**Tìm hiểu về Adapter Pattern**

**1. Adapter Pattern là gì**

Adapter Pattern là một design pattern thuộc nhóm Structural Pattern cho phép các interface không liên quan tới nhau có thể làm việc cùng nhau. Đối tượng giúp kết nối các interface gọi là Adapter.

Adapter Pattern giữ vai trò trung gian giữa hai lớp, chuyển đổi interface của một hay nhiều lớp có sẵn thành một interface khác, thích hợp cho lớp đang viết. Điều này cho phép các lớp có các interface khác nhau có thể dễ dàng giao tiếp tốt với nhau thông qua interface trung gian, không cần thay đổi code của lớp có sẵn cũng như lớp đang viết.

Adapter Pattern còn gọi là Wrapper Pattern do cung cấp một interface “bọc ngoài” tương thích cho một hệ thống có sẵn, có dữ liệu và hành vi phù hợp nhưng có interface không tương thích với lớp đang viết.

**2. Cài đặt Adapter Pattern**

Một Adapter Pattern bao gồm những thành phần cơ bản sau:

* **Adaptee**: định nghĩa interface không tương thích, cần được tích hợp vào
* **Adapter**: lớp tích hợp, giúp interface không tương thích được tích hợp với interface đang làm việc. Thực hiện việc chuyển đổi interface cho Adaptee và kết nối Adaptee với Client
* **Target**: một interface chứa các chức năng được sử dụng bởi Client
* **Client**: lớp sử dụng các đối tượng có interface Target

Có hai cách để thực hiện Adapter Pattern dựa theo cách cài đặt của chúng:

* **Object Adapter – Composition**: trong mô hình này, một lớp mới (Adapter) sẽ tham chiếu đến một (hoặc nhiều) đối tượng của lớp có sẵn với interface không tương thích (Adaptee), đồng thời cài đặt interface mà người dùng mong muốn (Target). Trong lớp mới này, khi cài đặt các phương thức của interface mà người dùng mong muốn, sẽ gọi phương thức cần thiết thông qua đối tượng thuộc lớp có interface không tương thích
* **Class Adapter – Inheritance** (Kế thừa): trong mô hình này, một lớp mới (Adapter) sẽ kế thừa lớp có sẵn với interface không tương thích (Adaptee), đồng thời cài đặt interface mà người dùng mong muốn (Target). Trong lớp mới, khi cài đặt các phương thức của interface mà người dùng mong muốn, phương thức này sẽ gọi các phương thức cần thiết mà nó kế thừa được từ lớp interface không tương thích.

**So sánh Class Adapter với Object Adapter**

* Sự khác biệt chính là Class Adapter sử dụng Inheritance (kế thừa) để kết nối Adapter với Adaptee trong khi Object Adapter sử dụng Composition (chứa trong) để kết nối Adapter với Adaptee
* Trong cách tiếp cận của Class Adapter, nếu một Adapter là một class và không phải là một interface thì Adapter sẽ là một lớp con của Adaptee. Do đó, nó sẽ không phục vụ tất cả các lớp con khác theo cùng một cách vì Adapter là một lớp phụ cụ thể của Adaptee.

**Tại sao Object Adapter lại tốt hơn?**

* Nó sử dụng Composition để giữ một thể hiện của Adaptee, cho phép một Adapter hoạt động với nhiều Adaptee nếu cần thiết.

**3. Ưu nhược điểm của Adapter Pattern**

**Ưu điểm:**

* Cho phép nhiều đối tượng có interface giao tiếp khác nhau có thể tương tác và giao tiếp với nhau
* Tăng khả năng sử dụng lại thư viện của interface không thay đổi do không có mã nguồn

**Nhược điểm:**

* Tất cả các yêu cầu được chuyển tiếp, do đó làm tăng thêm một ít chi phí
* Đôi khi có quá nhiều Adapter được thiết kế trong một chuỗi Adapter trước khi đến được yêu cầu thật sự