dụng
* Protection Proxy: Phạm vi truy cập của các client khác nhau sẽ khác nhau. Protection Proxy sẽ kiểm tra các quyền truy cập của client khi có một dịch vụ được yêu cầu.
* Remote Proxy: Client truy cập qua Remote Proxy để chiếu tới một đối tượng nằm bên ngoài ứng dụng (trên cùng máy hoặc máy khác).
* Monitor Proxy: Monitor Proxy sẽ thiết lập các bảo mật trên đối tượng cần bảo vệ, ngăn không cho client truy cập một số trường quan trọng của đối tượng. Có thể theo dõi, giám sát, ghi log việc truy cập, sử dụng đối tượng.
* Firewall Proxy: bảo vệ đối tượng từ chối yêu cầu xuất xứ từ các client không tín nhiệm
* Cache Proxy: Cung cấp không gian lưu trữ tạm thời cho các kết quả trả về từ đối tượng nào đó, kết quả này sẽ được tái sử dụng cho các client chia sẻ chung một yêu cầu gửi đến. Loại Proxy này hoạt động tương tự Flyweight Pattern.
* Synchronization Proxy: Đảm bảo nhiều client có thể truy cập vào cùng một đối tượng mà không gây ra xung đột. Khi một client nào đó chiếm dụng khóa khá lâu khiến cho số lượng các client trong danh sách hàng đợi cứ tăng lên và do đó hoạt động của hệ thống bị ngưng trệ, có thể dẫn đến hiện tượng “tắc nghẽn”.
* Copy-On-Write-Proxy: Loại này đảm bảo rằng sẽ không có client nào phải chờ vô thời hạn, là một thiết kế rất phức tạp.

3. Cài đặt Proxy Pattern như nào

Proxy Pattern có những đặc điểm sau đây:

* Cung cấp mức truy cập gián tiếp vào một đối tượng
* Tham chiếu vào đối tượng đích và chuyển tiếp các yêu cầu đến đối tượng đó
* Cả Proxy và đối tượng đích đều kế thừa hoặc thực thi chung một lớp giao diện. Mã máy dịch cho lớp giao diện thường “nhẹ” hơn các lớp cụ thể và do đó có thể giảm được thời gian tải dữ liệu giữa server và client.

Các thành phần tham gia vào mẫu Proxy Pattern:

* Subject: là một interface định nghĩa các phương thức để giao tiếp với client. Đối tượng này xác định giao diện chung cho RealSubject và Proxy để Proxy có thể được sử dụng bất cứ nơi nào mà RealSubject mong đợi.
* Proxy: là một class sẽ thực hiện các bước kiểm tra và gọi tới đối tượng class service thật để thực hiện các thao tác sau khi kiểm tra. Nó duy trì một tham chiếu đến RealSubject để Proxy có thể truy cập nó. Nó cũng thực hiện các giao diện tương tự như RealSubject để Proxy có thể được sử dụng thay cho RealSubject. Proxy cũng điều khiển truy cập vào RealSubject và có thể tạo hoặc xóa đối tượng này.
* RealSubject: là một class service sẽ thực hiện các thao tác thực sự. Đây là đối tượng chính mà proxy đại diện
* Client: đối tượng cần sử dụng RealSubject nhưng thông qua Proxy.

4. Lợi ích của Proxy Pattern

* Cải thiện Performance thông qua lazy loading, chỉ tải các tài nguyên khi chúng được yêu cầu
* Nó cung cấp sự bảo vệ cho đối tượng thực từ thế giới bên ngoài
* Giảm chi phí khi có nhiều truy cập vào đối tượng có chi phí khởi tạo ban đầu lớn
* Dễ nâng cấp, bảo trì

5. Sử dụng Proxy Pattern khi nào

* Khi muốn bảo vệ quyền truy xuất vào các phương thức của object thực
* Khi cần một số thao tác bổ sung trước khi thực hiện phương thức của object thực
* Khi tạo đối tượng ban đầu là theo yêu cầu hoặc hệ thống yêu cầu sự chậm trễ khi tải một số tài nguyên nhất định (lazy loading).
* Khi có nhiều truy cập vào đối tượng có chi phí khởi tạo ban đầu lớn.
* Khi đối tượng gốc tồn tại trong môi trường từ xa (remote).
* Khi đối tượng gốc nằm trong hệ thống cũ hoặc thư viện của bên thứ ba.
* Khi muốn theo dõi trạng thái và vòng đời của đối tượng

6. So sánh Proxy Pattern với Decorator Pattern

Cấu trúc của Proxy Pattern và Decorator Pattern là tương tự nhau. Hai Pattern này đểu Wrap một đối tượng ở bên trong nó. Tuy nhiên, khác nhau thật sự giữa Proxy Pattern và Decorator Pattern nằm ở mục đích sử dụng. Với Decorator, người sử dụng sẽ nhắm tới mục tiêu là có thêm tính năng động vào một đối tượng có trước, trong khi đó Proxy Pattern cho phép ta tạo ra một đại diện cho một đối tượng khác.