Pablo Hugo Freitas Alves - 540193

Introdução

Este pequeno resumo apresenta alguns conceitos e definições sobre a programação orientada a objetos.
Serão abordados os seguintes tópicos:

- Objetos
- Classes
- Classes Abstratas
- Atributos
- Métodos
- Métodos especiais getters e setters
- Construtores
- Sobrecarga de métodos
- Visibilidade
- Encapsulamento
- Herança
- Sobrescrita de métodos
- Relação de objetos
- Polimorfismo

Objetos

Objetos são a principal entidade na programação orientada a objetos. Por definição, objetos são uma abstração de entidades da vida real: casas, pessoas, veículos, canetas, todos eles são objetos para esse paradigma. Com essa abstração, o programador é capaz de organizar seu código de maneira mais coerente e lógica com o mundo palpável. Programar nesse paradigma é trazer a vida real ao software.

Classes

Classes são os moldes dos objetos, elas dão atributos e ações para os objetos. Nesse olhar, as classes definem o que um objeto tem e o que ele pode fazer em seu software.

Classes Abstratas

Classes abstratas são um tipo especial de classe. Esse tipo de classe não pode ser instanciada. O principal objeto dessa classe é representar conceitos que não são concretos em um sistema.

Atributos

Atributos são as prioridades relevantes dos Objetos da vida real representados em Objetos dentro do domínio do seu software. Pessoas têm nome, idade, CPF, certidão de nascimento… entre outras características. Em software de banco, uma Pessoa pode ser somente um CPF. Em contrapartida, em um software de adoção, uma Pessoa tem atributos diferentes para suprir a necessidade do domínio que está presente.

Métodos

Métodos são ações que seu objeto pode realizar. O objeto pode tanto modificar seu estado atual quanto interagir com outros objetos por meio dos métodos.

Métodos especiais getters e setters

Em programação orientada a objetos, há uma nomenclatura especial para métodos que retornam os valores de atributos e para os métodos que definem os valores dos atributos.

- getters: são métodos que retornam valores de atributos do objeto. Normalmente, eles iniciam com o prefixo *get;*
- setters: são métodos que definem valores de atributos do objeto. Normalmente, eles iniciam com o prefixo *set*.

Construtor

Construtor é um método especial que leva o nome da classe. Esse método especial, diz o quê um objeto precisa para ser inicializado. É possível chamar métodos dentro do construtor.

Sobrecarga de métodos

Sobrecarga de métodos é um meio de escrever diferentes comportamentos, métodos, com o mesmo nome. Porém, para cada novo método sobrecarregado, é necessário que a assinatura da função seja diferente.

Visibilidade

Na criação de um objeto, podemos definir quais atributos e métodos podem ser acessados por outros objetos. São utilizados os seguintes modificações de visibilidade:

- **public**: qualquer objeto pode acessar a propriedade ou método;
- **private**: somente o próprio objeto tem acessa a propriedade ou método;
- **protected**: somente objetos no mesmo pacote tem acesso a propriedade ou método.

Encapsulamento

Encapsulamento é um conceito pilar em programação orientada a objetos. Encapsular informação, ou seja, manter a privacidade de atributos e métodos que interessa somente o próprio objeto ou ainda proteger propriedades e comportamentos que são convenientes somente ao próprio pacote é o que é chamado encapsulamento.

Herança

Em programação orientada a objetos, é possível que as classes sejam filhas de outras classes. Assim como a herança genética, as filhas herdam atributos e métodos da classe pai. Herança possibilita a escrita e modificação facilitada do código. Por isso, é um dos pilares da orientação a objetos.

Sobrescrita de métodos

Quando um objeto herda outro objeto, é possível que o filho sobrescreva os comportamentos do pai.Então, o filho é capaz de mudar “como se faz algo” herdado do pai.

Relação de objetos

Objetos podem(e devem) interagir entre si. As possíveis relações entre os objetos são:

- **Ter**: quando um objeto é composto por outro objeto, ou seja, dentro do objeto há outro objeto;
- **Usar**: quando um objeto outro objeto em suas ações;
- **Ser**: quando dois objetos de classes diferentes herdam de um mesmo pai. Podemos dizer que eles são o mesmo pai. Portanto, onde o pai é aceito, o filho também é.

Polimorfismo

O polimorfismo é um dos pilares fundamentais da programação orientada a objetos. Ele nos permite tratar objetos de diferentes tipos como se fossem do mesmo tipo, desde que compartilhem um conjunto comum de comportamentos. Isso significa que podemos chamar os mesmos métodos em objetos distintos, obtendo resultados adequados para cada um deles. Dessa forma, O polimorfismo nos permite lidar com os dados de forma mais homogênea e flexível.

Resumo

Em resumo, a programação orientada a objetos oferece uma abordagem poderosa e modular para o desenvolvimento de software. Com seus conceitos fundamentais, como abstração, herança, encapsulação e polimorfismo, os desenvolvedores podem criar sistemas mais eficientes, organizados e flexíveis.