ECM251 - Linguagens I

Laboratório - Exercícios com POO

Prof. Murilo Zanini de Carvalho
Prof. Tiago Sanches da Silva

Antes de começar!

Clone seu repositório do Github

- Lembre-se sempre antes de iniciar uma aula, clonar seu repositório remoto e realizar as atividades nele.
- Para cada atividade desenvolvida, criar um novo diretório.

Retirado de (https://miro.medium.com/max/4000/0*MZMl76wKo2FQLqG0.png), em 07/03/2021

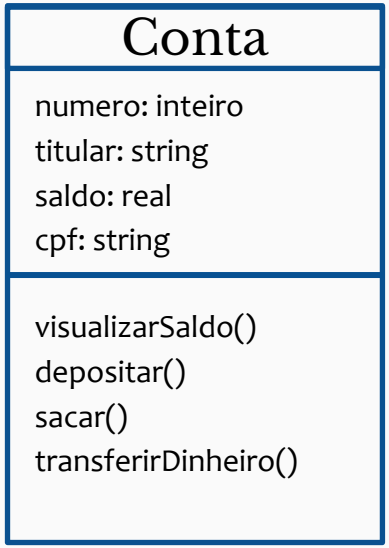
Desenvolver!!

O modelo para implementar

Para iniciar nossa implementação, vamos modelar uma classe que represente uma conta bancária.

Retirado de (https://s2.glbimg.com/2ZioxWDcGUQfPSKbPBBbkRgUyG4=/0x0:825x619/600x0/smart/filters:gifv():strip_icc()/i.s3.glbimg.com/v1/AUTH_08fbf48bc0524877943fe86e43087e7a/internal_photos/bs/2020/u/n/83nNsCQ8SWRrziGD1mAw/stonks-meme.png), em 07/03/2021

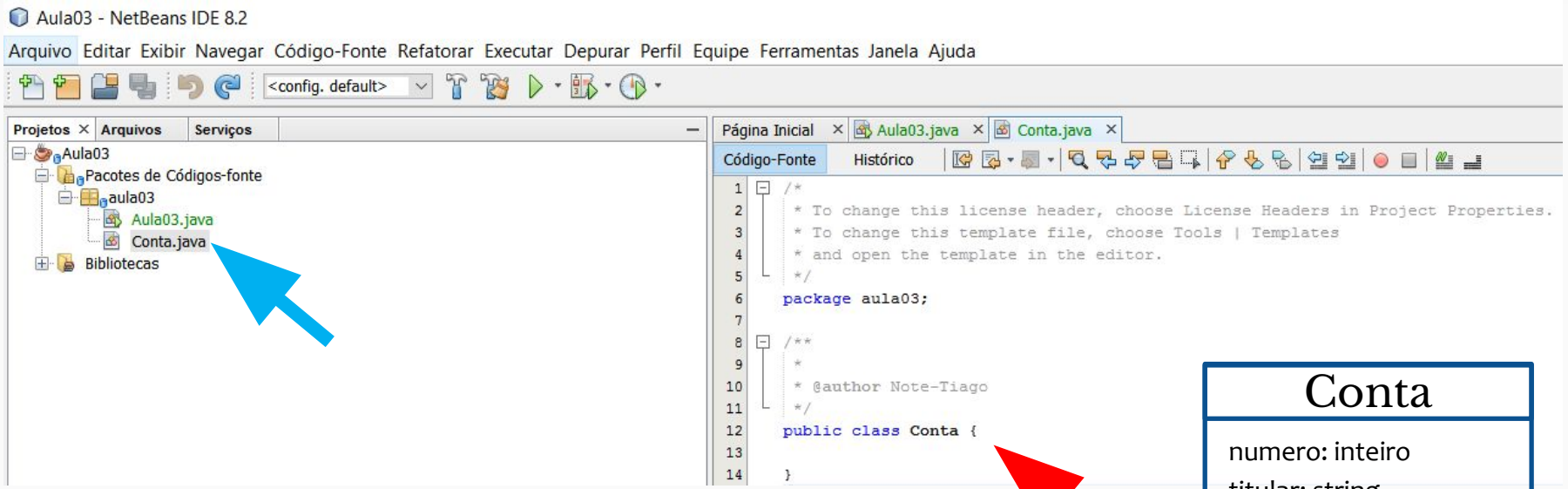
Modelo da Classe Conta



Atributos dessa classe

Métodos dessa classe

Modelo da Classe Conta



ATENÇÃO: Em Java, o nome do arquivo e da classe devem ser os mesmos!

Conta
numero: inteiro titular: string saldo: real cpf: string
visualizarSaldo() depositar() sacar() transferirDinheiro()

Modelo da Classe Conta

Página Inicial | Aula03.java | Conta.java

Código-Fonte | Histórico

```
1  package aula03;
2
3  public class Conta {
4      int numero;
5      String titular;
6      float saldo;
7      String cpf;
8
9      void visualizarSaldo(){
10
11     }
12
13     void depositar(){
14
15     }
16
17      void sacar(){
18
19     }
20
21     void transferirDinheiro(){
22
23     }
24
25
26 }
27
```

Atributos dessa classe

Métodos dessa classe

Conta
numero: inteiro titular: string saldo: real cpf: string
visualizarSaldo() depositar() sacar() transferirDinheiro()

Criação de um objeto

Para criar (construir, instanciar) uma Conta, basta usar a palavra chave new. Devemos utilizar também os parênteses, veremos mais pra frente o porque.

```
package aula03;

public class Aula03 {

    public static void main(String[] args) {

        new Conta();
```

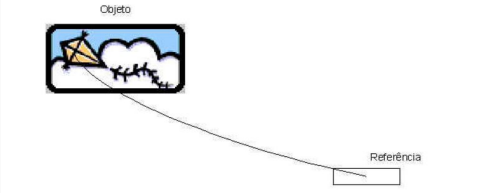
Nome da classe com “()”! Veremos o que é posteriormente!

Comando para criar o objeto na memória

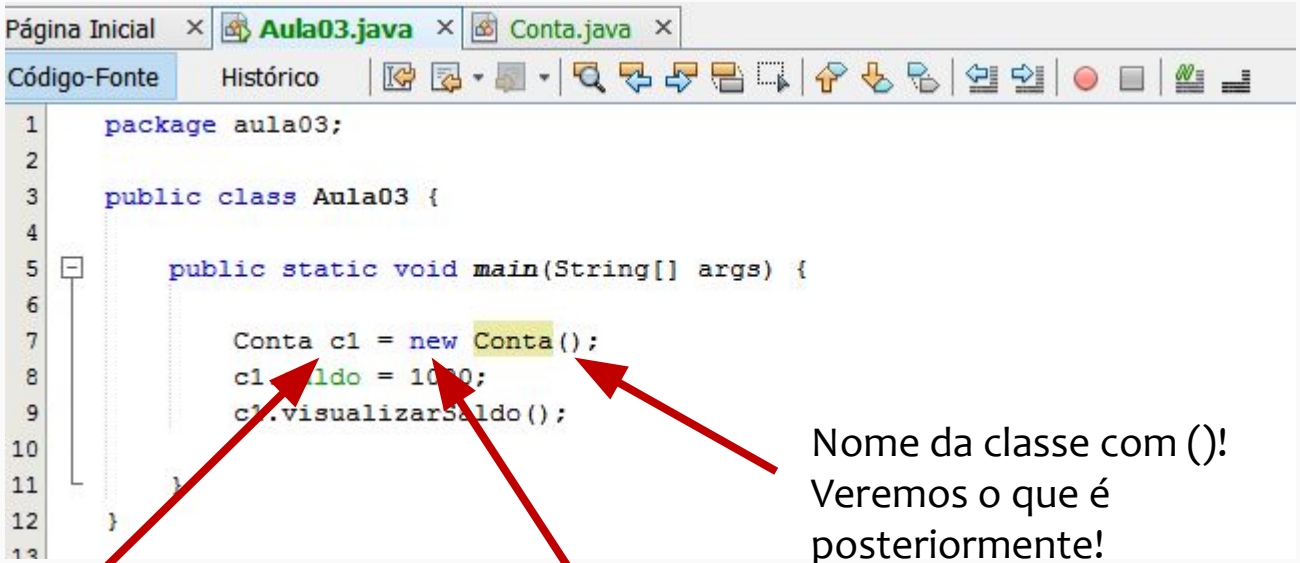
Criação de um objeto

Bem, o código acima cria um objeto do tipo Conta, mas como acessar esse objeto que foi criado? Precisamos ter alguma forma de nos referenciarmos a esse objeto. Precisamos de uma variável:

```
package aula03;

public class Aula03 {

    public static void main(String[] args) {

        Conta c1;
        c1 = new Conta();
```



Criação de um objeto



Nome da classe com ()! Veremos o que é posteriormente!

Comando para criar o objeto na memória

Declarou o objeto como sendo da classe Conta

Conta
numero: inteiro titular: string saldo: real cpf: string
visualizarSaldo() depositar() sacar() transferirDinheiro()

Criação de um objeto

```java
package aula03;

public class Conta {
    int numero;
    String titular;
    float saldo;
    String cpf;

    void visualizarSaldo(){
        System.out.println("Saldo= " + this.saldo);
    }

    void depositar(){

    }

    void sacar(){

    }

    void transferirDinheiro(){

    }
}
```

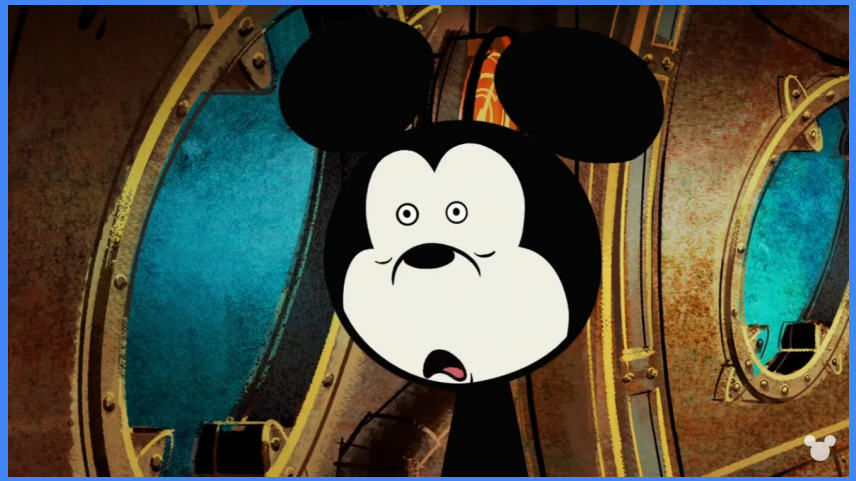
Auto-referencia

this = próprio objeto que esta utilizando o método

Conta
numero: inteiro titular: string saldo: real cpf: string
visualizarSaldo() depositar() sacar() transferirDinheiro()

E da para adicionar outro objeto da mesma classe, na nossa aplicação?

- ***Sim*** ou ***com certeza***?



Retirado de (https://i.imgflip.com/2iab5d.jpg?a448272) , em 07/03/2021

Diversos objetos da mesma classe

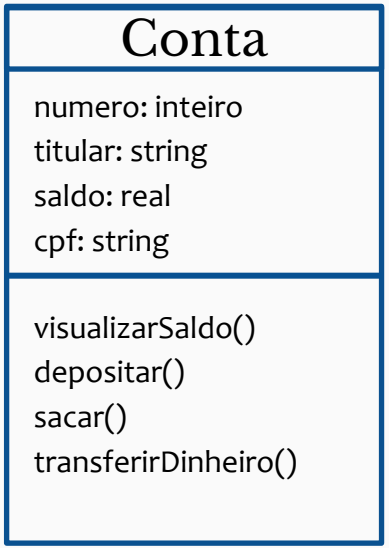
```java
package aula03;

public class Aula03 {

    public static void main(String[] args) {

        Conta c1 = new Conta();
        Conta minhaConta = new Conta();


        c1.saldo = 1000;
        c1.visualizarSaldo();

        minhaConta.saldo = 1800;
        minhaConta.visualizarSaldo();

    }
}
```

```java
public class Conta {
    int numero;
    String titular;
    double saldo;
    String cpf;

    void visualizarSaldo(){
        System.out.println("Saldo= " + this.saldo);
    }
```

this = próprio objeto que está utilizando o método

Nesse caso aqui, quem vai ser o "this" referenciado lá na classe?

Vamos fazer juntos os métodos: sacar e depositar



Desenvolvendo os métodos de uma classe

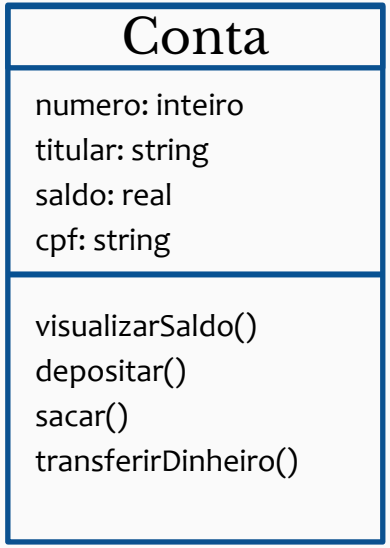
Um método pode retornar um valor para o código que o chamou. No caso do nosso método **sacar** , podemos devolver um valor booleano indicando se a operação foi bem sucedida.

```
boolean sacar(double valor) {
    if (this.saldo < valor) {
        return false;
    }
    else {
        this.saldo = this.saldo - valor;
        return true;
    }
}
```

Façam sozinhos o método: transferirPara

Podemos ir conversando a respeito!

☺️



Atributos - Valores default

As variáveis do tipo atributo, diferentemente das variáveis temporárias (declaradas dentro de um método), recebem um valor padrão. No caso numérico, assumem o 0(zero), no caso de boolean, assumem o valor false.

Você também pode dar valores **default**, como segue:

```
class Conta {
    int numero = 1234;
    String titular = "Niguem";
    double saldo = 0;
}
```

Imagine que comecemos a aumentar nossa classe **Conta** e adicionar nome, sobrenome e cpf do titular da conta.

Começaríamos a ter muitos atributos... e, se você pensar direito, uma **Conta** não tem nome, nem sobrenome nem cpf, quem tem esses atributos é um **Cliente**. Sugestão?

Podemos criar uma nova classe e fazer uma composição!

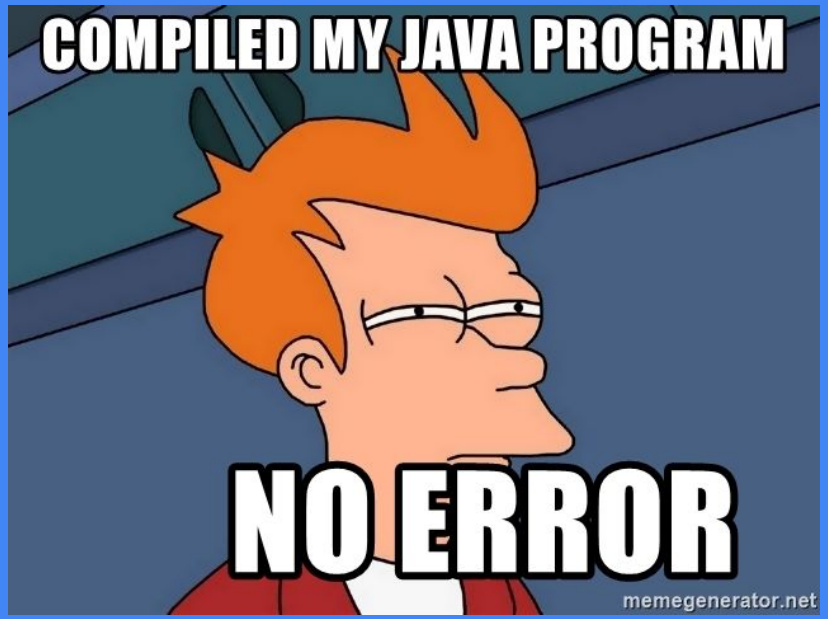
```
class Cliente {
    String nome;
    String sobrenome;
    String cpf;
}
```

```
class Cliente {
    String nome;
    String sobrenome;
    String cpf;
}

class Conta {
    int numero;
    double saldo;
    Cliente titular;
    // ..
}
```

NullPointerException

Como utilizar isso? Vamos analisar juntos!

Corrigir BUG!!



Retirado de (https://memegenerator.net/img/instances/65665341.jpg), em 07/03/2021

Exibir no programa principal as informações formatadas sobre a conta.

É possível criar um método que retorne todas as informações sobre a conta de uma maneira formatada?

Apenas Referências

Construa duas contas com o new e compare-os com o ==. E se eles tiverem os mesmos atributos?

1

```
Conta c1 = new Conta();
c1.titular = "Danilo";
c1.saldo = 100;

Conta c2 = new Conta();
c2.titular = "Danilo";
c2.saldo = 100;

if (c1 == c2) {
  System.out.println("iguais");
} else {
  System.out.println("diferentes");
}
```

Apenas Referências

Construa duas contas com o new e compare-os com o ==. E se eles tiverem os mesmos atributos?

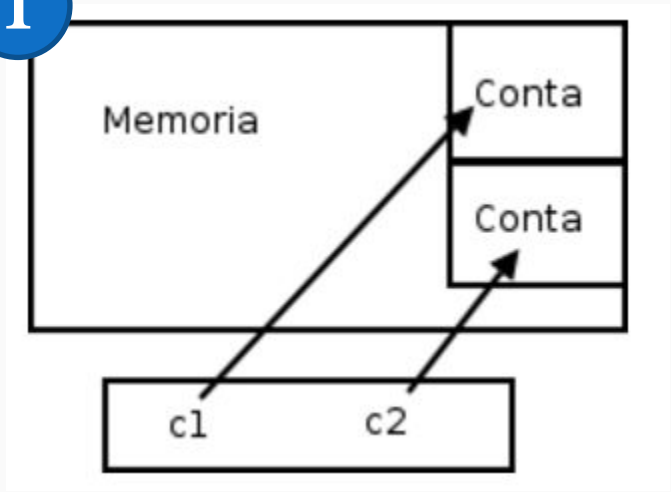
2

```
Conta c1 = new Conta();
c1.titular = "Hugo";
c1.saldo = 100;

Conta c2 = c1;

if (c1 == c2) {
  System.out.println("iguais");
} else {
  System.out.println("diferentes");
}
```

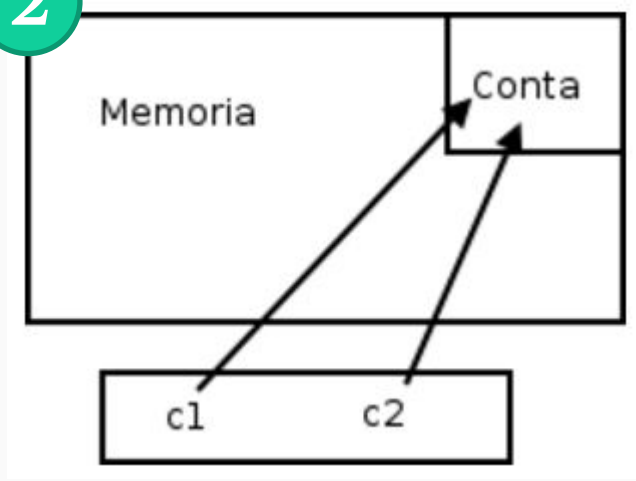
Apenas Referências

Crie duas referências para a mesma conta, compare-os com o ==. Tire suas conclusões. Para criar duas referências pra mesma conta:

1



2



Perguntas?