Bootcamp Java 14/10/22

Associação

Estrutural

Composição com parte todo

Ex: pessoa e endereço

Uma parte única pertence a um todo, caso o todo deixe de existir a parte também deixa.

Gráfico

Descrição gerada automaticamente

Em código

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Agregação: sem parte todo

Ex: disciplina aluno

Caso o todo deixe de existir a parte não deixa, já que ela pode fazer parte de mais de um todo

Em código

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Agregação x Composição

Não há melhor ou pior, e são usadas de acordo com a necessidade

Comportamental

Dependência: depende de.

O método de finalizar compra depende do objeto cupom

Gráfico

Descrição gerada automaticamente com confiança baixa

Em código

Texto, Carta

Descrição gerada automaticamente

Herança x Associação

Não há um melhor, e cada um deve ser usado de acordo com a necessidade.

A herança é mais rígida, só é feita no momento do desenvolvimento.

A associação podemos mudar, ela é mais flexível

Exercícios



Interface

Define um contrato que deve ser seguido pela classe que a implementa. Quando uma classe implementa uma interface, ela se compromete a realizar todos os comportamentos que a interface disponibiliza.

A interface possui seus métodos próprios, que são inseridos dentro da classe. Para realizar as operações deles.

Em código

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Exercícios

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Siga em Frente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Organização de Pacotes e Visibilidade

Objetivos

* Apresentar os conceitos que organizam a OO: pacotes e visibilidade

Pacotes

Organização física ou lógica criada para separar classes com responsabilidades distintas. Com isso, espera-se que a aplicação fique mais organizada e seja possível separar classes de finalidades e representatividades diferentes.

Em código

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Exercício

Ver projeto de exemplo para entender funcionamento.

Visibilidades

É um modificador de acesso que determina até que ponto uma classe, atributo ou método pode ser usado. A utilização de modificadores de acesso é fundamental para o uso efetivo da orientação a objetos. Algumas boas práticas e conceitos só são atingidos com o uso correto deles.

Tipos:

* Private
* Protected
* Public

Private

Mais restritiva, somente visível dentro da classe

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Para isso que serve os getters and setter, para acessar o método que usa os valores privados e retorna o que foi pedido

Protected

Dentro da classe, mesmo pacote e subclasses. Ou seja, pode ser acessada por esses 3

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Public

Visível em qualquer lugar

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Curiosidade

Qual usar: cada visibilidade possui uma finalidade, as privadas são mais usadas no encapsulamento e segurança dos dados. Para se comunicar com outros são usados métodos que devem ser públicos ou protegidos.

Exercício

Assistir

Próximos passos

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente