RELATÓRIO DE PROGRAMAÇÃO ORIENTADA A OBJETOS

Os 4 pilares da POO



**Eduardo Catunda**

09.06.2019

MDIC - Programador de Dispositivos Móveis

# INTRODUÇÃO

Programação Orientada a Objetos é um dos Paradigmas de Desenvolvimento de programação. Há outros como Programação Estruturada e Programação Orientada a Aspectos. Além de apresentar características da Programação Estruturada, como os conceitos de sequência, seleção e iteração, a POO reaproveita cada elemento do mundo real para o objeto, junto a isso tem-se conceitos que ajudam na leitura, na organização e na reutilização do código, como Encapsulamento e Herança. Nesse sentido, esse paradigma vem ganhando cada vez mais espaço, principalmente em questões voltadas para segurança e reaproveitamento de código, o que é muito importante no desenvolvimento de qualquer aplicação moderna.

# PRINCIPAIS CONCEITOS DO PARADIGMA ORIENTADO A OBJETOS

A POO é muito reconhecida pelos seus métodos e conceitos de organização de códigos, buscando relações entre entidades e distribuindo-as em classes de funções que tenham compatibilidade com cada uma delas, por exemplo. Nesse contexto, evidenciar-se-á os principais conceitos do paradigma orientado a objetos:

## Classe e Objeto

Uma classe é uma forma de definir um tipo de dado em uma linguagem orientada a objeto. Ela é um recurso que ajuda a organizar o código de acordo com o conceito que construímos sobre o mundo real. A classe envolve funções e dados, controlando o acesso a estes, defini-la implica em especificar os seus atributos (dados) e seus métodos (funções). Diversos objetos de tipos diferentes podem ser criados para classes inicializadas, dessa maneira, pode-se atribuir propriedades a eles, chamadas de atributos, e impor ações para um objeto realizar, chamados de métodos.

## Associações de classes

A associação de classes é o mecanismo pelo qual um objeto utiliza os recursos de outro, definindo como as classes interagem entre elas nas aplicações. Essas associações podem acontecer tanto de um para um quanto de um para *N* e de *N* para *N*.

# OS 4 PILARES DA POO

O paradigma da Programação Orientada a Objetos constitui em 4 critérios

## Encapsulamento

O encapsulamento é usado para esconder detalhes de implementação do objeto. Funcionando como uma “caixa-preta” do código, assim como nos aviões. Com o encapsulamento de dados, é possível acessar o valor armazenado de uma variável em qualquer parte do código, sem a necessidade de uma permissão.

## 

## Herança

A herança, assim como o DNA dos seres vivos, tem a capacidade de idealizar um objeto baseado em outro objeto. Os objetos pai podem passar seus atributos(propriedades) e seus métodos(funções) para os objetos filho, assim como esse objeto-pai pode ter tido as características herdadas de um objeto anterior. Esse pilar é de suma importância para a reutilização de código da Programação Orientada a Objetos.

## Polimorfismo

O polimorfismo atua em objetos que sejam compatíveis entre si permitindo que um objeto se passe por outro. Pode-se misturar os objetos, alterar os comportamentos, as funções e os atributos de qualquer maneira. Em outras palavras, o polimorfismo consiste na alteração do funcionamento interno de um método herdado de um objeto pai.

## Abstração

A abstração é essencial para a Programação Orientada a Objeto. Ela vai implicar a um objeto sua função dentro do código. Basicamente, a abstração se assemelha à união do encapsulamento com o polimorfismo, devido a sua capacidade de assimilar objetos entre si e chamar uma função de qualquer parte do código, desde que essa esteja em uma superclasse. Nela, dá-se a identidade de um objeto, sendo essa identidade única dentro do sistema para que não haja conflito; dá-se as propriedades, atributos dos objetos e, também, os métodos, ações que os objetos irão realizar.

# CONCLUSÃO

A representação da Programação Orientada a Objetos por classes e objetos aproxima-se ao que se pensa de um objeto real, com suas funções e características específicas. Esse tipo de representação traz algumas vantagens muito interessantes para os desenvolvedores e também para o usuário da aplicação. A reutilização de código é uma delas, permitindo um programa mais limpo e prático. Com a complexidade dos sistemas cada vez maior, o tempo de desenvolvimento iria aumentar exponencialmente caso não fosse possível a reutilização. Outrossim, a facilidade de leitura e manutenção do código faz da POO um paradigma sociável. Isso permite que a equipe de desenvolvimento não fique dependente de uma pessoa apenas, como acontecia com frequência em linguagens estruturadas, por exemplo. Contudo, na Programação Orientada a Objetos a execução é mais lenta do que na programação estruturada. Isso acontece devido à complexidade do modelo, que traz representações na forma de classes, como se pode ver a seguir:



Essa modelagem compromete na velocidade de execução de códigos. Entretanto, essa lentidão não é muito sentida, o que faz crescer ainda mais sua popularidade devido às vantagens dessa programação que suporta muitas linguagens. Como por exemplo, as 3 mais utilizadas no momento: Java, C# e C++. Cada uma delas possui uma abordagem diferente do problema que as torna muito boas para alguns tipos de aplicações e não tão boas para outros.

# REFERÊNCIAS

DEVMEDIA. **Os 4 pilares da Programação Orientada a Objetos.** Disponível em: <https://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264>. Acesso em: 09 jun. 2019.

Código Fonte TV. **Programação Orientada a Objetos (POO) // Dicionário do Programador**. Disponível em: <https://www.youtube.com/watch?v=QY0Kdg83orY>. Acesso em: 09 jun. 2019.

DEVMEDIA. **Principais conceitos da Programação Orientada a Objetos.** Disponível em: <https://www.devmedia.com.br/principais-conceitos-da-programacao-orientada-a-objetos/32285>. Acesso em: 09 jun. 2019.

Fábrica de Software. **Classes, Objetos, Atributos e Métodos em Java**. Disponível em: <http://fabrica.ms.senac.br/2015/03/classes-objetos-atributos-e-metodos-em-java/>. Acesso em: 09 jun. 2019.