POO - PROGRAMAÇÃO ORIENTADA A OBJETOS

Classes e Métodos Finais
Atributos e Métodos de Classe

Rodrigo R Silva

Instituto Federal de Educação, Ciência e Tecnologia Sul-Rio-Grandense
Campus Bagé

Nesta Aula Veremos...

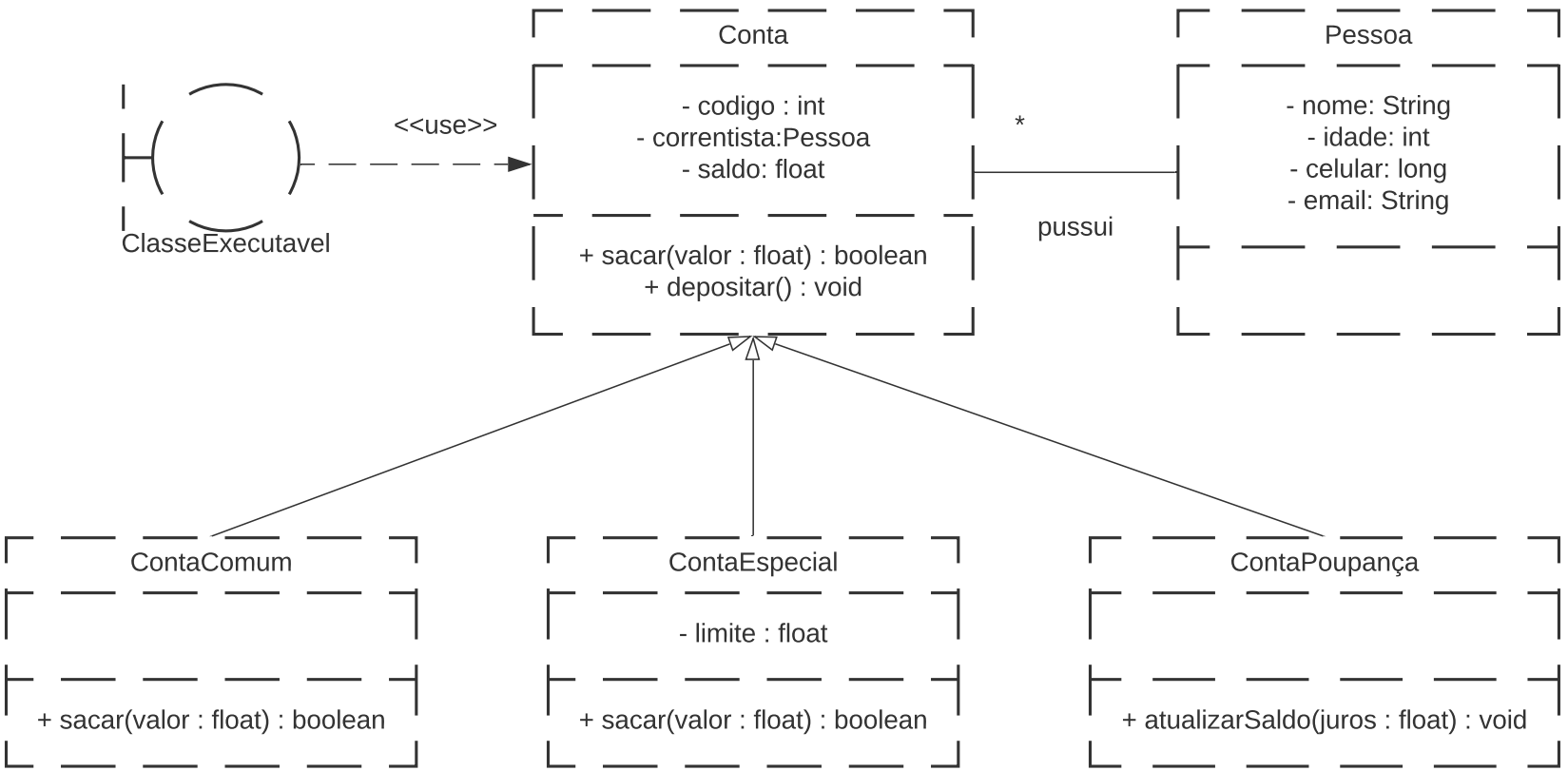
Introdução

Contextualizando...

- Podemos limitar a criação de classes especializadas. Em uma estrutura hierárquica de herança.
- Exemplo: digamos que as únicas especializações de Conta são: Conta Comum, Conta Poupança e Conta Especial.
- Dessa forma, podemos tornar tais classes finais.
- Isso quer dizer que não podemos criar especializações a partir de Conta Comum, Conta Poupança ou Conta Especial.

Hierarquia de Classes

A hierarquia de classes



Final

Uso de final

- A palavra reservada final é utilizada para definirmos uma classe que não permite especializações.
- Também conhecida como classe folha.
- Ex:

```
package classes.java;

public final class ContaComum extends Conta {

    @Override
    public boolean sacar(float valor) {
        if(this.saldo - valor >= 0) {
            this.saldo = this.saldo - valor;
            return true;
        }
        return false;
    }

}
```

Métodos imutáveis

- Também podemos utilizar final na definição de métodos de uma classe.
- Um método final não pode ser reescrito.
- Ex: podemos tornar o método depositar imutável, visto que terá o mesmo comportamento em todas especializações.

```
public final void depositar (float valor) {
    this.saldo = this.saldo + valor;
}
```

TADS Tecnologia em Análise e Desenvolvimento de Sistemas

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

Definição de constantes

- Final também pode ser utilizada para definirmos atributos constantes na estrutura de uma classe.
- Vejamos um exemplo.
- criaremos dois atributos constantes na classe Conta, são eles: SACAR e DEPOSITAR.

```
package classes.java;

public abstract class Conta {

    protected int codigo;
    protected Pessoa correntista;
    protected float saldo;

    public final int SACAR = 1;
    public final int DEPOSITAR = 0;

    public Conta() {

    }
```

TADS Tecnologia em Análise e Desenvolvimento de Sistemas

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

Uso das constantes

- Faremos uso das constantes definidas na classe conta na elaboração de um novo método.
- O método movimentar, que será utilizado tanto para fazer saques ou depósitos.

```
public final boolean movimentar(float valor, int tipo) {
    if(tipo == SACAR) {
        this.sacar(valor);

    } else if( tipo == DEPOSITAR ){
        this.depositar(valor);
    }
    return true;
}
```

Utilizando o método

- Criaremos uma aplicação para movimentar o saldo de uma conta especial.
- Para isso, utilizaremos o método movimentar.

```
package classe.executavel;
import classes.java.Conta;

public class ClasseExecutavel {

    public static void main(String[] args) {

        Pessoa correntista = new Pessoa("Ciclano",33,999887766,"ciclano@gmail.com");

        ContaEspecial especial = new ContaEspecial(3322, correntista, 1500, 500);

        System.out.println(especial.movimentar(500, especial.DEPOSITAR));

    }
}
```

TADS Tecnologia em Análise e Desenvolvimento de Sistemas

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

Atributos e métodos de classe

Atributo de Classe

- Em algumas situações o valor de um determinado atributo deve ser compartilhado por todos objetos criados.
- Exemplo: um controle do número de contas criadas até o momento.
- A alternativa é incluir um atributo na estrutura da classe conta para realizar este controle.

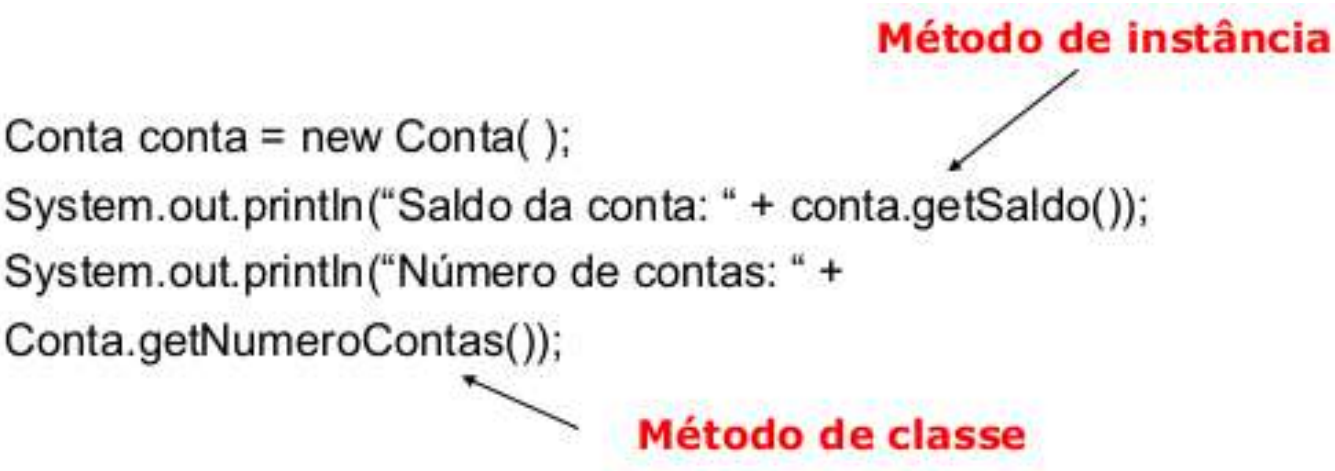
Atributo de Classe

- Para definição de um atributo de classe em Java utilizamos a palavra reservada static.
- Ex:

```
protected int codigo;
protected Pessoa correntista;
protected float saldo;
protected static int numeroContas;
```

Acesso a atributos da classe

- Os atributos de classe são acessados por operações chamadas métodos de classe.
- Um método da classe é referenciado diretamente a partir do identificador da própria classe.
- Já os métodos de instância são acessíveis a partir do objeto.
- Ex:



Declarando métodos de classe

```
public static int getNumeroContas() {
    return numeroContas;
}
```

OBRIGADO!