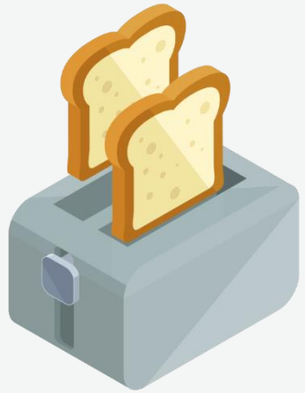
SÃO
PAULO
TECH
SCHOOL

Linguagem de Programação

UML & Encapsulamento

Célia Taniwaki
Diego Brito
Giuliana Miniguiti

Encapsulamento



Conceito importante na Programação Orientada a Objeto

Mecanismo que consiste em:

- Controlar o **acesso** aos atributos e comportamentos de uma classe por parte das outras classes.
- Ocultar os detalhes internos de implementação do restante do sistema.

Analogia: Utilizamos a torradeira, sem precisar conhecer ou alterar seu funcionamento interno, precisamos conhecer apenas a interface (botões para fazê-la funcionar).

Encapsulamento

Em POO também:

- “Mundo externo” (outros objetos) precisam conhecer apenas a interface, ou seja, os métodos que estão definidos na classe como **public**.
- Esses métodos **public** é que controlam e alteram os valores dos atributos definidos na classe.

Vantagem: Assegura que cada objeto mantenha sua identidade e estado e que seus atributos só possam ser alterados por uma interface bem definida.

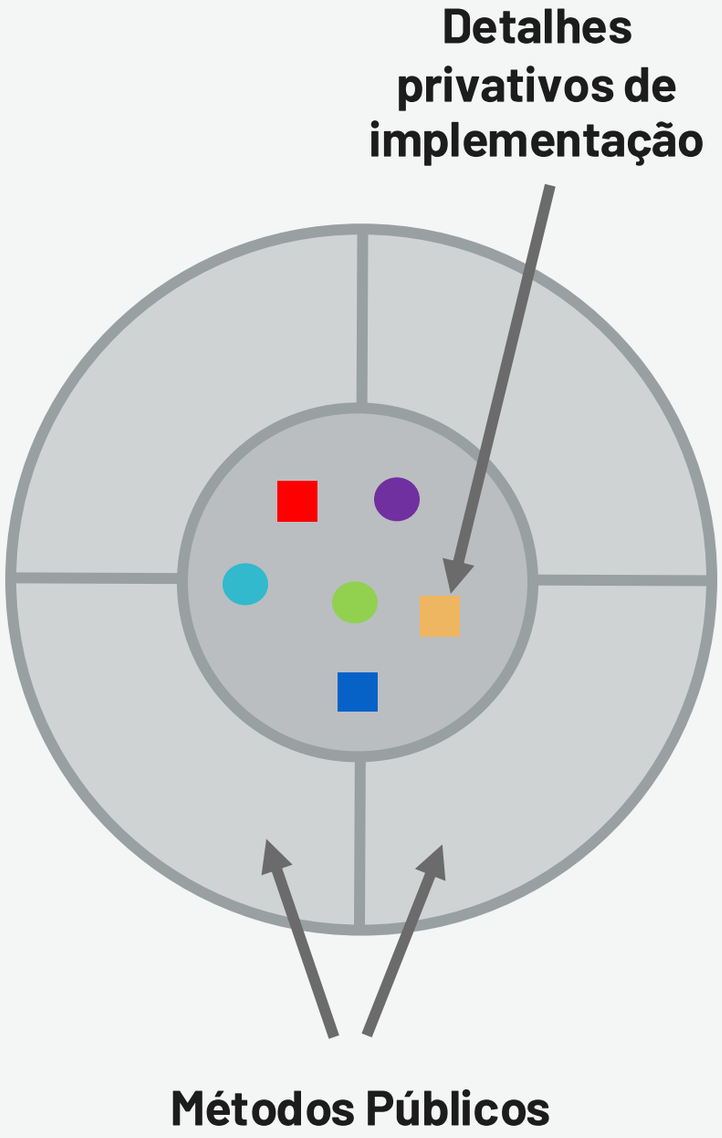
Encapsulamento

Implementado **na definição da classe**

Ideia básica da implementação do encapsulamento:

- Uma classe **nunca deve permitir acesso direto** aos seus atributos ao mundo externo
- A classe deve alterar os atributos como parte de sua funcionalidade ou **prover métodos** para acessar e alterar os atributos.

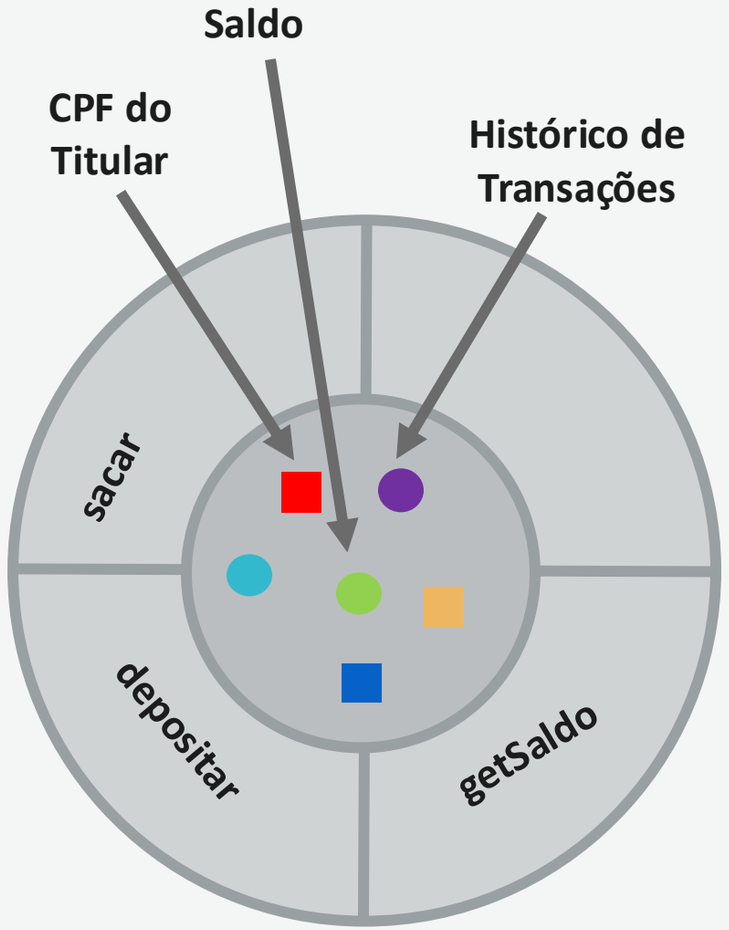
Mantendo uma interface bem definida com o mundo exterior, pode-se modificar facilmente sua implementação interna sem prejudicar o restante do sistema.



Encapsulamento

Exemplo: Classe **ContaBancaria**

- Por segurança, não é interessante permitir que qualquer código do sistema possa ter acesso direto e alterar o valor do saldo (um dos atributos da classe)
- Ideal é ter esse atributo encapsulado
- Quando se encapsula um atributo, só se pode "mexer" nos seus valores através dos métodos dessa classe



Encapsulando os atributos

- Para impedir que um atributo seja acessado de fora da classe deve-se declará-lo como **private**.
- **default** representa quando não se especifica nenhum modificador na declaração do atributo.

Modificadores de acesso também podem ser usados na declaração dos métodos

Modificador	Símbolo UML	Mesma classe	Mesmo pacote	Subclasses	Universo
private	-	*			
default	~	*	*		
protected	#	*	*	*	
public	+	*	*	*	*

Getters e Setters

Prática comum de **encapsulamento**:

- Definir os **atributos** como **private**
- Dessa forma, outras classes não conseguem mais acessar diretamente os valores dos atributos private

Mas, e agora? Como fazer para atribuir valores ou obter valores dos atributos a partir de outras classes?

Prática comum: **oferecer métodos especiais para cada atributo encapsulado**:

- **Getter**: método que **obtém o valor** de um atributo
- **Setter**: método que **configura o valor** de um atributo

Vantagens do Encapsulamento

- **Ocultar** a implementação interna dos usuários
- **Proteger** as informações, não permitindo acesso direto aos atributos
- **Facilitar a programação**, oferecendo uma interface bem definida para quem vai utilizar a classe
- **Facilitar a manutenção** do programa, pois mesmo alterando a implementação interna, a interface continua a mesma
- Facilitar a **consistência dos dados**, já que “centraliza” o ponto onde o valor do atributo é alterado.

UML – Unified Modeling Language

Linguagem Unificada de Modelagem

- Utilizada para facilitar a modelagem de um sistema orientado a objetos, desde a fase inicial de levantamento de requisitos até a fase final do projeto

Ideia básica:

- Oferecer um padrão (visual e textual) para a modelagem do sistema
- Facilitar a comunicação / discussão entre as partes envolvidas: cliente, analista, levantador de requisitos, desenvolvedores, futuros usuários, etc.

UML - Diagramas

Especifica 13 diagramas:

- Para representar as diversas fases e aspectos da modelagem de um sistema
- Exemplos de Diagramas: Diagrama de Casos de Uso, Diagrama de Classes, Diagrama de Sequência, Diagrama de Pacotes, etc.

Em LP, veremos como é o **Diagrama de Classes**

Inicialmente, veremos como uma classe é representada no Diagrama de Classes

Diagrama de Classes

- Cada classe do sistema é representada por um retângulo (“caixa”) com, no máximo, 3 divisões
- Número de compartimentos depende do nível de detalhamento desejado, em cada fase do sistema

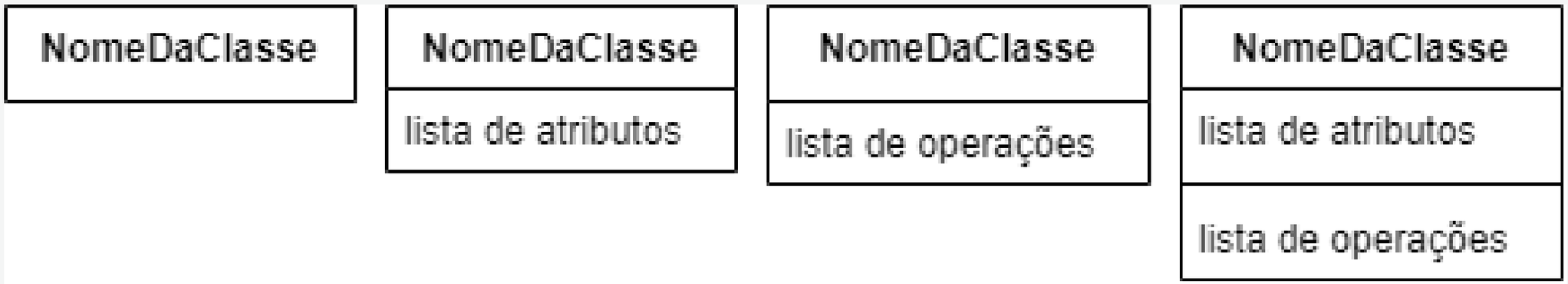
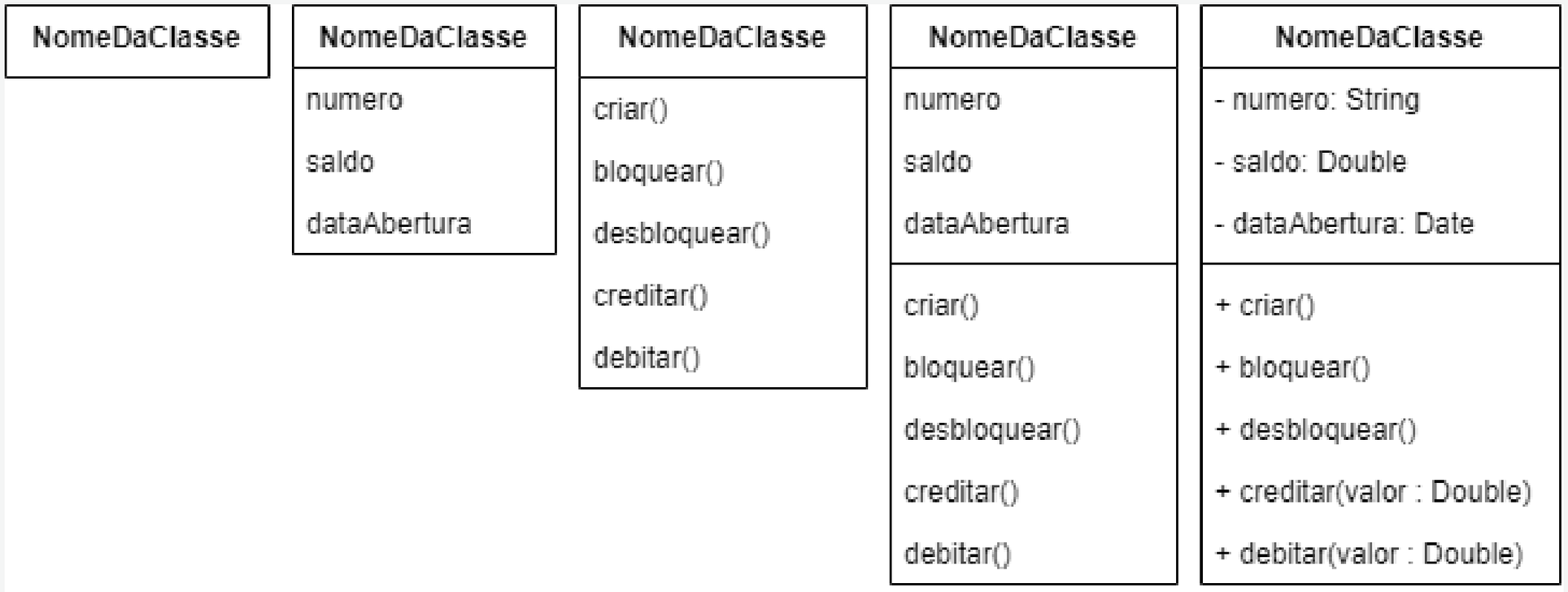


Diagrama de Classes

Exemplo: Classe ContaBancária

Atributos: número, saldo, dataAbertura

Operações: criar, bloquear, desbloquear, creditar, debitar



Agradeço

a sua atenção!

Obrigado!

Em caso de dúvidas, entre em contato com:

diego.lima@sptech.school

giuliana.franca@spetch.school

SÃO PAULO TECH SCHOOL