



Programação Orientada a Objetos

Webaula 3

Reutilização de Classes em Java



Quem já desenvolveu programas de computador certamente já desejou reaproveitar no programa atual algumas estruturas criadas em programas anteriores, com o mínimo de adaptações. Assim, nesta webaula abordaremos os recursos para reutilização de classes que o Java oferece, como:

Modificadores de acesso

Herança

Polimorfismo



Encapsulamento



Modificadores de acesso

Um dos pontos mais interessantes da programação orientada a objetos é seu suporte ao encapsulamento e ocultação de dados. Essa característica é implementada por meio dos modificadores de acesso, aplicados em classes, métodos e campos. Com ela, os campos das classes ficam protegidos de alterações indevidas de outros programadores.

A execução de um método é também conhecida como envio de mensagens a objetos. Na linha `hoje.inicializaDataSimples(umDia, umMês, umAno)`, a classe está enviando a mensagem `inicializaDataSimples` ao objeto `hoje`, com os argumentos `umDia`, `umMês`, `umAno`.





Há quatro modificadores de acesso disponibilizados pelo Java. Clique nas abas para conhecê-los.

public	private	protected	Sem modificadores
--------	---------	-----------	-------------------

Garante que o campo ou método da classe, declarado com esse modificador, poderá ser acessado ou executado a partir de qualquer outra classe.





Herança

Herança é uma característica do paradigma de orientação a objetos por meio da qual um objeto filho herda características e comportamentos do objeto mãe. É pela implementação da herança que conseguimos estabelecer relação entre uma classe mãe e uma classe filha, de modo que a segunda se torne uma extensão da primeira, herdando diretamente os métodos e os atributos públicos da classe mãe.

É interessante destacar que herança é sempre utilizada em Java, mesmo que não explicitamente. Quando uma classe é criada e não há nenhuma referência à sua superclasse, implicitamente a classe criada é derivada diretamente da classe Object (RICARTE, 2000).





Polimorfismo

A criação de classes derivadas estabelece uma relação *é-um-tipo-de*. Por exemplo, um Funcionário *é-um-tipo-de* Pessoa e um Aluno De Pós-Graduação *é-um-tipo-de* Aluno. É justamente a relação *é-um-tipo-de* que permite a existência do polimorfismo. Caelum (s.d.) assevera que polimorfismo é a capacidade de um objeto poder ser referenciado de várias formas, o que não quer dizer que esse fica se transformando. Ao contrário, um objeto nasce de um tipo e morre daquele tipo. O que pode mudar é a maneira como nos referimos a ele.





Encapsulamento

Arnold, Gosling e Holmes (2007, p. 41) afirmam que os verdadeiros benefícios da orientação a objetos provêm do ocultamento da implementação de uma classe atrás das operações realizadas sobre seus dados. Ocultar dados atrás de métodos de modo que sejam inacessíveis a outros objetos é a base fundamental do encapsulamento de dados.





Segundo Fernandes (s.d), encapsulamento e ocultação são conceitos diferentes. Observe:

O **encapsulamento** é um conceito da Programação Orientada a Objetos no qual o estado de objetos (as variáveis da classe) e seus comportamentos (os métodos da classe) são agrupados em conjuntos segundo o seu grau de relação.

A **ocultação** está relacionada à restrição de acesso a alguns componentes de objetos, tornando-os disponíveis apenas por meio de métodos.





Vídeo de encerramento



Você já conhece o Saber?

Aqui você tem na palma da sua mão a **biblioteca digital** para sua **formação profissional**.

Estude no celular, tablet ou PC em qualquer hora e lugar sem pagar mais nada por isso.

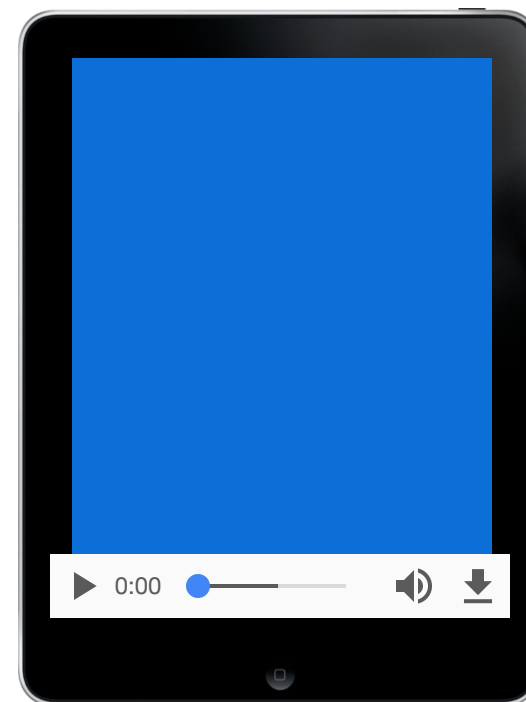
Mais de 450 livros com interatividade, vídeos, animações e jogos para você.



Android:
<https://goo.gl/yAL2Mv>



iPhone e iPad - IOS:
<https://goo.gl/OFWqcq>





Bons estudos!

