

FIA/P GRADUAÇÃO

DOMAIN DRIVEN DESIGN

Prof. Me. Thiago T. I. Yamamoto

#04 - CLASSES, ATRIBUTOS E MÉTODOS

TRAJETÓRIA



- ✓ Orientação a Objetos
- ✓ Introdução ao Java
- ✓ IDE e Tipos de Dados
- ✓ Classes, atributos e métodos

#04 - AGENDA

- Objetos
- Classes
- Atributos
- Métodos





OBJETOS

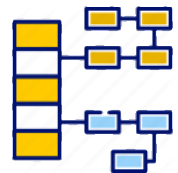
- Informalmente, um objeto representa uma entidade, seja física, conceitual ou de software:



Entidade física: Caminhão

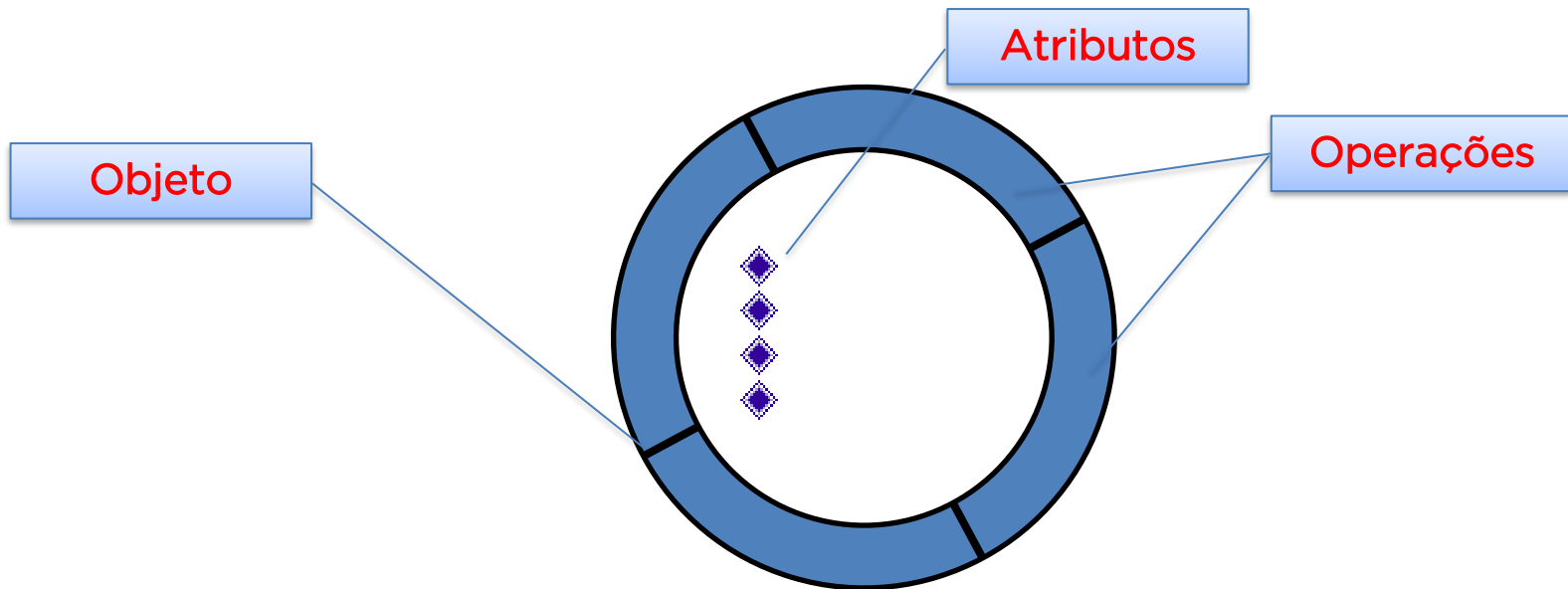


Entidade conceitual: Processo químico



Entidade de software: Lista

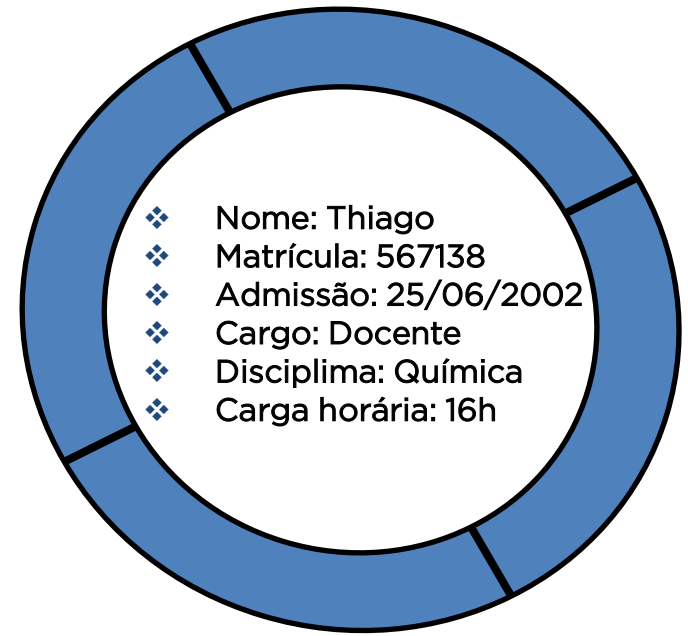
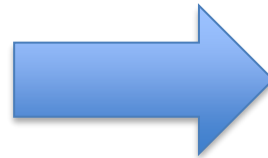
- Um **objeto** é uma **entidade** com **fronteira** e **identidade** bem definidas que encapsulam o **estado** e o **comportamento**:
 - O **estado** é representado pelos **atributos** e relacionamentos;
 - O **comportamento** é representado pelas **operações** e métodos;



- **Estado** é a **condição** ou **situação** durante a vida de um **objeto**, que satisfaz alguma condição, realiza alguma atividade ou aguarda algum evento;
- O **estado** de um objeto normalmente é **alterado** ao longo do tempo;



Nome: Thiago
Matrícula: 567138
Admissão: 25/06/2002
Cargo: Docente
Disciplina: Química
Carga horária: 16h

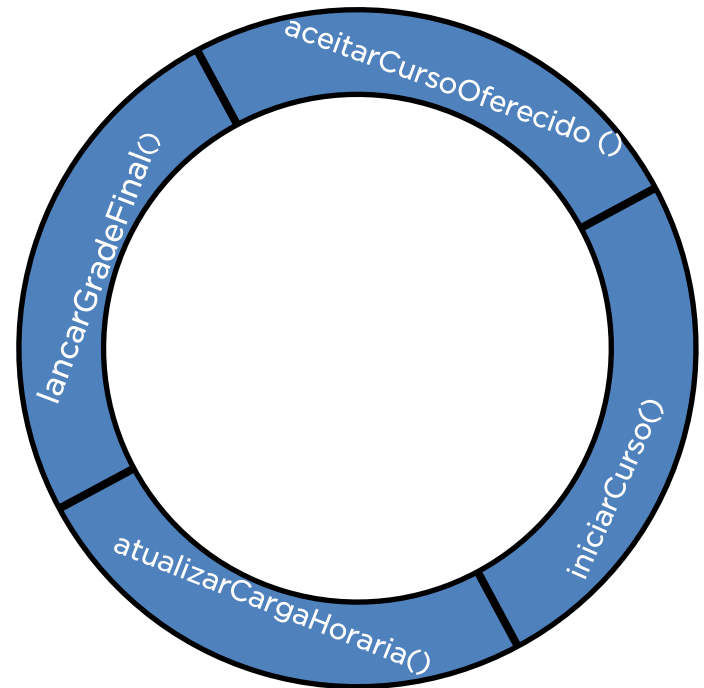
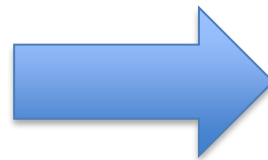


Professor Thiago

- O comportamento determina como um objeto age ou reage a uma requisição de outro objeto;
- O comportamento de um objeto é representado pelas operações que ele pode realizar;



Nome: Thiago
Matrícula: 567138
Admissão: 25/06/2002
Cargo: Docente
Disciplina: Química
Carga horária: 16h



Professor Thiago

- Cada objeto tem uma identidade única, mesmo se o estado é idêntico ao de outro objeto;



Professor “Roberto” ensina “Química”



Professor “Roberto” ensina “Química”



CLASSES

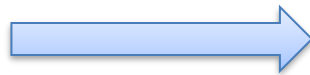
- **Abstração** é a habilidade de concentrar nos **aspectos essenciais** de um contexto qualquer, **ignorando características menos importantes**. Em modelagem **orientada a objetos**, uma **classe** é uma **abstração** de entidades existentes no domínio de um sistema de software;
- **Classes** são estruturas das **linguagens de programação O.O.** para descrever, em determinado modelo, os **dados** que devem ser representados e as **operações** que devem ser efetuadas com estes dados. Cada classe deve ter um **nome que seja facilmente associável** ao modelo que a classe representa;
 - **Classes** são somente **moldes** ou **formas** que representam os **modelos abstratamente**;
- Em outros termos, uma **classe** descreve os **serviços** providos por seus **objetos** e quais **informações** eles podem **armazenar**;

- [illegible]

- Definir uma **classe** significa **formalizar um tipo de dado** e as **operações associadas** a esse tipo. Uma **classe** estabelece o **comportamento** de seus objetos de métodos e os **estados** possíveis destes objetos através de atributos;
- A **diferença fundamental** entre classes e objetos está no fato de que um **objeto constitui uma entidade concreta** com tempo e espaço de existência, enquanto a **classe é somente uma abstração**, tipo de dado definido pelo programador;



Classe



Objetos ou instâncias da classe

- Na programação orientada a objetos a classe é a unidade básica de programação. Todos os programas são escritos como um conjunto de classes, e todos os códigos que você escrever devem fazer parte de uma classe;
- Em **Java**, as definições de **classe** são armazenadas em arquivos separados com a extensão **.java** e o nome do arquivo deve ser igual ao nome da classe que tiver sido definida dentro dele;
- Por convenção, uma **classe em Java** é sempre escrita em **UpperCamelCase**;
 - **Exemplos:**
 - Carro
 - FuncionarioPadrao
 - FuncionarioPadraoTerceirizado



- Sintaxe básica para declaração de uma classe em Java:

```
<modificador> class <NomeDaClasse>{  
    [declaracao de atributos]  
    [declaracao dos construtores]  
    [declaracao dos metodos]  
}
```

```
public class Veiculo{
```

```
    double cargaMaxima;
```

```
    public Veiculo(){  
        cargaMaxima = 100d;  
    }
```

```
    void setCargaMaxima(double novoPeso){  
        cargaMaxima = novoPeso;  
    }
```

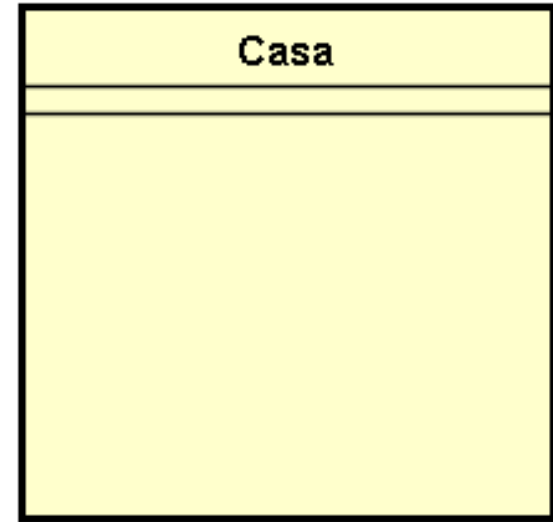
```
}
```

Classe

Atributo

Construtor

Método



- Lembre-se de que o arquivo .java deve possuir o mesmo nome da classe!

```
public class Casa{  
  
}
```



ATRIBUTOS

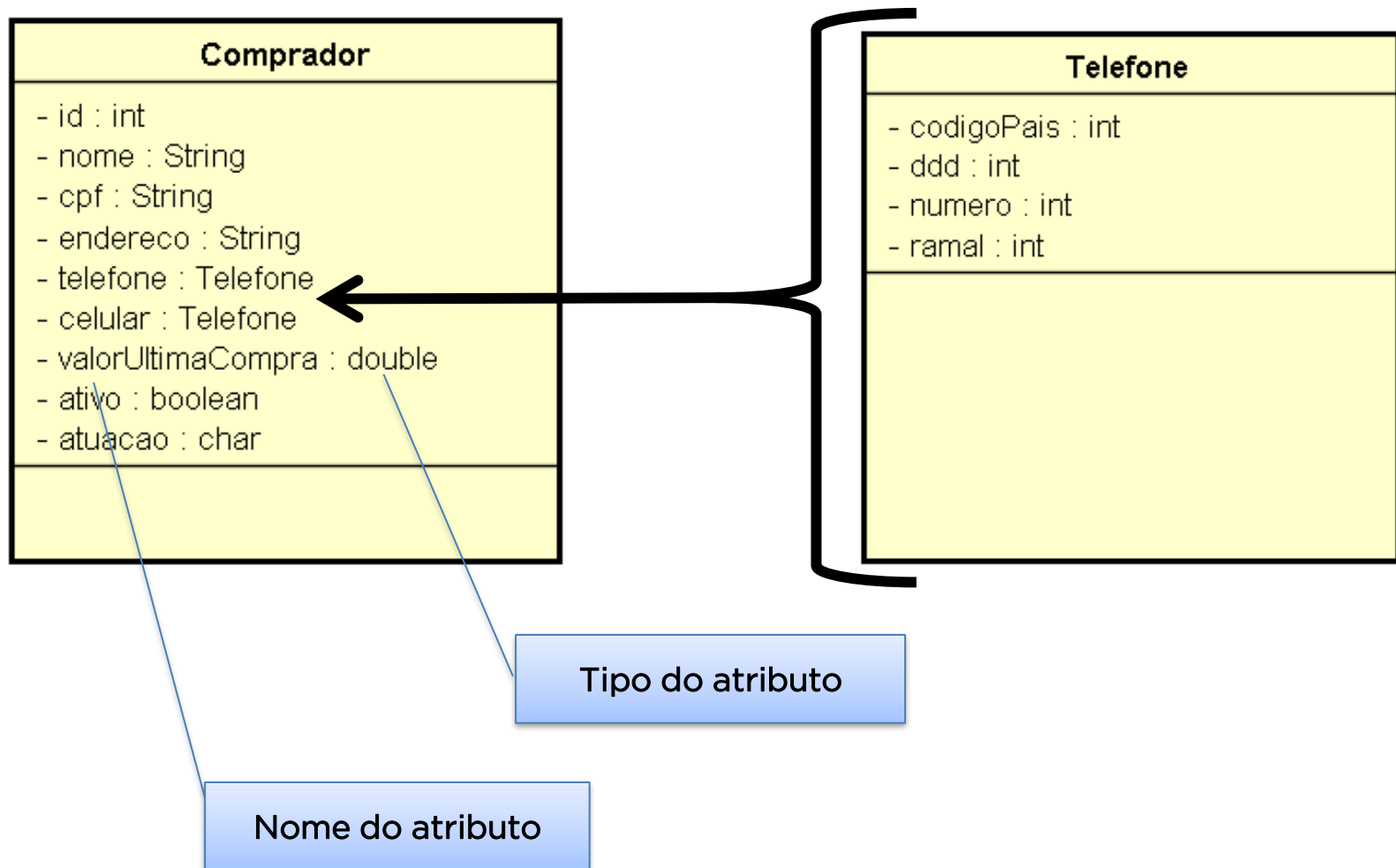
- **Atributos** são **características** específicas de um objeto, por exemplo, para a classe **Carro**, os possíveis atributos seriam:
 - Modelo
 - Ano de Fabricação
 - Fabricante
 - Cor
- Os **atributos** possuem **valores**. O conjunto de valores dos atributos de um determinado objeto é chamado de **estado do objeto**;
- Em **Java** os atributos de uma classe podem ser definidos através de:
 - **Tipos Primitivos**
 - **Variáveis do tipo:** int, float, double, char, boolean, etc.
 - **Tipos por Referência**
 - **Objetos do tipo:** String, Vetores (int, double, boolean, Carro, etc.), Casa, Profissao, etc.

- Os **atributos** devem possuir um tipo;

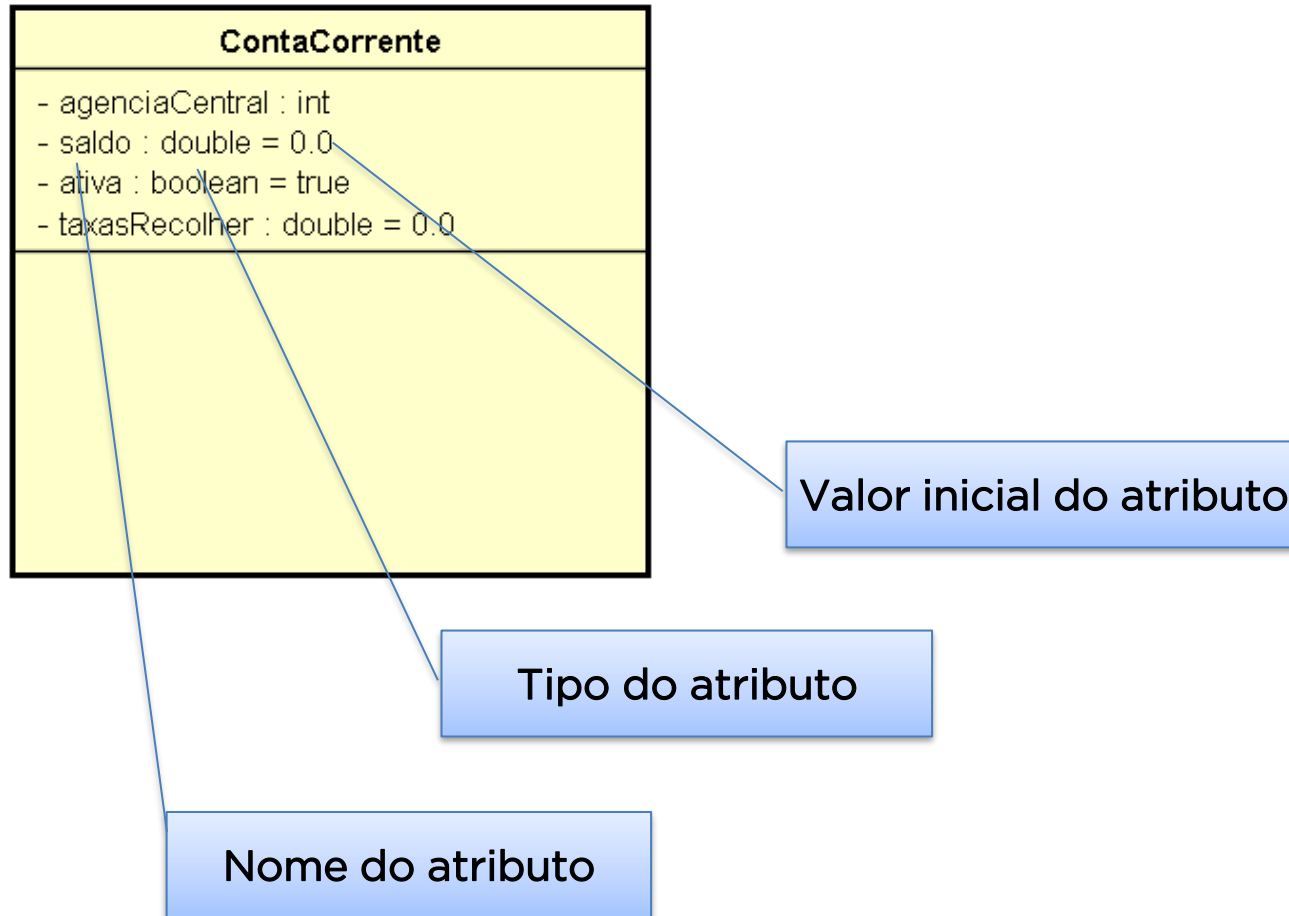
Tipo	Descrição	Valores de exemplos
boolean	Valor verdadeiro ou falso	<i>true</i> ou <i>false</i>
char	Caracter	a, b, c, 1, 2, 3, @, !, #
int	Número inteiro	1, 2, 3, 1000, 5687
double	Número real	45.78, 0.5, 8.9324, 0.000089
String	Texto de qualquer tamanho	João Paulo, 09985-361, 28@



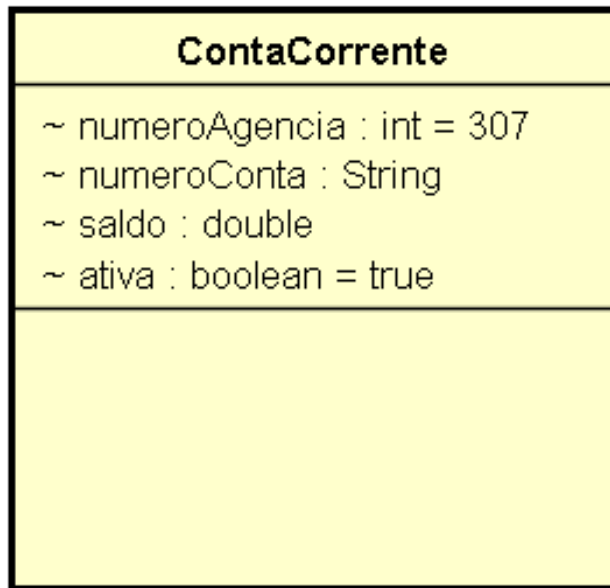
- Como fica na modelagem?



- É possível associar um **valor inicial** para os atributos;

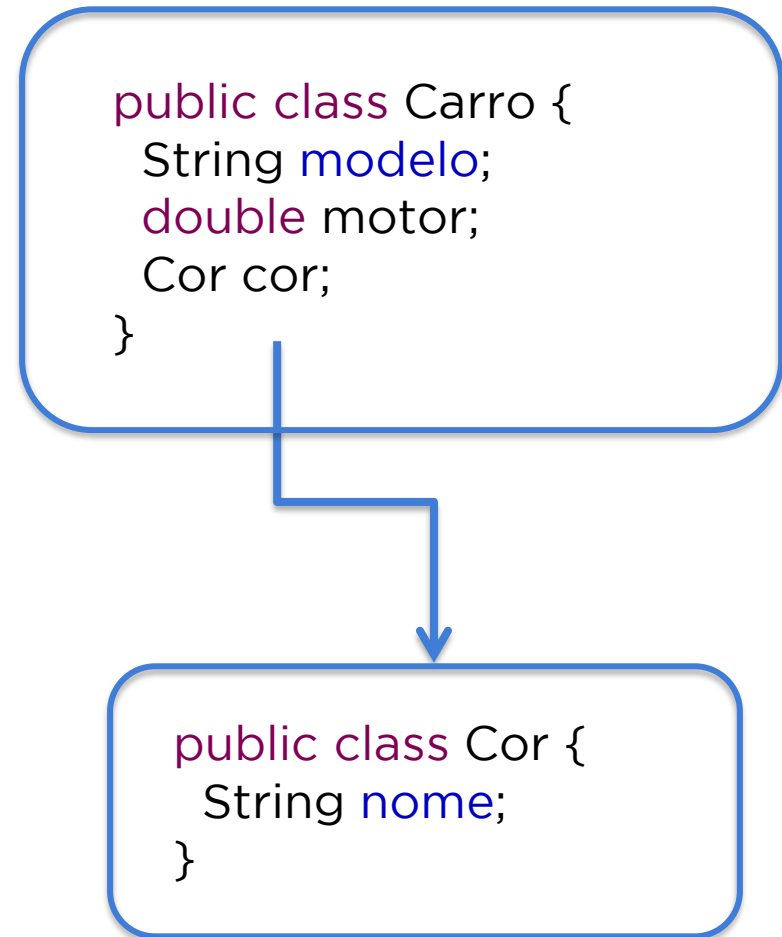
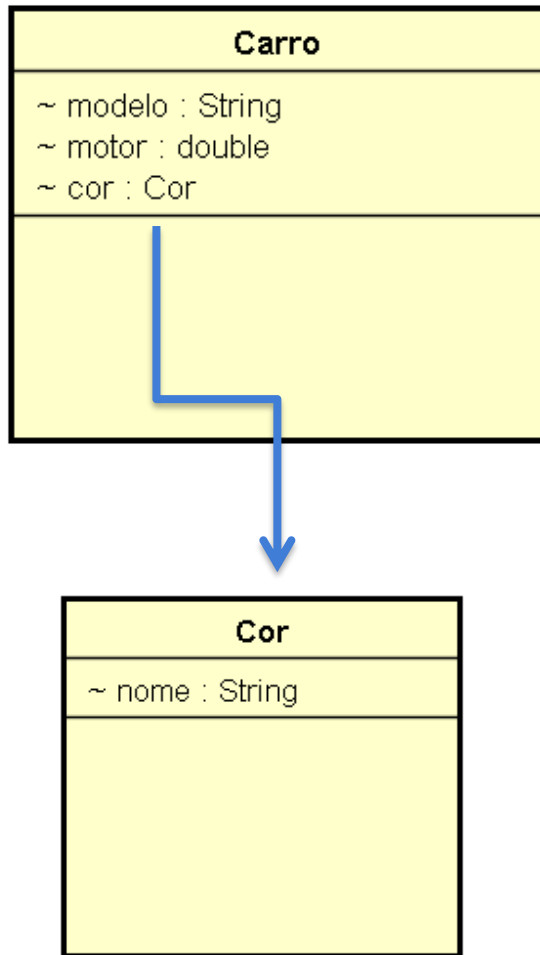


- Os **atributos** podem ser **definidos com ou sem** valores iniciais;
- Caso **não defina** um valor inicial, o **Java** irá colocar um **valor padrão**, dependendo do **tipo** do atributo;



```
public class ContaCorrente {  
  
    int numeroAgencia = 307;  
  
    String numeroConta;  
  
    double saldo;  
  
    boolean ativa = true;  
  
}
```

- Os atributos podem ser referência para outras classes:





MÉTODOS

- Um **método** pode ser entendido como uma **operação** ou **serviço** oferecido por um objeto;
- O **método** é um comportamento específico, **residente no objeto**, que define como ele **deve agir quando exigido**. Portanto, os métodos definem as **habilidades dos objetos**;
- Um método em uma classe é apenas uma **definição**. A ação só ocorre quando o **método é invocado(chamado)** através do objeto;
- Do ponto de vista da **implementação**, em um programa OO os métodos são implementados em funções colocadas no nível do objeto ao qual pertencem;
- Por convenção, o **nome de um método** em Java é sempre escrito em **lowerCamelCase**. Por exemplo: `exibirTotal`, `calcularAreaTriangulo`;

- O **parâmetro** de uma **operação** correspondem as informações que **esta recebe** quando é executada;
- Uma **operação** pode ter zero ou mais **parâmetros** e para cada parâmetro é necessário **definir** também o **seu tipo**;
- Os **parâmetros** são separados por vírgula e seguem a sintaxe:
 - **<nome do parâmetro> : <tipo do parâmetro>**

CadastroCliente	
- quantidadeCliente : int	
+ inserir(novoCliente : Cliente) : Cliente	
+ atualizar(cliente : Cliente) : void	
+ excluir(codigo : int) : boolean	
+ incluirContato(contato : Contato) : void	
+ ajustarCredito(valor : double) : int	
+ enviarEmailPromocao(assunto : String, enviarLink : boolean, cliente : Cliente) : void	

Nome do parâmetro

Tipo do parâmetro

- Sintaxe básica para declaração de um método:

```
<modificador> <tipo de retorno> <nomeDoMetodo>(<[lista de argumentos]>){  
    [instrucoes];  
}
```

```
public class Cachorro{
```

```
    double peso = 0.0d;
```

```
    double getPeso(){  
        return peso;  
    }
```

```
    void setPeso(double novoPeso){  
        peso = novoPeso;  
    }  
}
```

Classe

Atributo

Métodos

- A tag [lista de argumentos] permite **passar valores para o interior** de um **método**;
- Os elementos da lista ficam **separados por vírgula** e cada um pode ter um tipo de dado distinto;

```
void depositar(int agencia, String numeroConta, double valor){  
}
```

Tipo do parâmetro #1

Nome do parâmetro #1

parâmetro #2

parâmetro #3

```
double somar(double valor1, double valor2){  
    return (valor1 + valor2);  
}
```



- A tag <tipo de retorno> indica o tipo de valor que o método retornará;
- Se o método não retornar nenhum valor, deve-se declarar o método como void;
- Para retornar valor, utiliza-se a instrução **return**;

Tipo de Retorno

```
public class Cachorro{  
  
    double peso = 0.0d;  
  
    double getPeso(){  
        return peso;  
    }  
  
    void setPeso(double novoPeso){  
        peso = novoPeso;  
    }  
}
```

- Sintaxe básica para declaração de um método:

```
<modificador> <tipo de retorno> <nomeDoMetodo>(<[lista de argumentos]>){  
    [instrucoes];  
}
```

```
public class Cachorro{
```

```
    double peso = 0.0d;
```

```
    double getPeso(){  
        return peso;  
    }
```

```
    void setPeso(double novoPeso){  
        peso = novoPeso;  
    }  
}
```

Classe

Atributo

Métodos

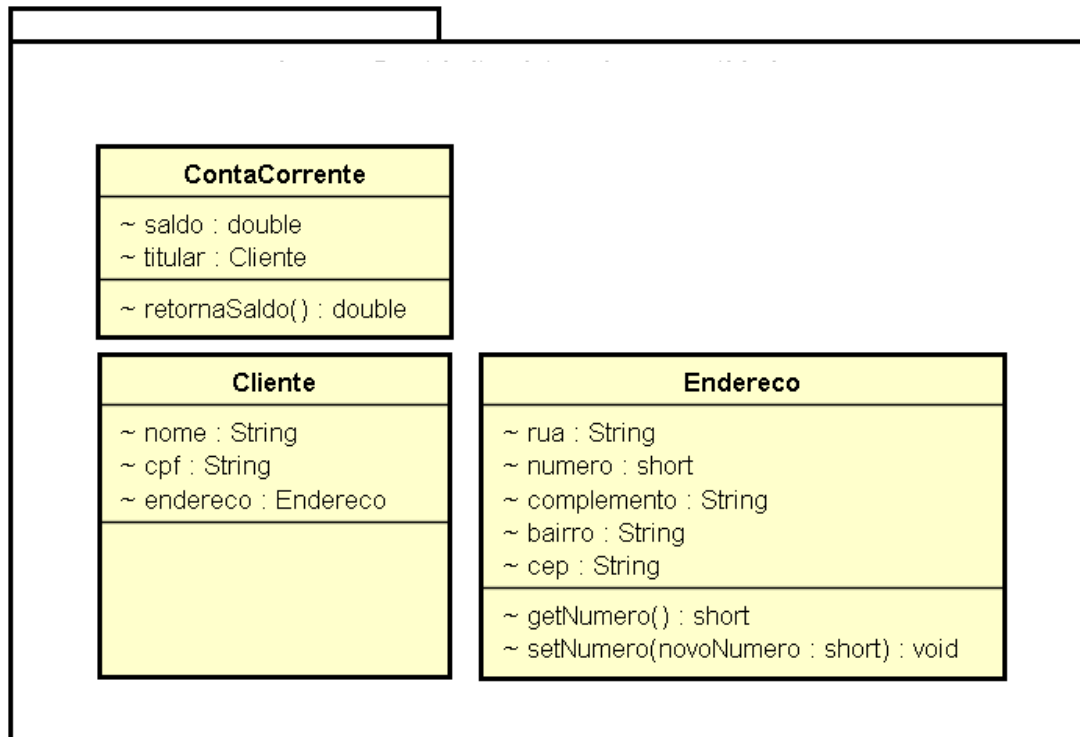
ContaCorrente

- ~ sacar(valor : double) : boolean
- ~ exibirExtrato(quantidadeDias : int) : void
- ~ exibirExtratoAntigo(quantidadeDias : int, mes : int, ano : int) : void

```
public class ContaCorrente {  
    boolean sacar(double valor) {  
        return false;  
    }  
  
    void exibirExtrato(int quantidadeDias) {  
  
    }  
  
    void exibirExtratoAntigo(int quantidadeDias, int mes, int ano) {  
  
    }  
}
```


PRÁTICA

- Crie um projeto chamado **FiapBank** e implemente as seguintes classes:



- Cria uma classe de teste para **instanciar** todas as classes; Os valores devem ser **inseridos pelo usuário**.



Copyright © 2020 - 2022 Prof. Thiago T. I. Yamamoto

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).

*“Se você realmente quer algo, não espere.
Ensine você mesmo a ser impaciente”
Gurbaksh Chahal*