**Czym różni się wzorzec Strategia od zwykłej implementacji interfejsu?**

**Wzorzec Strategia:**

Strategia jest wzorcem projektowym który umożliwia definiowanie zestawu algorytmów i oddzielenie ich od głównej logiki programu. W tym wzorcu tworzy się rodzaj interfejsu (np. interfejs Strategy), który definiuje zachowanie algorytmu. Następnie różne klasy implementujące ten interfejs reprezentują konkretne strategie, czyli różne sposoby realizacji danego zadania. Wzorzec Strategia pozwala na dynamiczną zmianę zachowania programu w czasie działania poprzez przekazanie różnych strategii.

**Zwykła implementacja interfejsu:**

Jest to po prostu sposób implementacji interfejsu w językach programowania, który umożliwia klasom implementowanie określonych metod zdefiniowanych w interfejsie.Nie musi to być związane bezpośrednio z wzorcem Strategia. Może to być po prostu technika programistyczna do tworzenia modularnego kodu, w którym interfejsy definiują zestaw metod, które muszą być zaimplementowane przez klasy, które je implementują.

**Jakie są wady i zalety tego wzorca?**

**Zalety:**

**Single Responsibility Principle -** Każda strategia ma własną klasę, co sprzyja separacji obowiązków i utrzymaniu modularności kodu.

**Łatwa wymienialność algorytmów**: Dzięki zastosowaniu tego wzorca możliwe jest dynamiczne zmienianie strategii w czasie działania programu, co umożliwia elastyczność i łatwe dostosowanie zachowania aplikacji.

**Wady:**

**Zwiększona liczba klas**: Korzystanie z tego wzorca może prowadzić do zwiększenia liczby klas w programie, co może być problematyczne w przypadku bardzo prostych przypadków użycia.

**Dodatkowy narzut pamięciowy i czasowy**: Wprowadzenie dodatkowych obiektów (strategii) może zwiększyć zużycie pamięci i wymagać dodatkowych obliczeń związanych z wyborem odpowiedniej strategii.