




# Classes Abstratas, interfaces e princípios da Orientação a Objetos

## Pilares da Orientação a Objeto?

**Encapsulamento:** serve para controlar o acesso aos atributos e métodos de uma classe. É uma forma eficiente de proteger os dados manipulados dentro da classe, além de determinar onde esta classe poderá ser manipulada. Em uma classe na linguagem Java por exemplo, usamos os métodos Getters and Setter que tem o objetivo de controlar o acesso a cada um dos atributos e operações de uma certa classe. Ou seja, tem a função de disponibilizar externamente os métodos que alteram e acessam (lê) os atributos de uma classe.

Métodos getters	Métodos setters
<pre>public String getNome() {     return nome; }</pre>	<pre>public void setNome(String nome) {     this.nome = nome; }</pre>
<pre>public double getSalario() {     return salario; }</pre>	<pre>public void setSalario(double salario) {     this.salario = salario; }</pre>

## Interfaces em Java

Como uma classe, uma interface pode ter métodos e variáveis, mas os métodos declarados em uma interface são abstratos por padrão (apenas assinatura  método, sem corpo).

- As interfaces especificam o que uma classe deve fazer e não como. É o projeto da classe.
- Ela especifica um conjunto de métodos que a classe deve implementar.
- Se uma classe implementa uma interface e não fornece implementação de método para todas as funções especificadas na interface, a classe deve ser declarada abstrata.
- Um exemplo de biblioteca Java é, **Comparator Interface** . Se uma classe implementa essa interface, ela pode ser usada para classificar uma coleção.

Sintaxe para declaração de interface

```
interface {  
  
//métodos  
  
}
```

Para usar uma interface em sua classe, acrescente a palavra-chave “implements” após o nome da classe, seguido pelo nome da interface.

Exemplo para implementação de interface

```
class Cachorro implements Animal
```



## Classe Abstrata

Uma classe que é declarada com a palavra-chave `abstract` é conhecida como classe abstrata em **Java**. Pode haver métodos abstratos e não abstratos (método com o corpo).

Antes de aprender a classe abstrata Java, vamos primeiro entender a abstração em Java. Abstração é um processo de ocultar os detalhes de implementação e mostrar apenas a funcionalidade ao usuário.

De outra forma, mostra apenas o essencial ao usuário e oculta os detalhes internos, por exemplo, enviar SMS onde você digita o texto e envia a mensagem. Você não conhece o processamento interno sobre a entrega da mensagem. A abstração permite que você se concentre no que o **objeto** faz em vez de em como o faz.

Maneiras de alcançar a abstração

Existem duas maneiras de obter abstração em java

1) Classe abstrata

2) Interface

Pontos para lembrar

- Uma classe abstrata deve ser declarada com uma palavra-chave `abstract`.
- Ele pode ter métodos abstratos e não abstratos.

- Não pode ser instanciado.



- Ele também pode ter **construtores** e métodos estáticos.
- Podem haver métodos finais que forçam a subclasse a não alterar o corpo do método.

### Atividade extra

Para saber mais sobre Herança e Composição leia o artigo de José Carlos Macoratti “Quando usar herança e composição” no site do próprio autor ([Macoratti.net](http://Macoratti.net))

### Referências Bibliográficas

Gilleanes T. A. Guedes. **UML 2 - Uma Abordagem Prática**. NovaTec, 2018.

Grady Booch. **Uml - Guia do Usuário**. Editora Campos, 2018.

Ian Sommerville. **Engenharia de software**. Pearson, 2015.

Roger Pressman, Bruce Maxim. **Engenharia de Software**. Bookman, 2010.

**Ir para exercício**