UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO

PRÓ-REITORIA DE GRADUAÇÃO

DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIA

ANNA CLARA FERNANDES

ARTHUR ALVES DE ABREU

BRUNO VICTOR PAIVA DA SILVA

JONAS VICTOR CRECENCIO

VINICIUS ANACLETO ALMEIDA

LISTA DE EXERCÍCIOS - PROJETO DETALHADO DE SOFTWARE

PAU DOS FERROS

2024

01 - Código no github

02 - Código no github

03 - O método createXMLReader() da classe XMLReaderFactory não segue o padrão Factory Method, pois ele não delega a responsabilidade de criação do objeto para subclasses.
Código no github

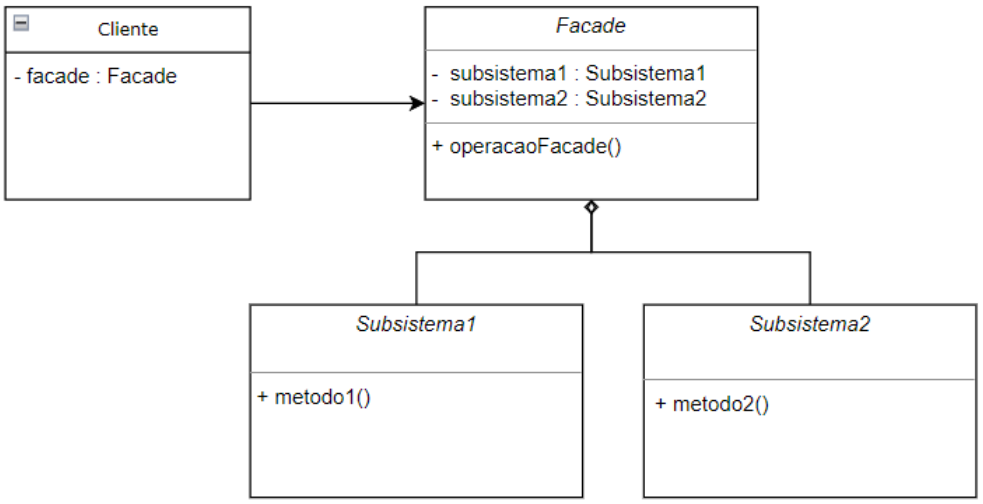
04 - Código no github

05 - A) O padrão Facade é um padrão de projeto que organiza a interação entre o cliente e um sistema ou subsistema complexo, fornecendo uma interface simplificada. Ele age como uma "fachada", ocultando os detalhes internos de implementação e "escondendo" a complexidade do sistema. Com isso, o cliente não precisa conhecer a estrutura interna ou a lógica por trás do funcionamento, tornando o código mais limpo, organizado e de fácil manutenção. E ao fazer uso do facade traz uma separação clara entre a lógica interna e a interface exposta ao cliente, resultando em um código mais simples e com menor acoplamento.

B) O Facade é útil em sistemas grandes e complexos, pois facilita a manutenção e a organização do código. Ele permite que as funcionalidades internas do sistema não precisem ser conhecidas ou manipuladas diretamente pelo cliente. Em vez disso, o cliente interage com a classe Facade que "esconde" as complexidades e assim fornecendo uma interface mais interativa e simplificada ao cliente. Fazendo assim, com que o sistema seja mais intuitivo e fácil do cliente entender.

C) A implementação do Facade é direta e segue uma estrutura simples que tem o objetivo de fornecer uma interface simplificada e unificada para interagir com um ou mais subsistemas, para começar a implementar em nosso sistema devemos ver se o padrão irá realmente simplificar consideravelmente o nosso sistema, pois caso já seja um sistema simples não há necessidade. Para implementarmos o padrão facade em nosso sistema teríamos que criar uma nova classe que iria ficar toda a complexidade que ficava no lado do cliente, essa seria nossa fachada que iria fazer a ponte para a comunicação do cliente com todo o subsistema, então todas as chamadas de métodos do subsistema, instâncias dos objetos, todas as chamadas de código que inicializa o nosso sistema seria refatorados e adaptados

para a nossa classe fachada. Na nossa classe Facade, devemos implementar métodos que encapsulam toda a lógica necessária para seguir o fluxo do sistema de maneira correta. Dessa forma, o cliente poderá simplificar suas interações com o sistema, diminuindo a complexidade e deixando o código mais limpo e legível , pois só precisará instanciar a classe Facade e chamar os métodos disponíveis. Isso permitirá que a comunicação com o sistema ocorra de forma eficiente, iniciando todo o fluxo desejado diretamente pela Facade.



06- Resposta feita e disponibilizada por código java no repositório do Git.

07 - O padrão Prototype deve ser utilizado quando seu código não deve depender de classes concretas de objetos que você precisa copiar.

O padrão Adapter ele pode ser preferido para ser utilizado nos projetos em que é necessário a utilização de interfaces ou bibliotecas de outro sistema como as API.Ele pode ser escolhido para ser implementado em cenários que há um sistema legado em constante atualização, onde novas funcionalidades estão sendo implementadas, mas ainda existem partes do sistema que não foram refatoradas. Nesse cenário, o Adapter permite que o sistema legado e o novo sistema se comuniquem e funcionem juntos, garantindo que o sistema em produção continue operando sem interrupções, mesmo com a coexistência de componentes antigos e

novos.

Exemplo:

Imagine uma empresa que usa o salesforce e possui uma organização (sistema utilizado para gerenciar uma empresa) desorganizada, com códigos que estão sobrecarregando o processo do sistema. A equipe de consultoria que foi contratada por essa empresa decide implementar uma nova base uma nova organização, como no Salesforce, onde a nova organização será refatorada. No entanto, quando essa nova org começa a funcionar, alguns desenvolvimentos, como integrações e objetos, ainda não foram refatorados e será necessário utilizar os legados e por conta disso o Adapter seria a escolha correta de padrão de projeto, permitindo que a nova organização funcione corretamente, mesmo utilizando alguns desenvolvimentos legados da organização antiga fazendo com que possamos continuar refatorando até que não necessita do desenvolvimento legado.

08 -

a) O padrão composite realiza a identificação de um componente do sistema de maneira uniforme, pois não é preciso verificar de forma explícita a complexidade de um objeto, por agrupar (sendo a classe *Main* é responsável pelo agrupamento) todos esses objeto como se fossem uma árvore, o que facilita bastante na manipulação de objetos mais complexos

b) Sim, pode-se ser implementado sim quando a maioria dos objetos são folhas sem filhos, ele teria a mesma implementação padrão do Composite, funcionando com uma classe base que é a interface do Composite, as folhas em sua maioria implementadas de forma direta, enquanto os outros objetos que têm filhos, serão estruturados para armazenar e operar os filhos.

9-