



Testes com Design Patterns

Helder da Rocha
(helder.darocha@gmail.com)
31 de março de 2005

Testes com Design Patterns

71. Que padrão de design pode ser usado para permitir que uma implementação específica e uma hierarquia de abstrações possa variar independentemente?

- a) Adapter
- b) Proxy
- c) Façade
- d) Bridge
- e) Flyweight



Testes com Design Patterns

71. Que padrão de design pode ser usado para permitir que uma implementação específica e uma hierarquia de abstrações possa variar independentemente?

- a) Adapter
- b) Proxy
- c) Façade
- d) Bridge**
- e) Flyweight



Testes com Design Patterns

72. Qual o melhor padrão de design para gerenciar a interação entre objetos que precisam trocar informações entre si mas não podem ter nenhum acoplamento?

- a) Observer
- b) Façade
- c) Mediator
- d) Chain of Responsibility
- e) State



Testes com Design Patterns

72. Qual o melhor padrão de design para gerenciar a interação entre objetos que precisam trocar informações entre si mas não podem ter nenhum acoplamento?

- a) Observer
- b) Façade
- c) Mediator**
- d) Chain of Responsibility
- e) State



Testes com Design Patterns

73. Qual o padrão de design que permite encapsular instruções em objetos para que um cliente possa executar uma ação usando o mesmo método sem precisar saber exatamente qual ação está sendo executada?

- a) Strategy
- b) State
- c) Command
- d) Factory Method
- e) Template Method



Testes com Design Patterns

73. Qual o padrão de design que permite encapsular instruções em objetos para que um cliente possa executar uma ação usando o mesmo método sem precisar saber exatamente qual ação está sendo executada?

- a) Strategy
- b) State
- c) Command**
- d) Factory Method
- e) Template Method



Testes com Design Patterns

74. Que padrão de design abaixo pode ser utilizado quando existe a necessidade de lidar com uma grande quantidade de objetos e a possibilidade de se reutilizar instâncias para tornar mais eficiente a utilização de recursos (por exemplo, na implementação de um cache)?
- a) Adapter
 - b) Proxy
 - c) Façade
 - d) Bridge
 - e) Flyweight



Testes com Design Patterns

74. Que padrão de design abaixo pode ser utilizado quando existe a necessidade de lidar com uma grande quantidade de objetos e a possibilidade de se reutilizar instâncias para tornar mais eficiente a utilização de recursos (por exemplo, na implementação de um cache)?

- a) Adapter
- b) Proxy
- c) Façade
- d) Bridge
- e) Flyweight**



Testes com Design Patterns

81. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Façade?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne uma única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

81. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Façade?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes**
- d) Um cliente precisa de uma interface que retorne uma única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

82. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Singleton?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

82. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Singleton?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente**
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

83. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Adapter?

- a) Um cliente precisa de uma interface que é diferente da fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

83. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Adapter?

- a) Um cliente precisa de uma interface que é diferente da fornecida pela classe existente**
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

85. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Proxy?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

85. Qual das situações abaixo é o cenário típico onde poderia ser utilizado um Proxy?

- a) Um cliente precisa de uma interface que é diferente da interface fornecida pela classe existente
- b) Um cliente precisa de uma interface idêntica à da classe existente mas não tem acesso direto a ela**
- c) Um cliente precisa de uma interface que simplifique o acesso a uma hierarquia de classes
- d) Um cliente precisa de uma interface que retorne a única instância de uma classe existente
- e) Um cliente precisa ser notificado sobre alterações no estado de objetos



Testes com Design Patterns

91. Que padrão de design pode ser usado para garantir que um objeto só tenha uma única instância?

- a) Façade
- b) Abstract Factory
- c) Factory Method
- d) Iterator
- e) Singleton



Testes com Design Patterns

91. Que padrão de design pode ser usado para garantir que um objeto só tenha uma única instância?

- a) Façade
- b) Abstract Factory
- c) Factory Method
- d) Iterator
- e) **Singleton**



Testes com Design Patterns

92. Que padrão de design permite a criação de objetos através de um método genérico sem que o cliente precise ter conhecimento de qual implementação concreta está sendo usada?

- a) Template Method
- b) Abstract Factory
- c) Factory Method
- d) Iterator
- e) Singleton



Testes com Design Patterns

92. Que padrão de design permite a criação de objetos através de um método genérico sem que o cliente precise ter conhecimento de qual implementação concreta está sendo usada?
- a) Template Method
 - b) Abstract Factory
 - c) Factory Method**
 - d) Iterator
 - e) Singleton



Testes com Design Patterns

93. Uma classe possui um método concreto e final que implementa um algoritmo de compressão de dados. O algoritmo é escrito em função de outros métodos que são chamados em diferentes etapas da compressão. Estes outros métodos podem ser sobrepostos em diferentes subclasses para prover versões personalizadas do algoritmo de compressão. Que padrão de design é representado pelo método concreto?

- a) Factory Method
- b) Command
- c) Builder
- d) Template Method
- e) Proxy



Testes com Design Patterns

93. Uma classe possui um método concreto e final que implementa um algoritmo de compressão de dados. O algoritmo é escrito em função de outros métodos que são chamados em diferentes etapas da compressão. Estes outros métodos podem ser sobrepostos em diferentes subclasses para prover versões personalizadas do algoritmo de compressão. Que padrão de design é representado pelo método concreto?

- a) Factory Method
- b) Command
- c) Builder
- d) Template Method**
- e) Proxy



Testes com Design Patterns

94. Que padrão de design é implementado pelas classes da API Java `java.io`: `FileInputStream`, `ObjectInputStream` e `GZIPInputStream` que permitem ler bytes de um arquivo e transformar os dados durante a leitura para obtê-los já descomprimidos e instanciados como objetos?
- a) Decorator
 - b) Memento
 - c) Visitor
 - d) Prototype
 - e) Singleton



Testes com Design Patterns

94. Que padrão de design é implementado pelas classes da API Java `java.io`: `FileInputStream`, `ObjectInputStream` e `GZIPInputStream` que permitem ler bytes de um arquivo e transformar os dados durante a leitura para obtê-los já descomprimidos e instanciados como objetos?

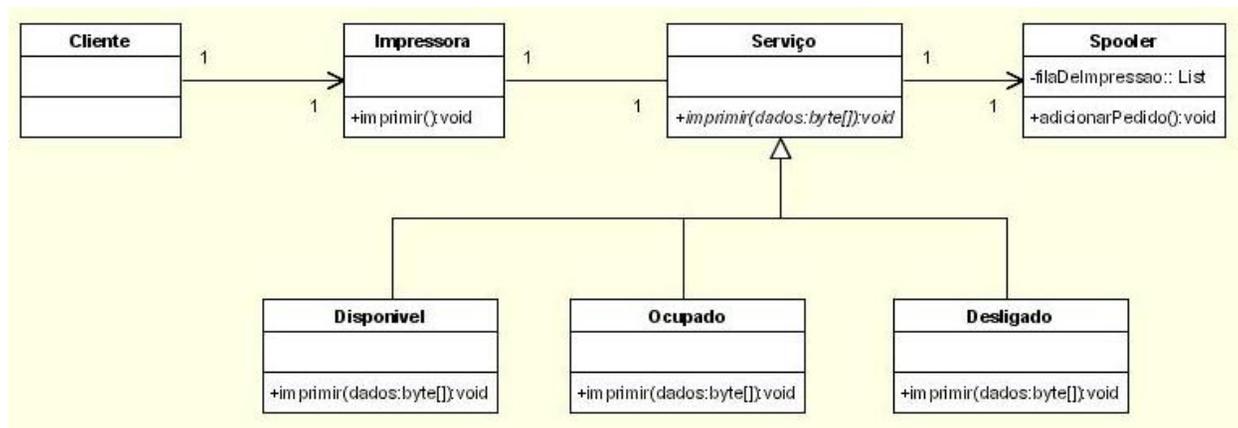
- a) **Decorator**
- b) Memento
- c) Visitor
- d) Prototype
- e) Singleton



Testes com Design Patterns

101. Um sistema de impressão utiliza o objeto Serviço para controlar quando, ao receber uma ordem de impressão, uma tarefa será enviada diretamente para a impressora ou para a fila de impressão. O diagrama UML abaixo ilustra o modelo de implementação usado. Que padrão de design foi utilizado?

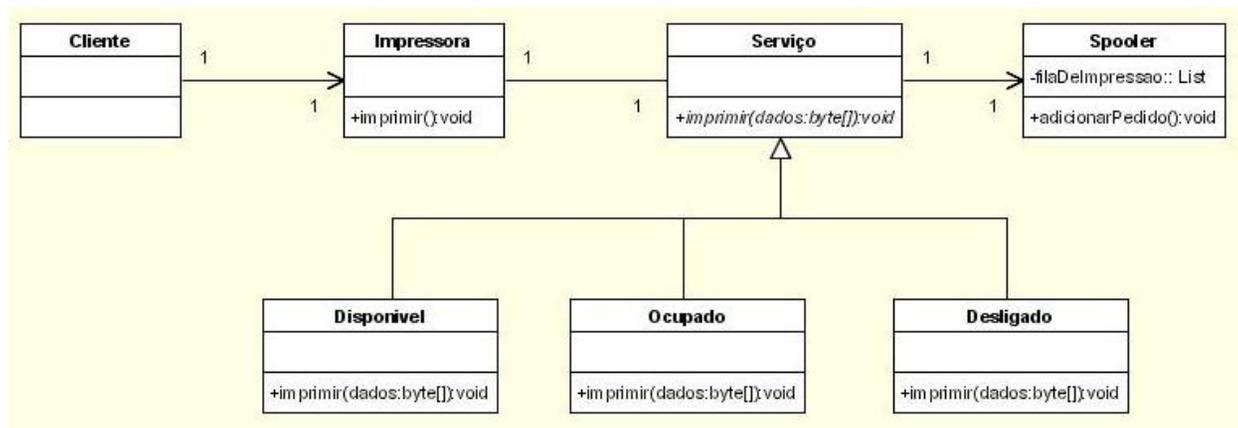
- a) Command
- b) Interpreter
- c) State
- d) Strategy
- e) Chain of Responsibility



Testes com Design Patterns

101. Um sistema de impressão utiliza o objeto Serviço para controlar quando, ao receber uma ordem de impressão, uma tarefa será enviada diretamente para a impressora ou para a fila de impressão. O diagrama UML abaixo ilustra o modelo de implementação usado. Que padrão de design foi utilizado?

- a) Command
- b) Interpreter
- c) State**
- d) Strategy
- e) Chain of Responsibility

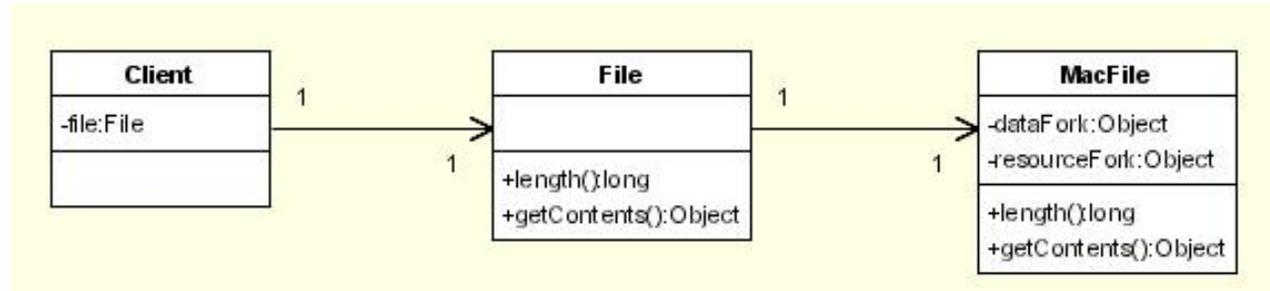


Testes com Design Patterns

111. A classe MacFile abaixo representa um arquivo em um sistema MacOS 9.x e implementa a mesma interface que a classe File, que representa um arquivo genérico e intermedia a comunicação.

Um objeto Client pode manipular um objeto File sem ter acesso ao objeto MacFile. Que padrão de design é implementado pelo objeto File?

- a) Adapter
- b) Proxy
- c) Mediator
- d) Flyweight
- e) Façade

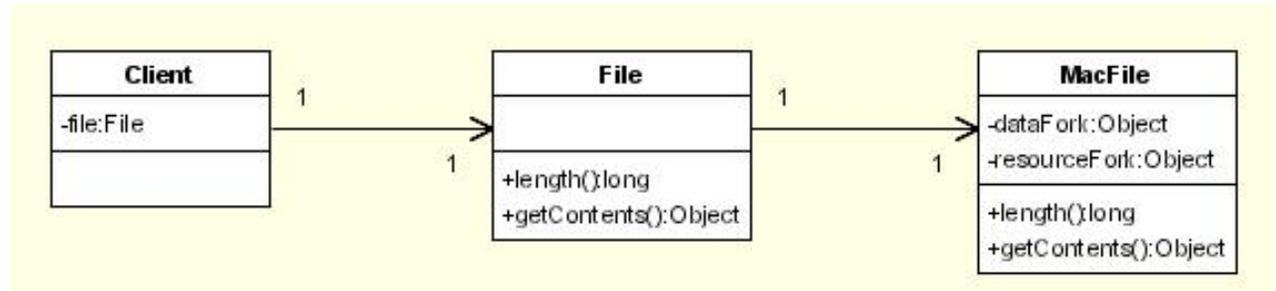


Testes com Design Patterns

111. A classe MacFile abaixo representa um arquivo em um sistema MacOS 9.x e implementa a mesma interface que a classe File, que representa um arquivo genérico e intermedia a comunicação.

Um objeto Client pode manipular um objeto File sem ter acesso ao objeto MacFile. Que padrão de design é implementado pelo objeto File?

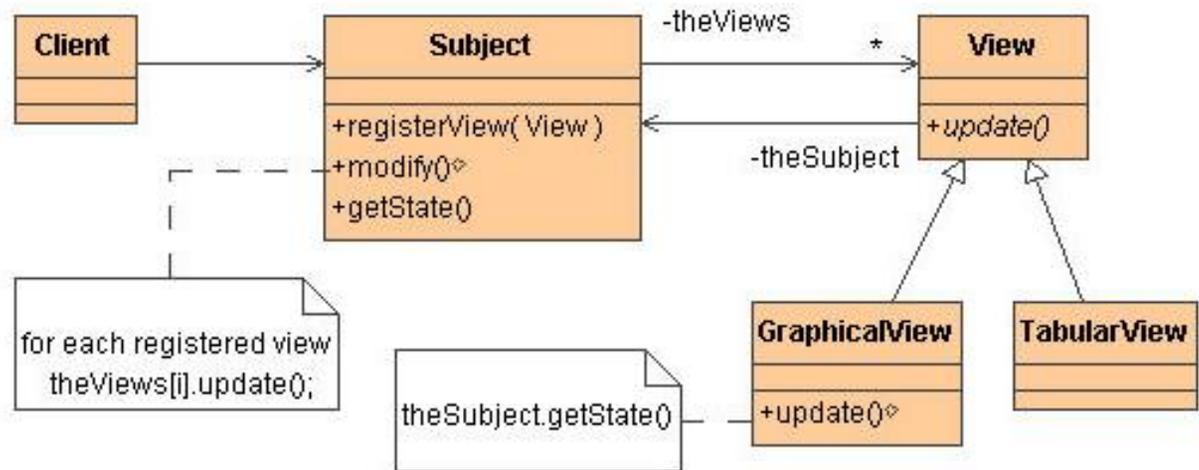
- a) Adapter
- b) Proxy**
- c) Mediator
- d) Flyweight
- e) Façade



Testes com Design Patterns

121. A classe Subject abaixo mantém uma coleção de objetos View, que exibem dados em uma interface gráfica. Quando o método modify() é chamado, todos os View da coleção têm seus métodos update() chamados. Que padrão é representado pelo objeto View?

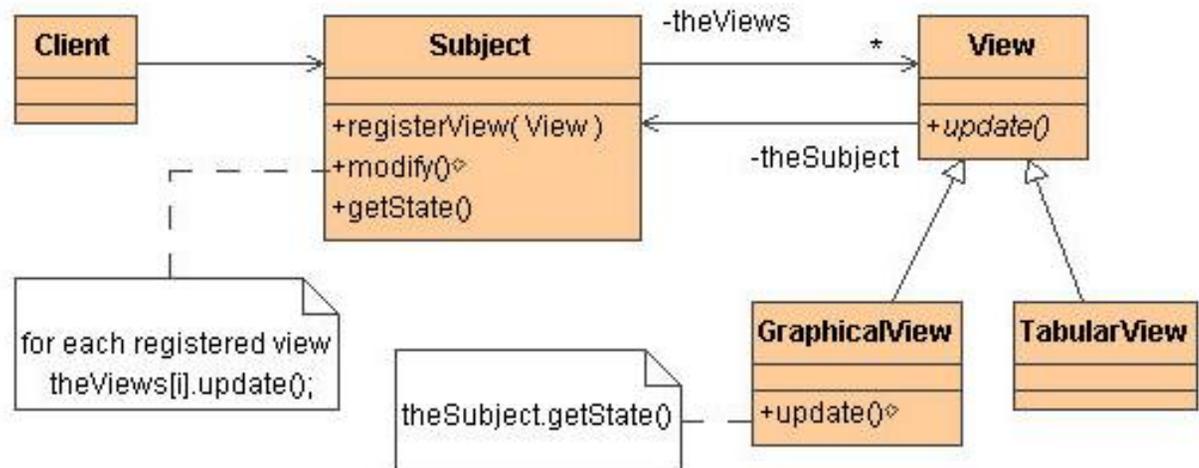
- a) Mediator
- b) Proxy
- c) Adapter
- d) Observer
- e) Façade



Testes com Design Patterns

121. A classe Subject abaixo mantém uma coleção de objetos View, que exibem dados em uma interface gráfica. Quando o método modify() é chamado, todos os View da coleção têm seus métodos update() chamados. Que padrão é representado pelo objeto View?

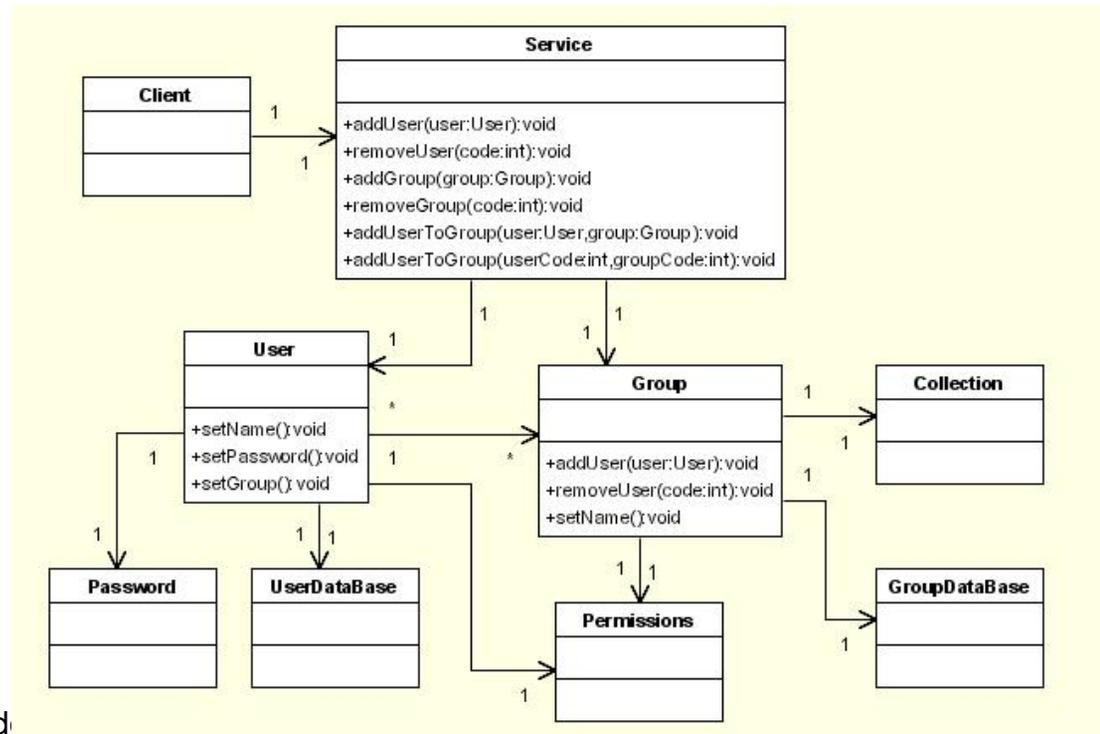
- a) Mediator
- b) Proxy
- c) Adapter
- d) Observer**
- e) Façade



Testes com Design Patterns

131. A classe Service abaixo concentra todas as operações que podem ser executadas pela classe Client, simplificando a interface da aplicação. Que padrão de design é representado por esta classe?

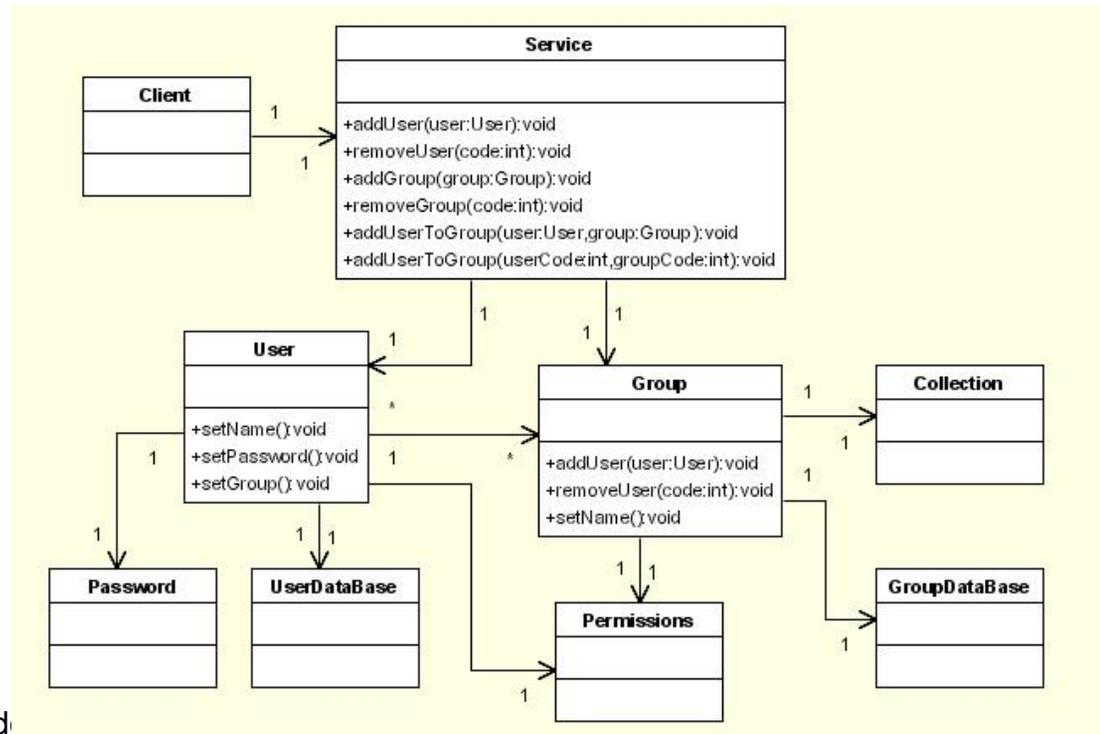
- a) Adapter
- b) Command
- c) Strategy
- d) Singleton
- e) Façade



Testes com Design Patterns

131. A classe Service abaixo concentra todas as operações que podem ser executadas pela classe Client, simplificando a interface da aplicação. Que padrão de design é representado por esta classe?

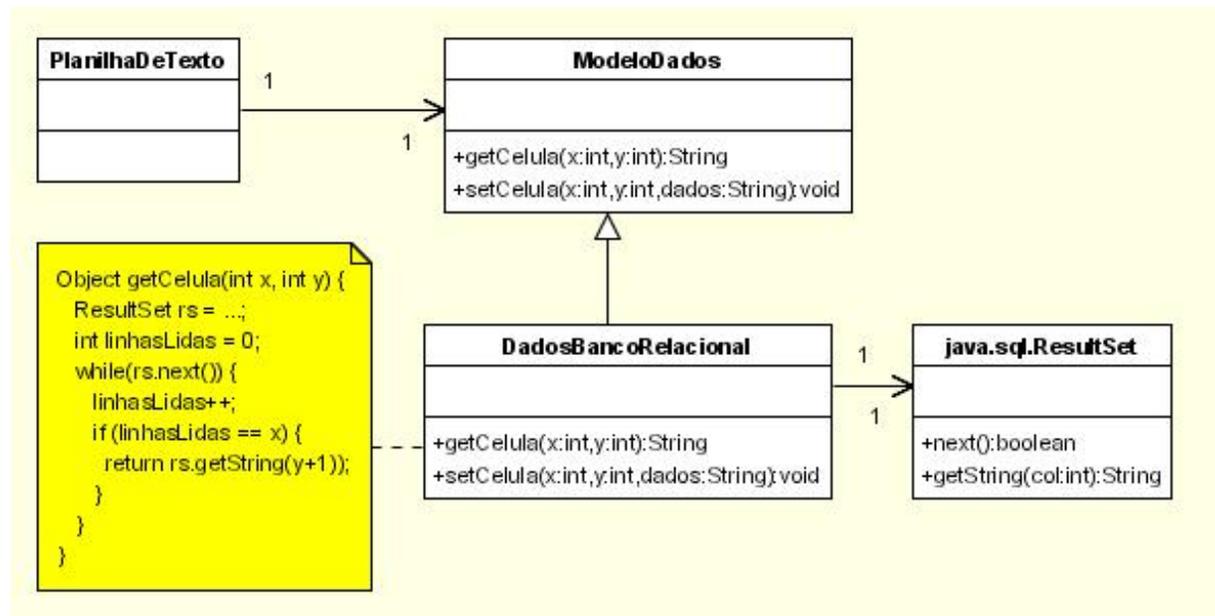
- a) Adapter
- b) Command
- c) Strategy
- d) Singleton
- e) Façade**



Testes com Design Patterns

141. Que padrão de design é representado pela classe DadosBancoRelacional no diagrama UML abaixo?

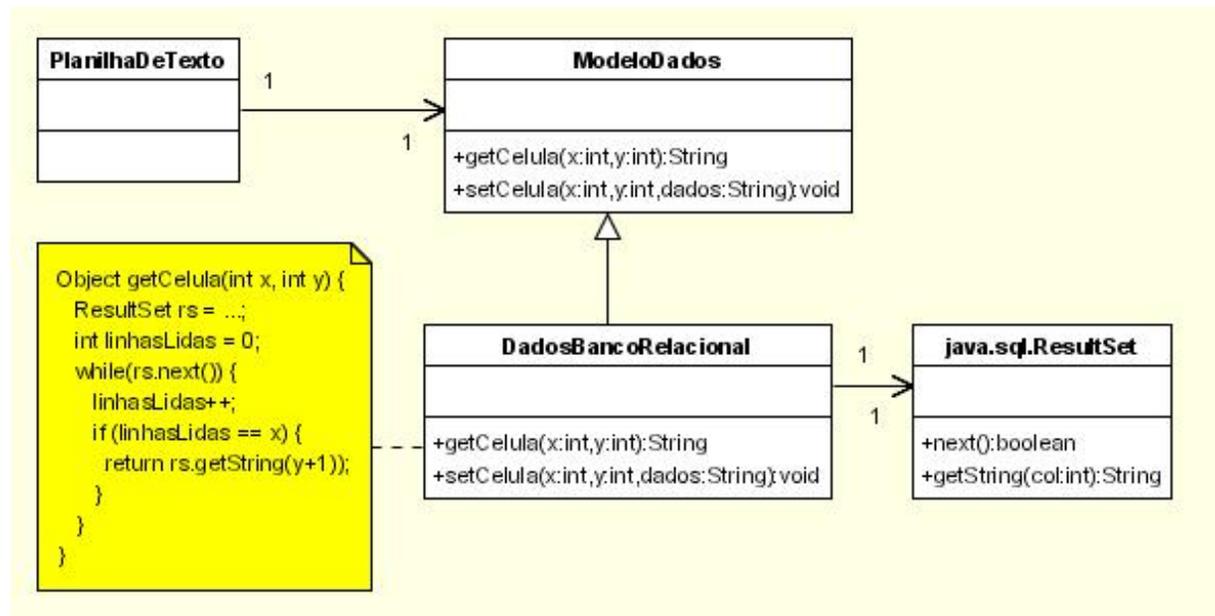
- a) Proxy
- b) Decorator
- c) Adapter
- d) Composite
- e) Façade



Testes com Design Patterns

141. Que padrão de design é representado pela classe DadosBancoRelacional no diagrama UML abaixo?

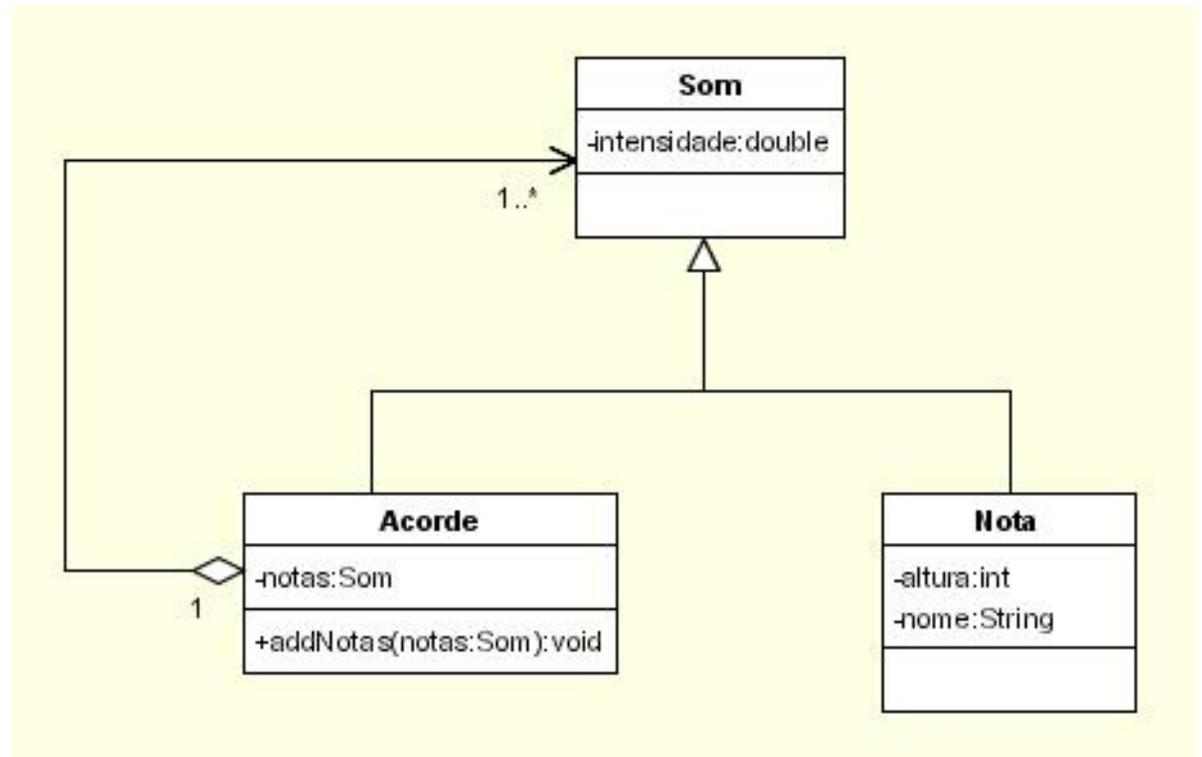
- a) Proxy
- b) Decorator
- c) Adapter**
- d) Composite
- e) Façade



Testes com Design Patterns

151. Qual padrão de design está representado no diagrama UML abaixo?

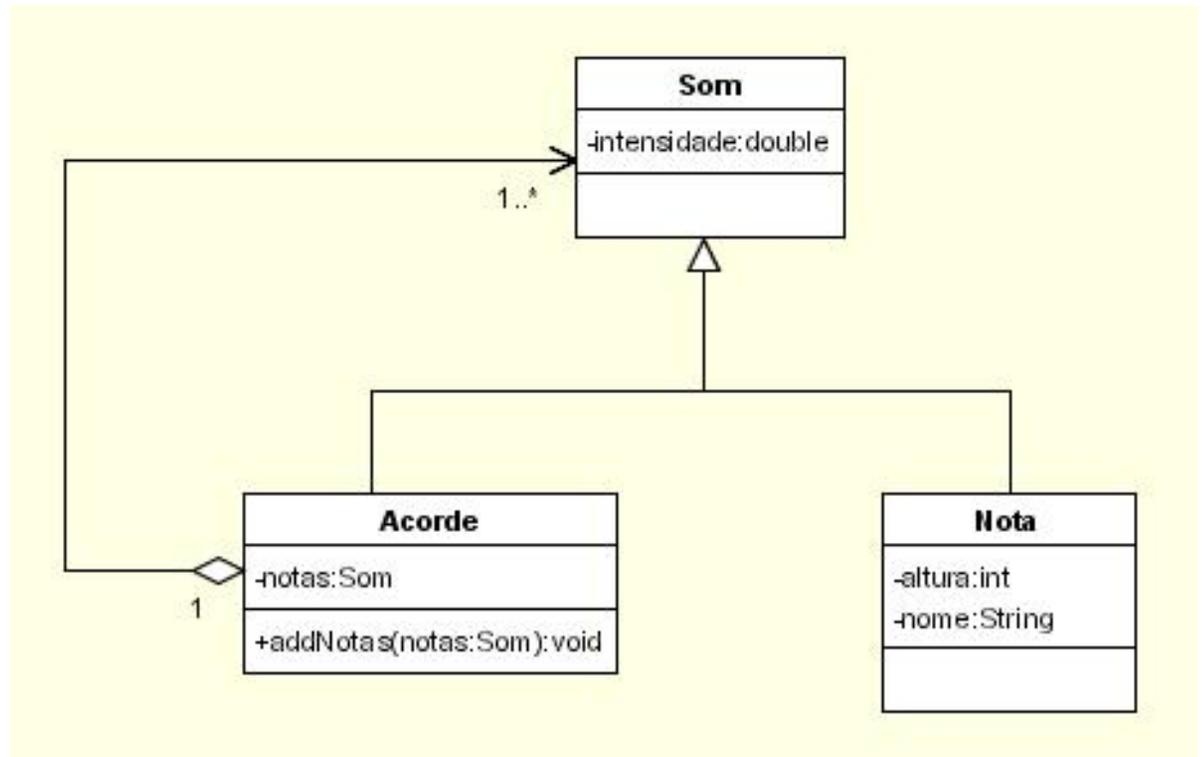
- a) Adapter
- b) Proxy
- c) Mediator
- d) Composite
- e) Façade



Testes com Design Patterns

151. Qual padrão de design está representado no diagrama UML abaixo?

- a) Adapter
- b) Proxy
- c) Mediator
- d) Composite**
- e) Façade





www.argonavis.com.br

Visite o site e baixe outros materiais sobre
Java, XML, HTML, CSS, OO, metodologias
ágeis e tecnologias relacionadas