

# ROTEIRO

---

- Sobrecarga
  - De métodos
  - De construtores
- Package
- Import
- Modificadores

# Sobrecarga de métodos

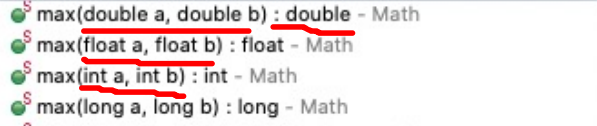
- Métodos com o mesmo nome podem ser declarados na mesma classe, contanto que tenham diferentes conjuntos de parâmetros, determinados pelo número, tipo e ordem dos parâmetros.

```
1
2 public class ExemploSobrecarga {
3     public int soma(int x, int y) {
4         return x + y;
5     }
6
7     public int soma(int x, int y, int z) {
8         return x + y;
9     }
10    public double soma (double x, double y) {
11        return x + y;
12    }
13    /* public long soma (int x, int y) {
14        return x + y;
15    }
16    */
17    public static void main(String[] args) {
18        ExemploSobrecarga es = new ExemploSobrecarga();
19        System.out.println(es.soma(12, 20));
20    }
21 }
22
```

# Sobrecarga de métodos

- Métodos com o mesmo nome podem ser declarados na mesma classe, contanto que tenham diferentes conjuntos de parâmetros, determinados pelo número, tipo e ordem dos parâmetros.

```
1
2 public class ExemploSobrecarga {
3     public int soma(int x, int y) {
4         return x + y;
5     }
6
7     public int soma(int x, int y, int z) {
8         return x + y;
9     }
10    public double soma (double x, double y) {
11        return x + y;
12    }
13    /* public long soma (int x, int y) {
14        return x + y;
15    }
16 */
17    public static void main(String[] args) {
18        ExemploSobrecarga es = new ExemploSobrecarga();
19        System.out.println(es.soma(12, 20));
20        Math.
21    }
22 }
23
```



# SOBRECARGA DE CONSTRUTORES

- Ao criar uma instância de objeto, o programa pode querer fornecer vários construtores com base nos dados para o objeto que está sendo criado.

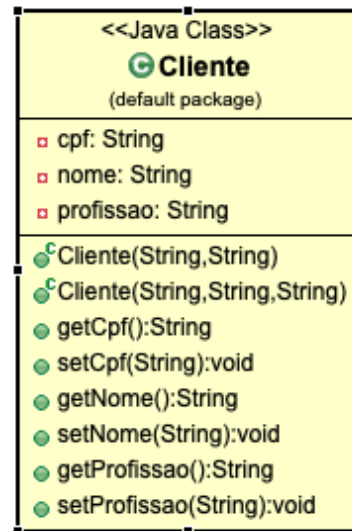
```
1
2 public class Cliente {
3     private String cpf;
4     private String nome;
5     private String profissao;
6
7     public Cliente() {
8     }
9     public Cliente (String cpf) {
10         setCpf(cpf);
11     }
12     public Cliente (String nome, String cpf) {
13         setNome(nome);
14         setCpf(nome);
15     }
16     public Cliente (String nome, String cpf, String profissao) {
17         setNome(nome);
18         setCpf(nome);
19         setProfissao(profissao);
20     }
21     public String getCpf() {
22         return cpf;
23     }
24     public void setCpf(String cpf) {
25         this.cpf = cpf;
26     }
27     public String getNome() {
28         return nome;
29     }
30     public void setNome(String nome) {
31         this.nome = nome;
```

# SOBRECARGA DE CONSTRUTORES

---

```
2 public class TesteBanco {  
3  
4     public static void main(String[] args) {  
5         Cliente c1 = new Cliente("12312312312");  
6         Cliente c2 = new Cliente();  
7         System.out.println(c1.getCpf());  
8         System.out.println(c2.getCpf());  
9     }  
10  
11 }
```

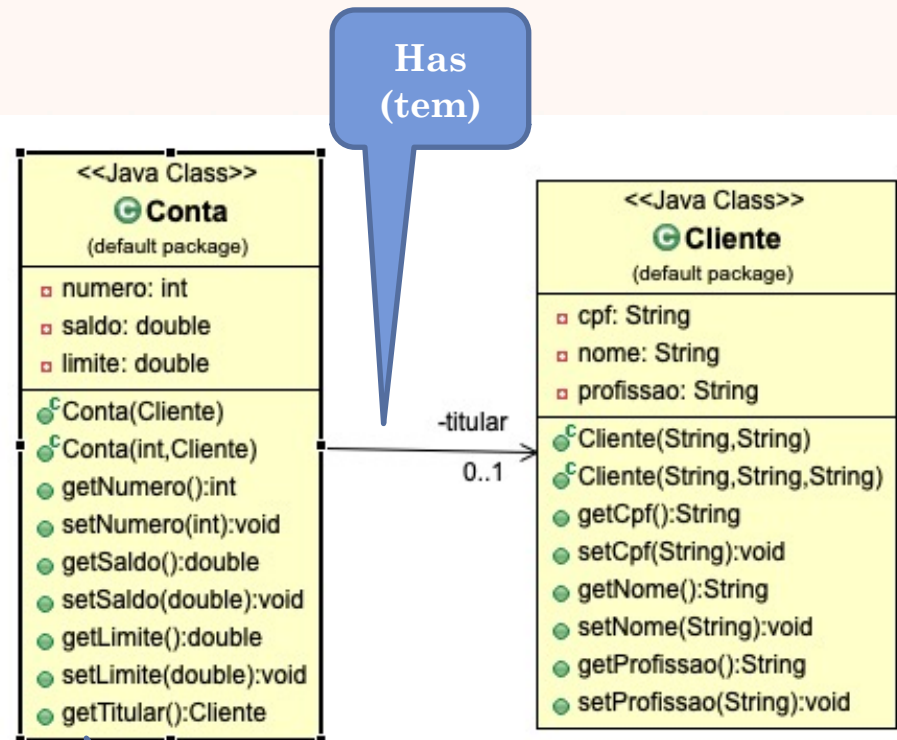
# SOBRECARGA DE CONSTRUTORES



```
1
2 public class TesteBanco {
3
4     public static void main(String[] args) {
5         Cliente c1 = new Cliente("12312312312");
6         Cliente c2 = new Cliente();
7
8         Cliente c3 = new Cliente("12312312312", "Arthur");
9         Cliente c4 = new Cliente("12312312313", "Manuela", "Professora");
10
11         System.out.println(c3.getCpf());
12         System.out.println(c4.getCpf());
13     }
14
15 }
```

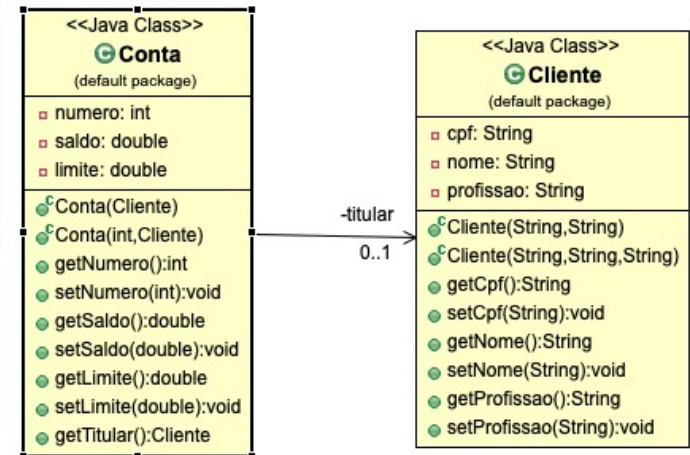
# SOBRECARGA DE CONSTRUTORES

```
2 public class Conta {
3     private int numero;
4     private Cliente titular;
5     private double saldo;
6     private double limite;
7
8     public Conta (Cliente titular) {
9         this.titular = titular;
10    }
11    public Conta (int numero, Cliente titular) {
12        setNumero(numero);
13        this.titular = titular;
14    }
15    public int getNumero() {
16        return numero;
17    }
18    public void setNumero(int numero) {
19        this.numero = numero;
20    }
21    public double getSaldo() {
22        return saldo;
23    }
24    public void setSaldo(double saldo) {
25        this.saldo = saldo;
26    }
27    public double getLimite() {
28        return limite;
29    }
30    public void setLimite(double limite) {
31        this.limite = limite;
32    }
33    public Cliente getTitular() {
34        return titular;
35    }
36 }
```



# SOBRECARGA DE CONSTRUTORES

```
1
2 public class TesteBanco {
3
4     public static void main(String[] args) {
5         Cliente c3 = new Cliente("12312312312", "Arthur");
6         Conta c = new Conta(444, c3);
7
8         System.out.println(c.getNumero());
9         System.out.println(c.getSaldo());
10        System.out.println(c.getTitular());
11        System.out.println(c.getTitular().getCpf());
12        System.out.println(c.getTitular().getNome());
13    }
14 }
15
```



Problems Javadoc Declaration Console

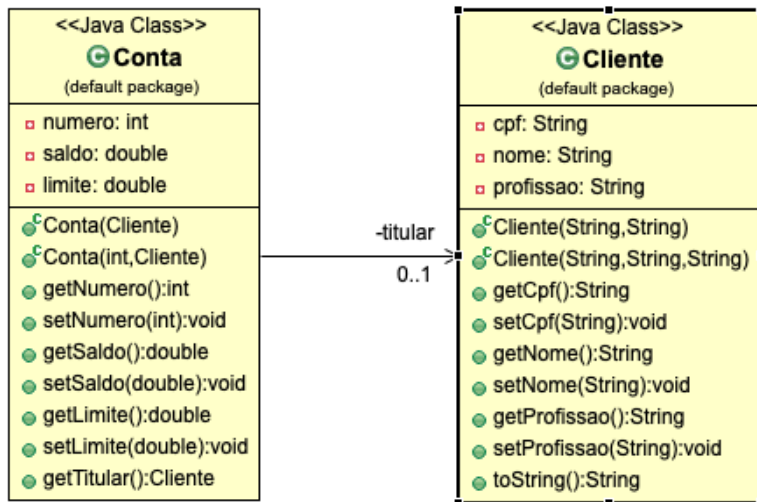
<terminated> TesteBanco [Java Application] /Users/gustavo/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.mac

```
444
0.0
Cliente@1eb44e46
12312312312
12312312312
|
```



# SOBRECARGA DE CONSTRUTORES

```
39 public String toString () {  
40     return "[nome: " + getNome() + "]---[cpf: " + getCpf() + "];"  
41 }
```



Dentro da  
classe Cliente

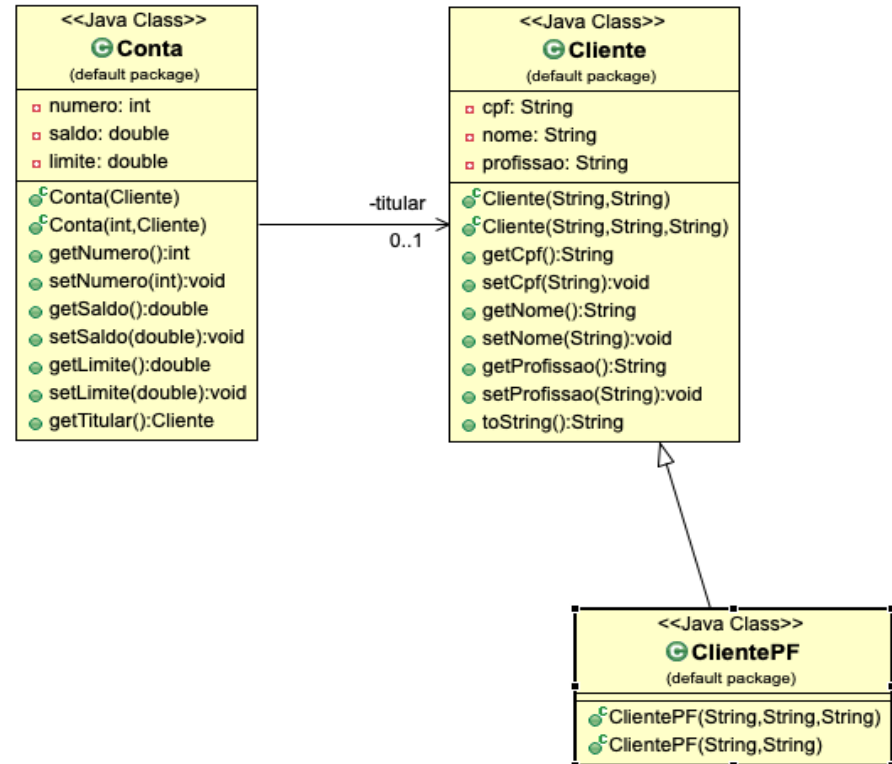
```
1  
2 public class TesteBanco {  
3  
4 public static void main(String[] args) {  
5     Cliente c3 = new Cliente("12312312312", "Arthur");  
6     Conta c = new Conta (444, c3);  
7  
8     System.out.println(c.getNumero());  
9     System.out.println(c.getSaldo());  
10    System.out.println(c.getTitular());  
11    System.out.println(c.getTitular().getCpf());  
12    System.out.println(c.getTitular().getNome());  
13 }  
14 }
```

```
444  
0.0  
[nome: 12312312312]---[cpf: 12312312312]  
12312312312  
12312312312
```

# SOBRECARGA DE CONSTRUTORES (HERANÇA)

```
1
2 public class Cliente {
3     private String cpf;
4     private String nome;
5     private String profissao;
6
7     public Cliente (String nome, String cpf) {
8         setNome(nome);
9         setCpf(nome);
10    }
11    public Cliente (String nome, String cpf, String profissao) {
12        setNome(nome);
13        setCpf(nome);
14        setProfissao(profissao);
15    }
16    public String getCpf() {
17        return cpf;
18    }
19    public void setCpf(String cpf) {
20        this.cpf = cpf;
21    }
}
```

```
2 public class ClientePF extends Cliente{
3     /* public ClientePF () {
4     }
5     */
6     public ClientePF (String nome, String cpf, String profissao) {
7         //super();
8         super(nome, cpf, profissao);
9     }
10    public ClientePF (String nome, String cpf) {
11        //super();
12        this(nome, cpf, null);
13    }
14 }
```



# SOBRECARGA DE CONSTRUTORES (HERANÇA)

---

- Podemos notar que o segundo construtor chama o primeiro construtor.
- OBS: A palavra `this` (na chamada a outro construtor) dentro de um construtor precisa estar na primeira linha de código do construtor. Pode haver mais código de inicialização após a chamada `this`, mas **nunca** antes.
- **Construtores não são herdados.** Assim como os métodos, os construtores podem chamar os construtores não privados de sua superclasse imediata. Para isso, basta usar a palavra reservada `super` a partir da primeira linha do construtor filho. Quando não há uma chamada para `super` com argumentos, o construtor da superclasse com zero argumentos é chamado implicitamente. Nesse caso, se não houver nenhum construtor na superclasse com zero argumentos, ocorrerá um erro de compilação.

# Chamada a métodos com super

```
2 public class Cliente {
3     private String cpf;
4     private String nome;
5     private String profissao;
6
7     public Cliente (String nome, String cpf) {...}
11    public Cliente (String nome, String cpf, String profissao) {...}
16    public String getCpf() {...}
19    public void setCpf(String cpf) {...}
22    public String getNome() {...}
25    public void setNome(String nome) {...}
28    public String getProfissao() {...}
31    public void setProfissao(String profissao) {...}
34    public String toString () {
35        return "[nome: " + getNome() + "]---[cpf: " + getCpf() + "];
36    }
37 }
```

```
1
2 public class ClientePF extends Cliente{
3     /* public ClientePF () {
4         }
5     */
6     public ClientePF (String nome, String cpf, String profissao) {
7         //super();
8         super(nome, cpf, profissao);
9     }
10    public ClientePF (String nome, String cpf) {
11        //super();
12        this(nome, cpf, null);
13    }
14    public String toString() {
15        return "[Pessoa Fisica] " + super.toString();
16    }
17 }
```

# Chamada a métodos com super

```
1
2 public class TesteBanco {
3
4     public static void main(String[] args) {
5         Cliente c3 = new ClientePF("12312312312", "Arthur");
6         Conta c = new Conta (444, c3);
7
8         System.out.println(c.getNumero());
9         System.out.println(c.getSaldo());
10        System.out.println(c.getTitular());
11        System.out.println(c.getTitular().getCpf());
12        System.out.println(c.getTitular().getNome());
13    }
14 }
```

Problems @ Javadoc Declaration Console ✕

<terminated> TesteBanco [Java Application] /Users/gustavo/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86

444

0.0

[Pessoa Fisica] [nome: 12312312312]---[cpf: 12312312312]

12312312312

12312312312