Strategy

O padrão é aplicado em situações em que muitas classes se relacionam e diferem apenas no modo de atuação, com isso o *Strategy* irá configurar a classe que tenha um dentre muitos comportamentos fornecidos. Também pode ser usado quando há a necessidade da variação de um algoritmo, ou seja, pode-se implementar diferentes códigos que chegam no mesmo objetivo, mas que possuem em determinadas situações mais vantagens do que os demais.

Outra situação oportuna para o uso do padrão é em uma aplicação na qual se tem um cliente e este não pode ficar exposto a estrutura de dados do algoritmo. Além disso, quando uma classe tem muitos comportamentos e usam vários comandos condicionais, o desempenho do algoritmo poderá ficar insatisfatório, pois há a possibilidade de existir uma quantidade grande de condições, podendo deixar o código mais lento. Com o padrão pode-se retirar as condições, criando novas classes com estas estratégias, portanto melhorando o desempenho.

Vantagens

Entre os benefícios do padrão Strategy pode-se citar a reutilização por parte do Contexto que permite escolher entre uma família de algoritmos que possuem funcionalidades em comum; os algoritmos em classes Strategy possuem variações do seus algoritmos independentemente do seu contexto, assim é mais fácil utilizá-los, trocá-los, compreendê-los e entendê-los; diminuição ou eliminação da lógica condicional clarificando ainda mais os algoritmos; a Strategy permite que se escolham diferentes implementações do mesmo comportamento; utilizando Strategy há uma grande simplificação na classe ao mover variações de um algoritmo para uma hierarquia; habilita-se que um algoritmo seja substituído por outro em tempo de execução.

Desvantagens

As desvantagens na utilização do Padrão Strategy é a complicação que há de como os algoritmos obtém ou recebem dados de suas classes de contexto; o cliente deve conhecer como que os Strategies diferem, antes mesmo que ele possa selecionar um mais apropriado para o contexto da aplicação; o custo da comunicação entre o contexto e o Strategy é significativo, dado que os Strategies concretos não necessariamente usarão todas as informações da Strategy abstrata, portanto podem haver situações em que o contexto criará e inicializará parâmetros que nunca serão usados; Strategies aumentam o número de objetos no sistema, que pode ser ruim

em determinadas situações em termos de custo e por fim pessoas inexperiente podem ter dificuldade sobre o funcionamento do código por não entender o que é e como funciona o padrão.

Referências

https://www.treinaweb.com.br/blog/padroes-de-projeto-o-que-sao-e-o-que-resolvem

https://www.devmedia.com.br/conheca-os-padroes-de-projeto/957

https://www.devmedia.com.br/estudo-e-aplicacao-do-padrao-de-projeto-strategy/25856