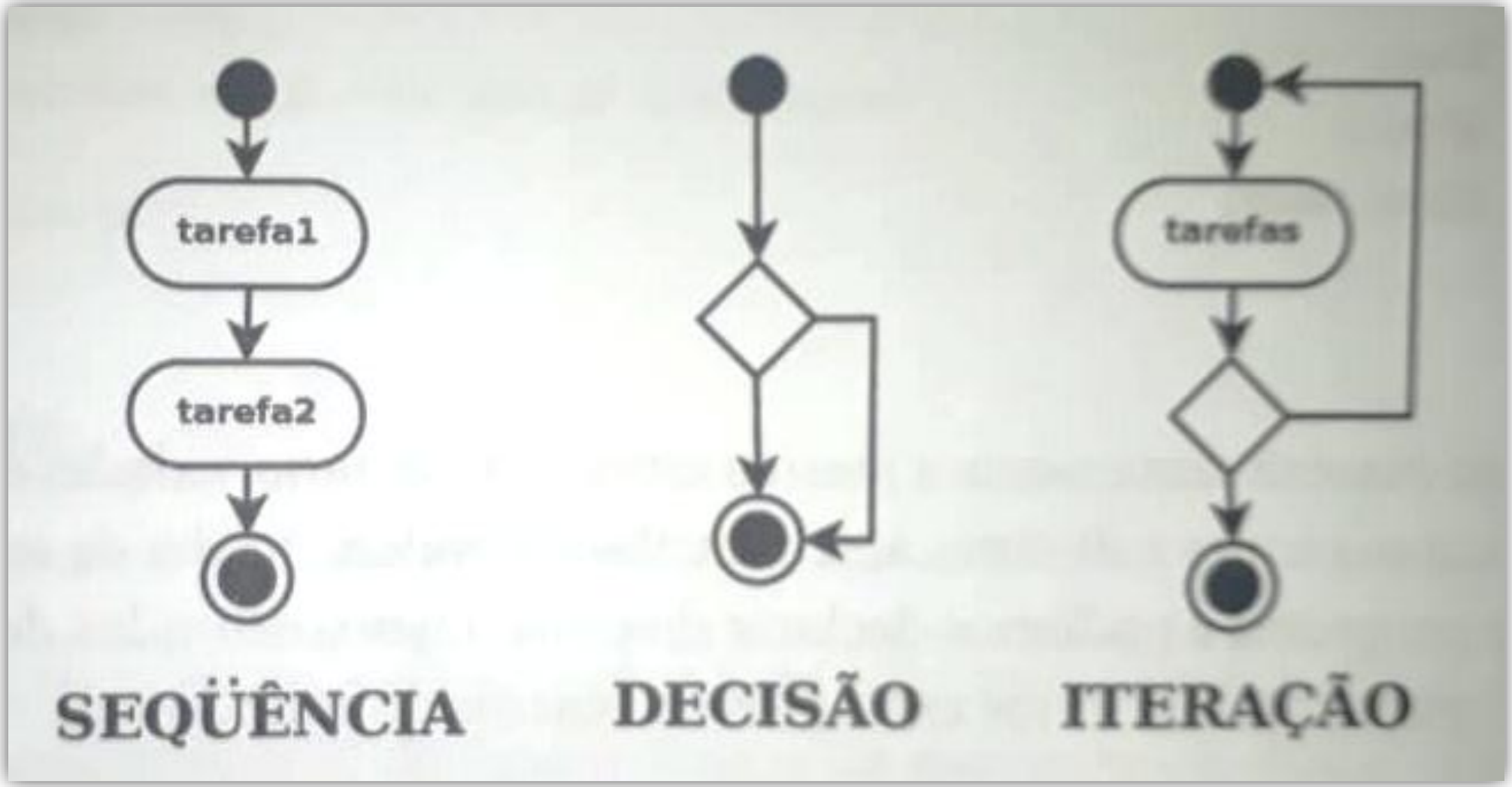
SW-I
SISTEMAS WEB I

Prof. Anderson Vanin

AULA 09 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM PHP - INTRODUÇÃO

Programação Estruturada



O que é Orientação à Objetos?

A orientação a objetos é um **paradigma** que representa uma filosofia para construção de sistemas. Em vez de construir um sistema formado por um conjunto de procedimentos e variáveis nem sempre agrupadas de acordo com o contexto, como se fazia em linguagens estruturadas (Cobol, Clipper, Pascal), na orientação a objetos utilizamos uma ótica mais **próxima do mundo real**.

Lidamos com objetos: **estruturas que carregam dados e comportamento próprio, além de trocarem mensagens entre si** com o objetivo de formar algo maior, um sistema.

Programação Orientada à Objetos PHP

Material de consulta: https://www.w3schools.com/php/php_oop_what_is.asp

Relembrar funções em PHP

Parte importante do paradigma de Orientação à Objetos, é entender muito bem o uso de funções na linguagem de estudo, no nosso caso PHP.

Uma função é um bloco de código nomeado que executa uma tarefa específica, possivelmente processando um conjunto de valores fornecidos a ela e/ou retornando algum valor.

Orientação à Objetos

Ao trabalharmos com orientação à objetos é fundamental entender o conceito de **classes e objetos**. **Uma classe é uma estrutura que define um tipo de dados, podendo conter atributos (variáveis) e também funções (métodos) para manipular esses atributos**.

Conceitos Fundamentais

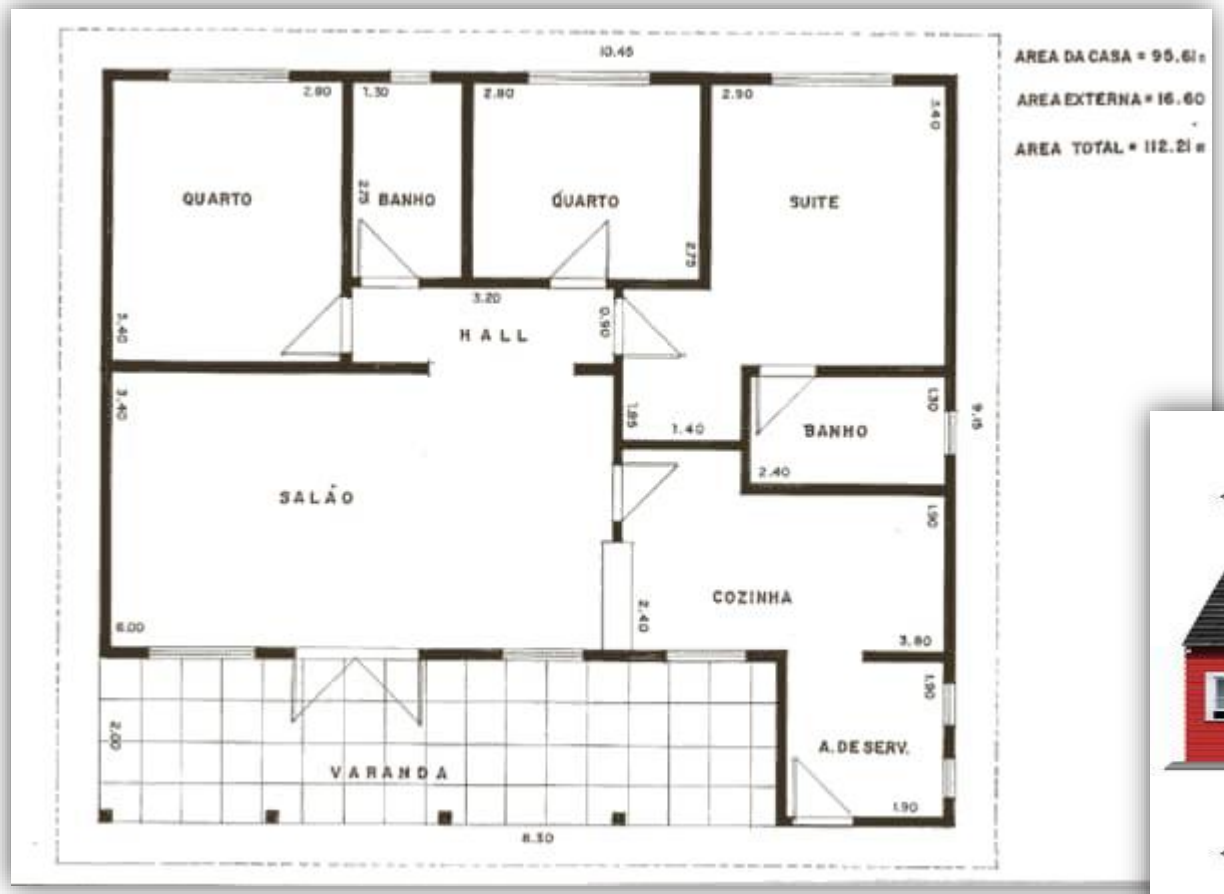
Objetos são porções de software descritas por variáveis e métodos. Objetos em software modelam os objetos do mundo real através do seu **comportamento** e seus **componentes**. Em linguagem estruturada, até então, vimos os conceitos de **atributos e funções**. Por outro lado, quando falamos em componentes de um objeto temos **métodos (relativos às funções)** e **atributos (relativos às variáveis)**.

Na prática, um objeto é uma representação em memória do computador de uma entidade do mundo real.

Classe

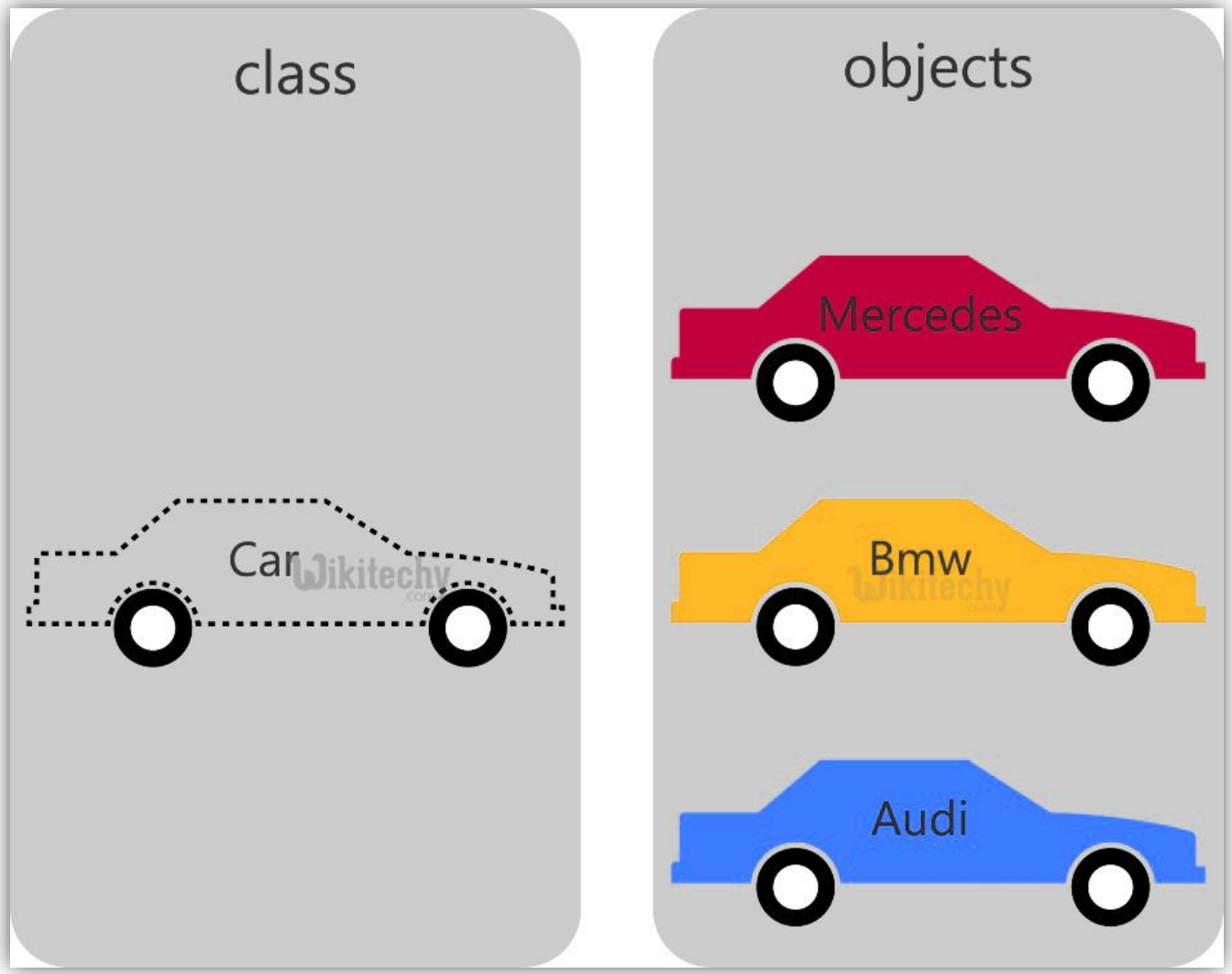
Classe é o modelo ou protótipo que define as variáveis e métodos comuns a um conjunto de objetos de certo tipo. Em geral, uma classe define como é cada objeto, sua estrutura e seu comportamento, enquanto os objetos são as entidades "vivas" dentro do programa. **Uma classe pode ter vários representantes (objetos) dentro de um mesmo programa, cada objeto com seus atributos em particular**.

Classe


10.45
AREA DA CASA = 95.61 m
AREA EXTERNA = 16.60
AREA TOTAL = 112.21 m
QUARTO
BANHO
QUARTO
SUITE
HALL
SALÃO
BANHO
COZINHA
A. DE SERV.
VARANDA
6.00
8.30

Classe

Classe



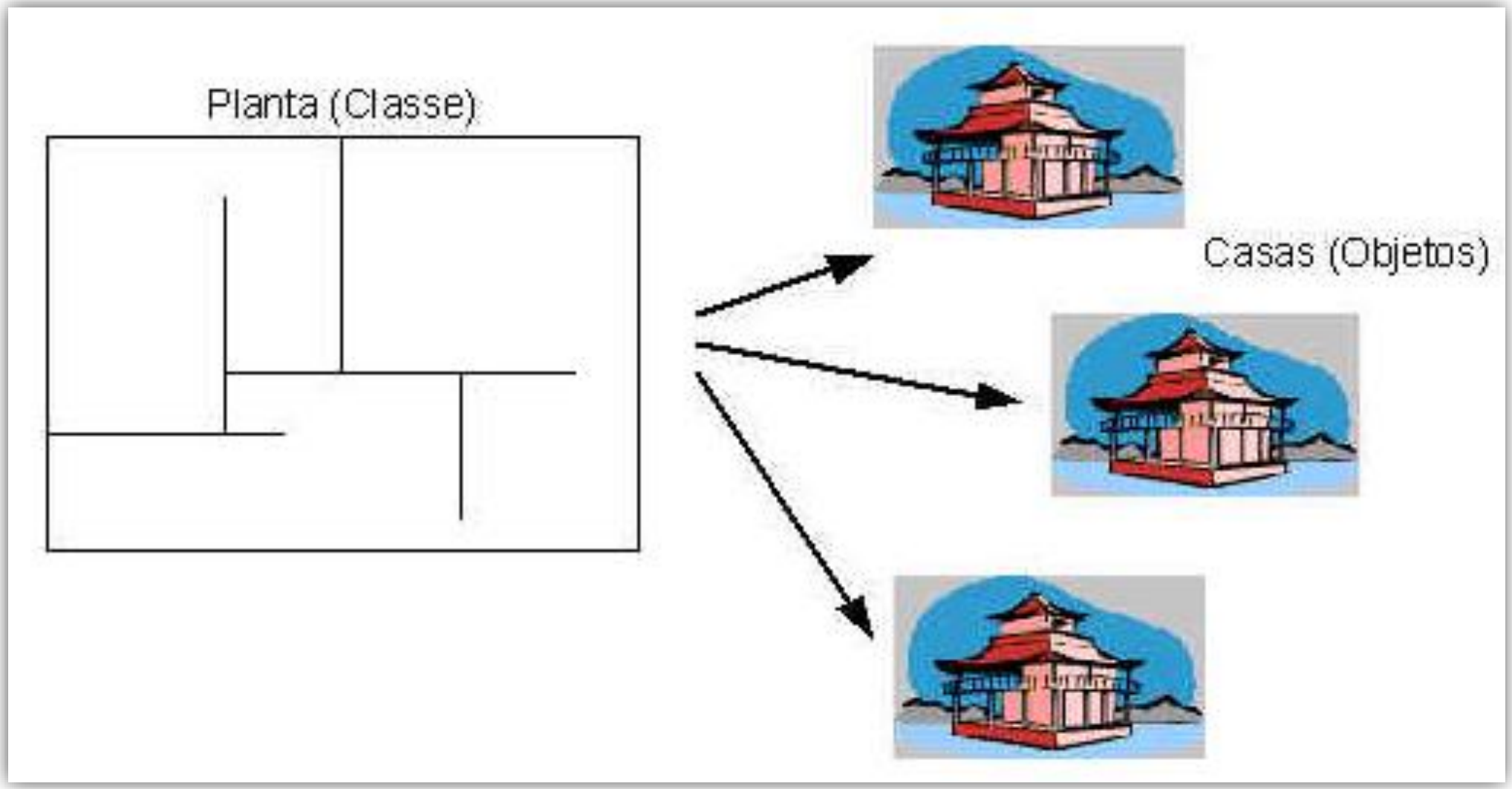
Objeto

Um objeto contém exatamente a mesma estrutura e as propriedades de uma classe, no entanto **sua estrutura é dinâmica**, **seus atributos podem mudar de valor durante a execução do programa e podemos declarar diversos objetos oriundos de uma mesma classe**.

Atributos

- Também conhecidos como variáveis de instância descrevem as características dos objetos.
- O tempo de vida dos atributos é igual ao tempo de vida do objeto.
- Cada objeto tem seus próprios valores de atributos em tempo de execução

Classes e Objetos

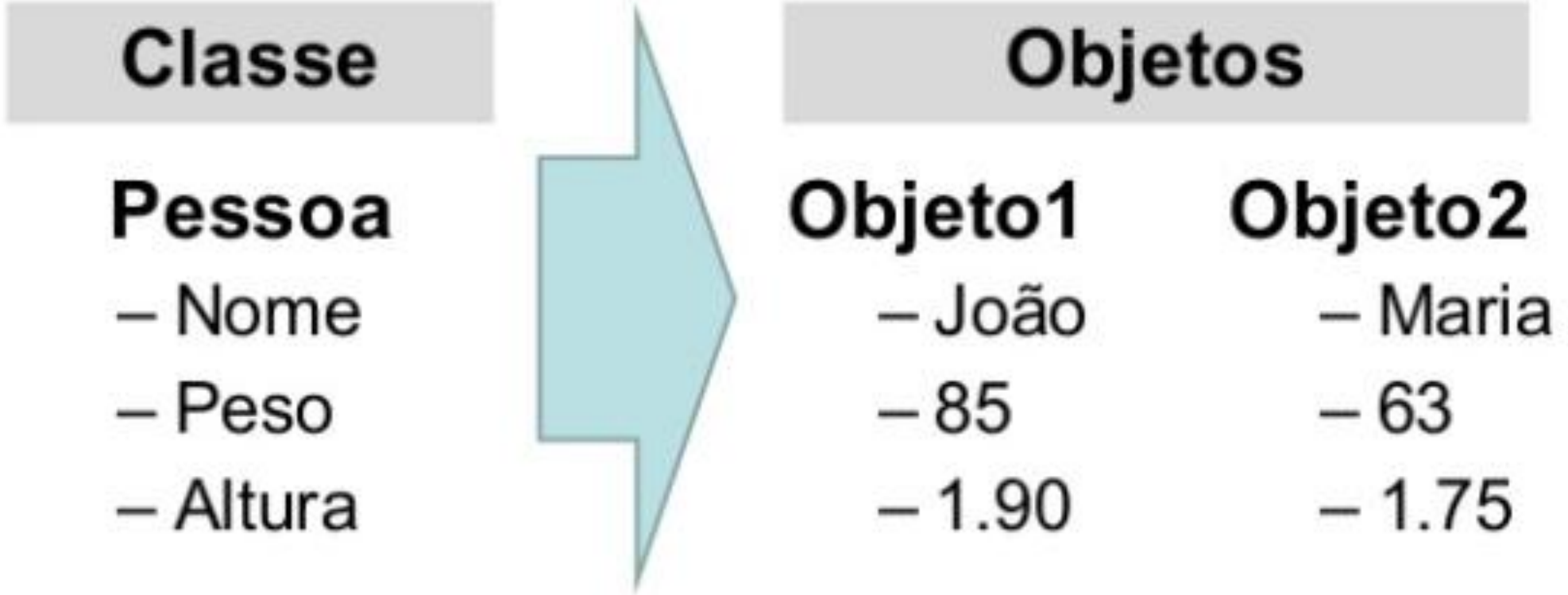


Instância de uma classe

Em programação orientada a objetos, chama-se **instância de uma classe**, um objeto cujo comportamento e estado são definidos pela classe. Lembre-se do exemplo do slide anterior onde a partir de uma classe Casa temos vários objetos diferentes que foram "**instanciados**".

Objetos

– **Objetos:** Para entender melhor...



Métodos

Os métodos são os componentes da classe que realizam as computações (funções).

- Métodos recebem parâmetros, ou seja, valores que podem ser utilizados durante a computação.
- Métodos podem ou não retornar um resultado, assim como funções não associadas a métodos.

Os métodos possuem uma **assinatura**, que corresponde a **uma descrição do que o método deve fazer** e ainda os valores necessários para a computação e o retorno da computação.

Funcionalidades (Comportamentos) - Métodos

Pessoa

- Andar
- Falar
- Comer

Computador

- Ligar
- Desligar
- Abrir Programa

Venda

- Concluir Venda
- Cancelar Venda
- Calcular Valor Total

Funcionário

- Salvar Funcionário
- Editar Funcionário
- Excluir Funcionário

Exemplo em PHP

Para criar uma classe em PHP use a seguinte sintaxe:

`<NomeDaClasse>.class.php`

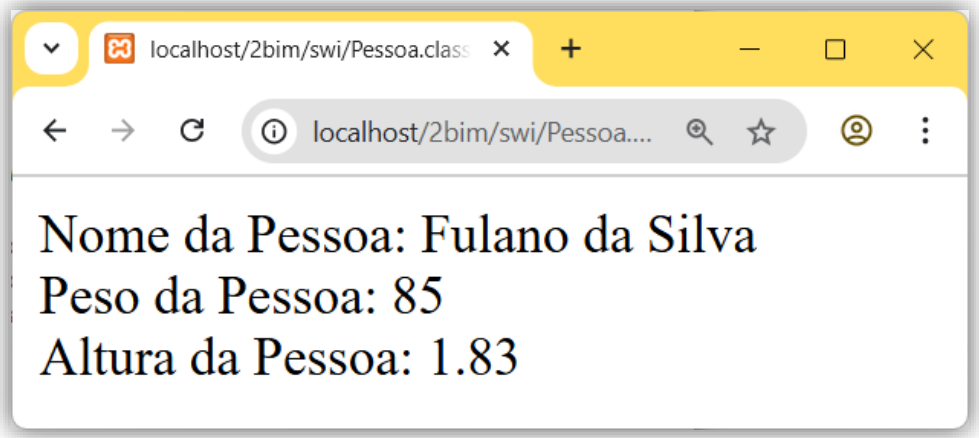
onde, preferencialmente o **nome da classe começa com letra Maiúscula**.

Use letras maiúsculas para os atributos, também.

Exemplo em PHP

Pessoa.class.php

```php
<?php
    // DEFINIÇÃO DA CLASSE PESSOA
    class Pessoa{
        public $Nome;
        public $Peso;
        public $Altura;

        // MÉTODO DA CLASSE PESSOA (UMA AÇÃO QUE A CLASSE PODE EXECUTAR)
        public function MostraDadosPessoa(){
            echo "Nome da Pessoa: " . $this->Nome . "<br>";
            echo "Peso da Pessoa: " . $this->Peso . "<br>";
            echo "Altura da Pessoa: " . $this->Altura . "<br>";
        }
    }

    // INSTANCIANDO UM OBJETO DA CLASSE PESSOA
    $fulano = new Pessoa;

    // ATRIBUINDO VALORES AOS ATRIBUTOS DO OBJETO CRIADO
    $fulano->Nome = 'Fulano da Silva';
    $fulano->Peso = 85;
    $fulano->Altura = 1.83;

    // CHAMANDO UM MÉTODO DA CLASSE PESSOA
    $fulano->MostraDadosPessoa();
?>
```



Exemplo em PHP

Vamos imaginar uma lâmpada...

Exemplo em PHP

Vamos imaginar uma lâmpada...Dessa forma temos a classe **Lampada**.

Exemplo em PHP

Quais seriam os possíveis atributos (características) desta lâmpada? Vamos escolher algumas...

Exemplo em PHP

Quais seriam os possíveis **atributos** (características) desta lâmpada? Vamos escolher algumas...

-**Fabricante**

-**Tensão(V) (110, 220)**

-**Potência(W) (15, 20, 25,...)**

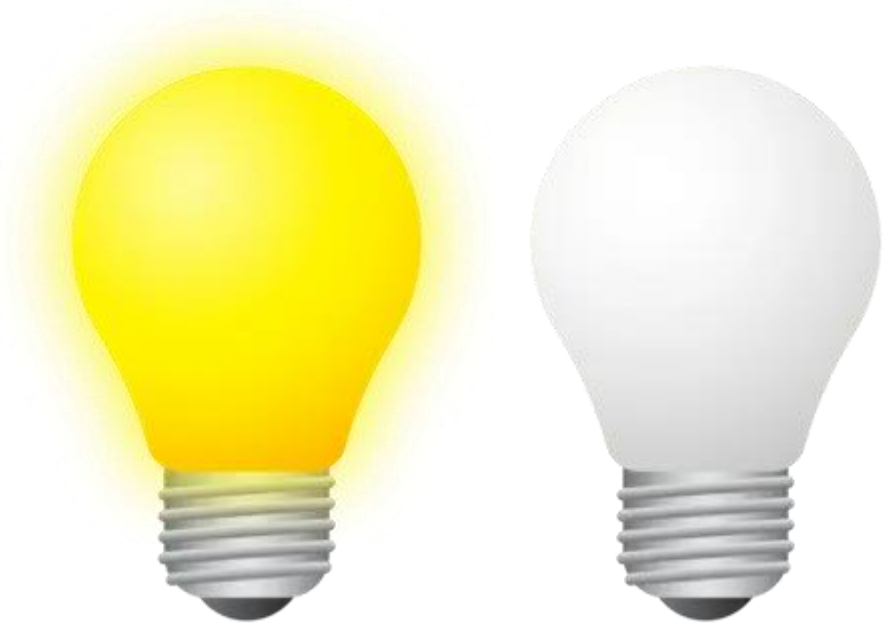
-**Cor (branca, vermelha, verde...)**

Exemplo em PHP

Agora vamos pensar em quais seriam as possíveis ações (**métodos**) que uma lâmpada pode fazer...

-**Ligar**

-**Desligar**

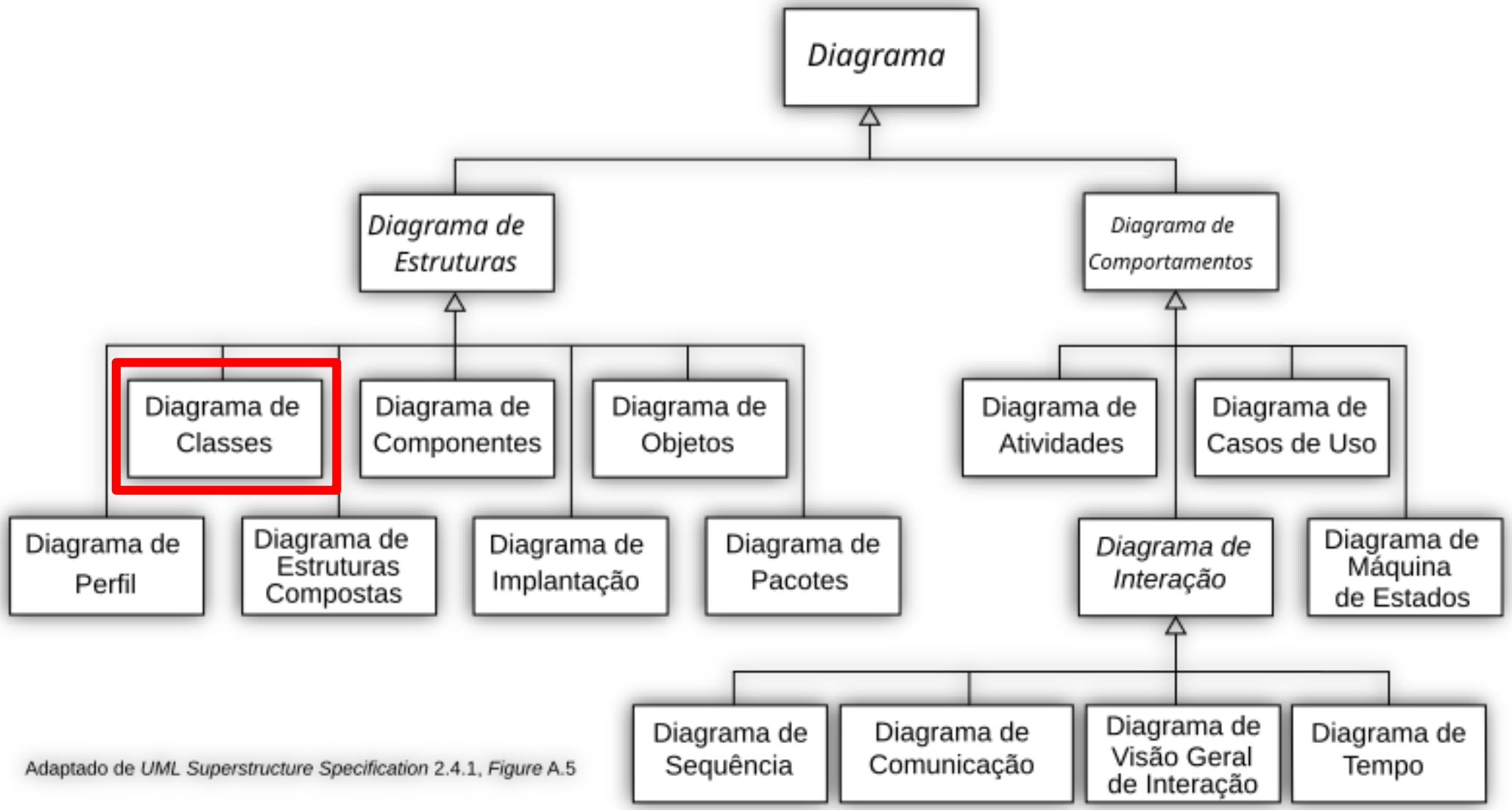


Exemplo em PHP - UML

Um próximo passo importante é a documentação do projeto, e para isso utilizaremos a **UML** (Unified Modeling Language).

A UML é uma linguagem de **modelagem visual** padrão usada em desenvolvimento de software e outras áreas. Ela oferece uma forma de representar visualmente a estrutura, comportamento e funcionalidades de um sistema de forma clara e consistente. Os diagramas UML são ferramentas essenciais para a comunicação e colaboração entre desenvolvedores, arquitetos e outros stakeholders.

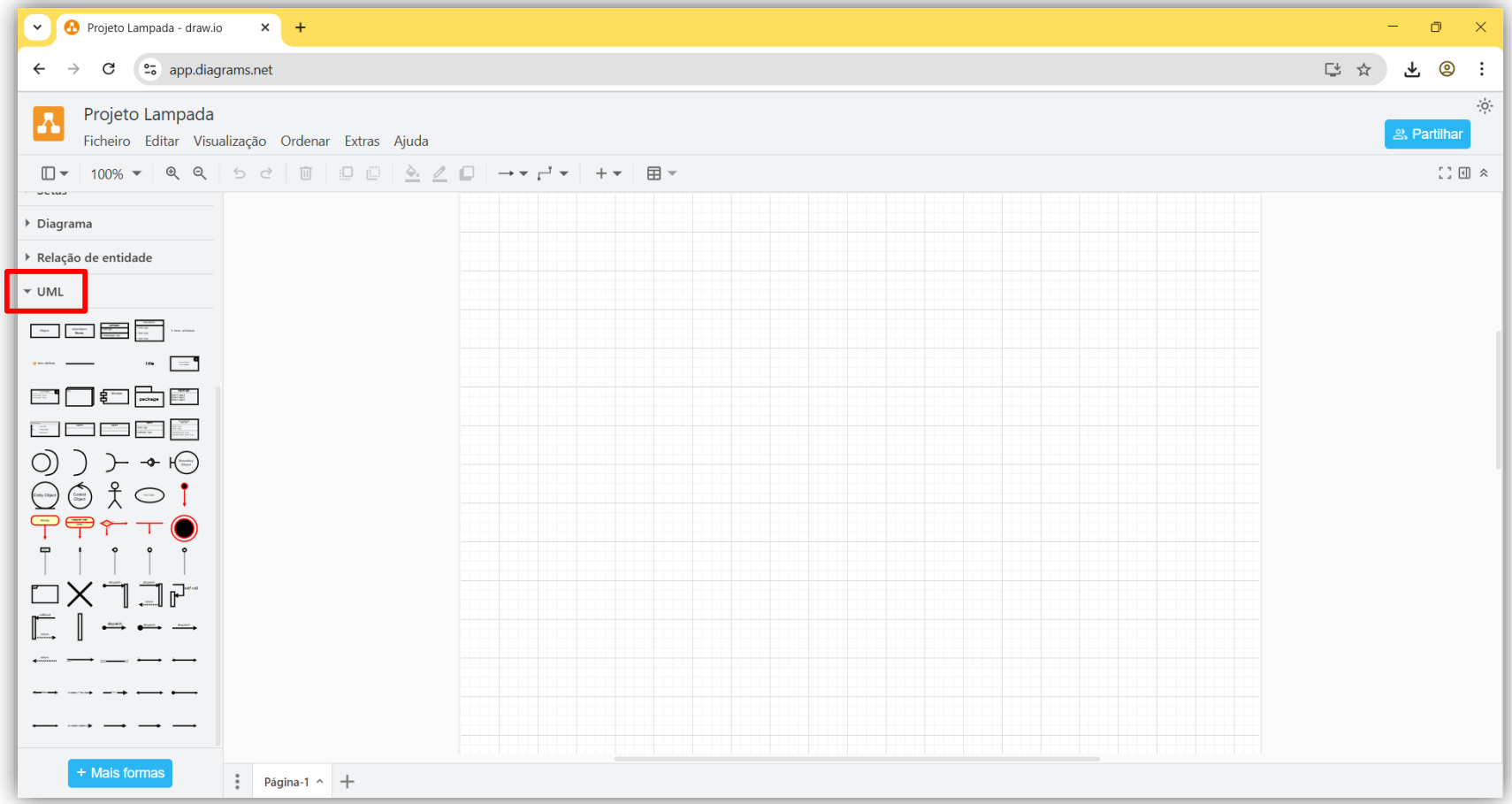
Exemplo em PHP - UML



Adaptado de *UML Superstructure Specification* 2.4.1, *Figure* A.5

Exemplo em PHP - UML

Para desenhar diagramas em UML, podemos utilizar a ferramenta online Draw IO (https://app.diagrams.net/)



Exemplo em PHP - UML

+	**Público**
-	**Privado**
#	**Protegido**

String	Caracteres alfanuméricos
int	Números inteiros
void	Significa um método sem retorno

Exemplo em PHP - UML

Lampada.class.php

```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;
    }
?>
```

Exemplo em PHP

Lampada.class.php

```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;

        // MÉTODOS DA LAMPADA
        public function Ligar(){
            // AÇÕES A SEREM EXECUTADAS PARA LIGAR A LAMPADA...
        }
        public function Desligar(){
            // AÇÕES A SEREM EXECUTADAS PARA DESLIGAR A LAMPADA...
        }
    }
?>
```

Exemplo em PHP

Lampada.class.php

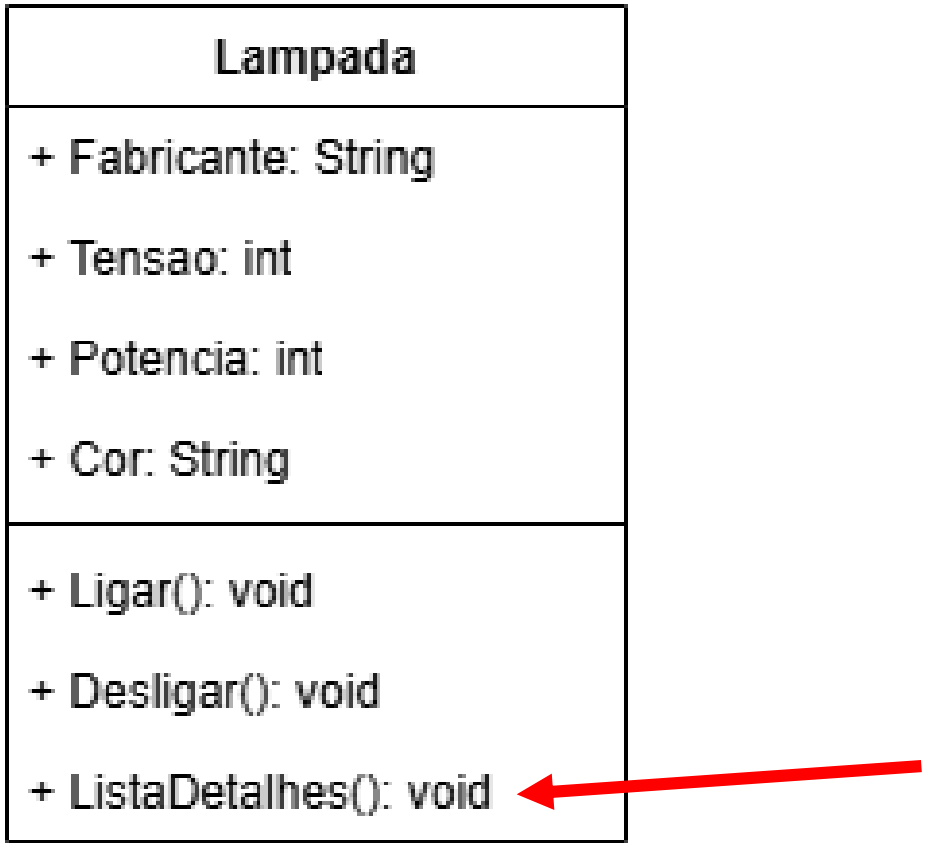
```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;

        // MÉTODOS DA LAMPADA
        public function Ligar(){
            // AÇÕES A SEREM EXECUTADAS PARA LIGAR A LAMPADA...
            echo "A LÂMPADA ESTÁ LIGADA AGORA!";
        }
        public function Desligar(){
            // AÇÕES A SEREM EXECUTADAS PARA DESLIGAR A LAMPADA...
            echo "VOCÊ DESLIGOU A LÂMPADA!";
        }
    }
?>
```

Exemplo em PHP

Podemos criar métodos adicionais, também, como por exemplo, um método que mostre todas as características de uma lâmpada. Aqui vamos criar o método **ListaDetalhes**.

Lampada.class.php

```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;

        // MÉTODOS DA LAMPADA
        public function Ligar(){
            // AÇÕES A SEREM EXECUTADAS PARA LIGAR A LAMPADA...
            echo "A LÂMPADA ESTÁ LIGADA AGORA!";
        }
        public function Desligar(){
            // AÇÕES A SEREM EXECUTADAS PARA DESLIGAR A LAMPADA...
            echo "VOCÊ DESLIGOU A LÂMPADA!";
        }
        public function ListaDetalhes(){
            echo "Fabricante: ".$this->Fabricante."<br>";
            echo "Tensão de Alimentação (V): ".$this->Tensao."<br>";
            echo "Potência (W): ".$this->Potencia."<br>";
            echo "Cor: ".$this->Cor."<br>";
        }
    }
?>
```



Exemplo em PHP

Agora que temos uma classe chamada Lampada, seus atributos e alguns métodos criados, podemos instanciar um ou mais objetos a partir desta classe.

A instância de um novo objeto, pode ser feito no mesmo arquivo da classe ou pode ser incluída em um outro arquivo php (que esteja na mesma pasta do projeto). Vamos fazer a inclusão em um novo arquivo.

Exemplo em PHP

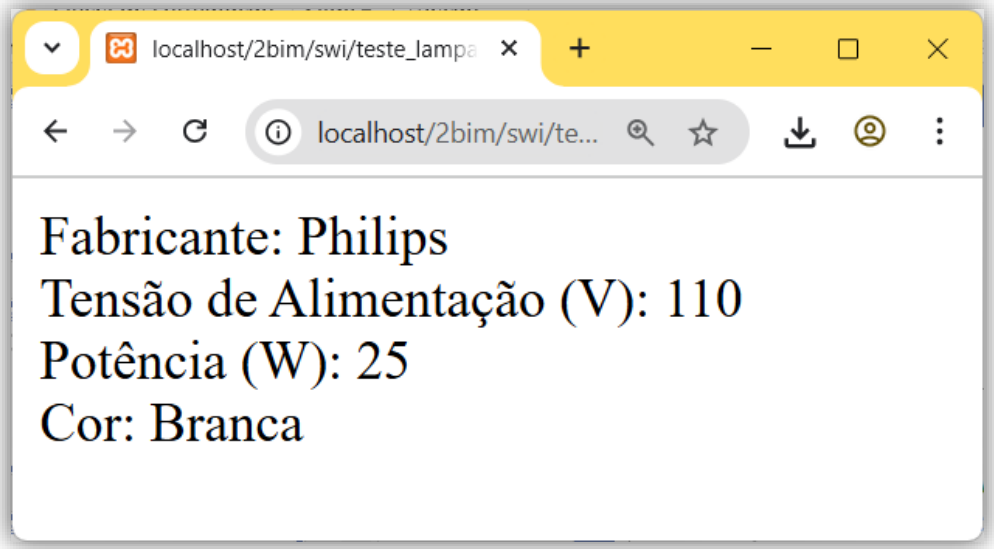
Agora que temos uma classe chamada Lampada, seus atributos e alguns métodos criados, podemos instanciar um ou mais objetos a partir desta classe.

A instância de um novo objeto, pode ser feito no mesmo arquivo da classe ou pode ser incluída em um outro arquivo php (que esteja na mesma pasta do projeto). Vamos fazer a inclusão em um novo arquivo.

Exemplo em PHP

teste_lampada.php

```php
<?php
    // INSERE A CLASSE CRIADA - LAMPADA
    include_once 'Lampada.class.php';

    // INSTANCIA DE UM NOVO OBJETO DA CLASSE LAMPADA
    $lamp1 = new Lampada;

    // ATRIBUINDO NOVOS VALORES PARA O OBJETO CRIADO
    $lamp1->Fabricante = "Philips";
    $lamp1->Tensao = 110;
    $lamp1->Potencia = 25;
    $lamp1->Cor = "Branca";

    // CHAMANDO O MÉTODO QUE MOSTRA OS DETALHES INSTANCIADOS NESTE OBJETO
    $lamp1->ListaDetalhes();
?>
```

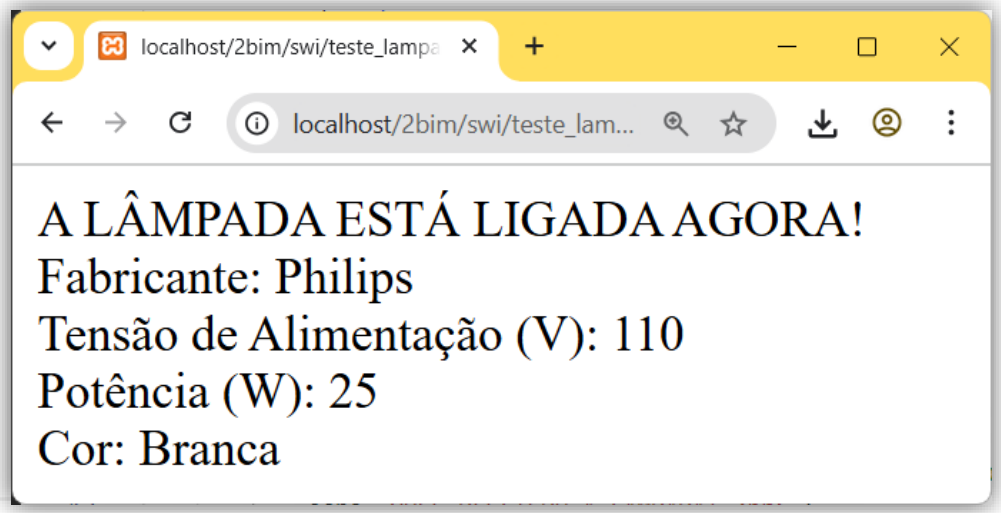


Exemplo em PHP

Agora temos a seguinte situação: Ligar a lâmpada e ver os detalhes deste objeto. A sequência será: chamar o método para ligar a lâmpada e depois o método que mostra os detalhes.

teste_lampada.php

```php
<?php
    // INSERE A CLASSE CRIADA - LAMPADA
    include_once 'Lampada.class.php';

    // INSTANCIA DE UM NOVO OBJETO DA CLASSE LAMPADA
    $lamp1 = new Lampada;

    // ATRIBUINDO NOVOS VALORES PARA O OBJETO CRIADO
    $lamp1->Fabricante = "Philips";
    $lamp1->Tensao = 110;
    $lamp1->Potencia = 25;
    $lamp1->Cor = "Branca";

    // CHAMANDO O MÉTODO PARA LIGAR A LAMPADA
    $lamp1->Ligar();

    // CHAMANDO O MÉTODO QUE MOSTRA OS DETALHES INSTANCIADOS NESTE OBJETO
    $lamp1->ListaDetalhes();
?>
```

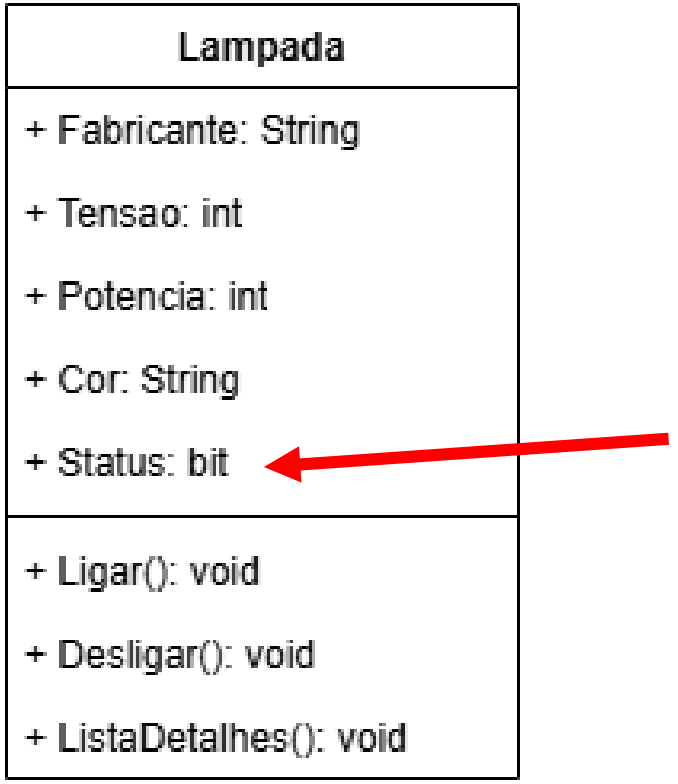


Exemplo em PHP

Seria interessante possuir um **atributo que guardasse o status dessa lâmpada** (ligada ou desligada) toda vez que acionássemos os métodos Ligar ou Desligar. Assim poderíamos criar uma lógica que se a lâmpada estivesse ligada, o status poderia ser **1** ou caso contrário **0**.

Lampada.class.php

```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;
        public $Status;

        // MÉTODOS DA LAMPADA
        public function Ligar(){
            // AÇÕES A SEREM EXECUTADAS PARA LIGAR A LAMPADA...
            echo "A LÂMPADA ESTÁ LIGADA AGORA! <br>";
        }
        public function Desligar(){
```



Exemplo em PHP

Agora vamos trabalhar na lógica para que quando o método **Ligar** for chamado, o atributo **`$Status`** tenha seu valor modificado para `1`. Já quando o método **Desligar** for chamado, alteraremos o valor para `0`.

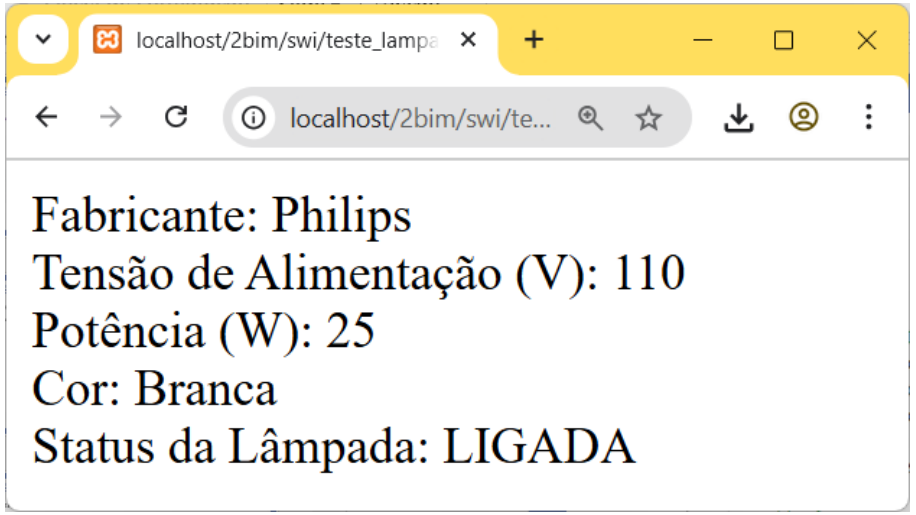
Exemplo em PHP

Lampada.class.php

```php
<?php
    class Lampada{

        // ATRIBUTOS DA LAMPADA
        public $Fabricante;
        public $Tensao;
        public $Potencia;
        public $Cor;
        public $Status;

        // MÉTODOS DA LAMPADA
        public function Ligar(){
            // AÇÕES A SEREM EXECUTADAS PARA LIGAR A LAMPADA...
            //echo "A LÂMPADA ESTÁ LIGADA AGORA! <br>";
            $this->Status = 1;
        }
        public function Desligar(){
            // AÇÕES A SEREM EXECUTADAS PARA DESLIGAR A LAMPADA...
            //echo "VOCÊ DESLIGOU A LÂMPADA! <br>";
            $this->Status = 0;
        }
        public function ListaDetalhes(){
            echo "Fabricante: ".$this->Fabricante."<br>";
            echo "Tensão de Alimentação (V): ".$this->Tensao."<br>";
            echo "Potência (W): ".$this->Potencia."<br>";
            echo "Cor: ".$this->Cor."<br>";
            if($this->Status == 1){
                echo "Status da Lâmpada: LIGADA <br>";
            }else{
                echo "Status da Lâmpada: DESLIGADA <br>";
            }
        }
    }
?>
```

teste_lampada.php

```php
<?php
    // INSERE A CLASSE CRIADA - LAMPADA
    include_once 'Lampada.class.php';

    // INSTANCIA DE UM NOVO OBJETO DA CLASSE LAMPADA
    $lamp1 = new Lampada;

    // ATRIBUINDO NOVOS VALORES PARA O OBJETO CRIADO
    $lamp1->Fabricante = "Philips";
    $lamp1->Tensao = 110;
    $lamp1->Potencia = 25;
    $lamp1->Cor = "Branca";

    // CHAMANDO O MÉTODO PARA LIGAR A LAMPADA
    $lamp1->Ligar();

    // CHAMANDO O MÉTODO QUE MOSTRA OS DETALHES INSTANCIADOS NESTE OBJETO
    $lamp1->ListaDetalhes();
?>
```

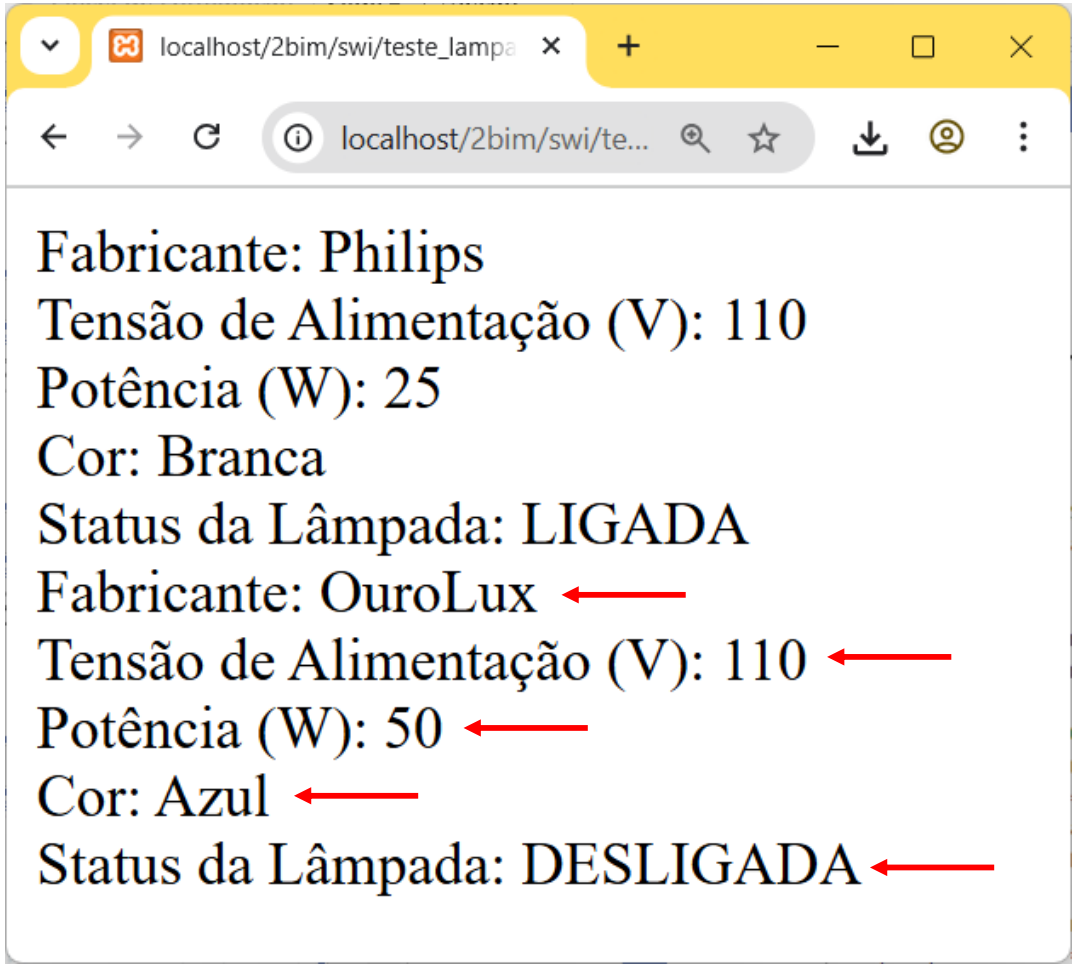


Exemplo em PHP

Agora vamos tentar instanciar mais um objeto, só que com características diferentes.

teste_lampada.php

```php
<?php
    // INSERE A CLASSE CRIADA - LAMPADA
    include_once 'Lampada.class.php';

    // INSTANCIA DE UM NOVO OBJETO DA CLASSE LAMPADA
    $lamp1 = new Lampada;
    $lamp2 = new Lampada;

    // ATRIBUINDO NOVOS VALORES PARA O OBJETO CRIADO
    $lamp1->Fabricante = "Philips";
    $lamp1->Tensao = 110;
    $lamp1->Potencia = 25;
    $lamp1->Cor = "Branca";

    $lamp2->Fabricante = "OuroLux";
    $lamp2->Tensao = 110;
    $lamp2->Potencia = 50;
    $lamp2->Cor = "Azul";

    // CHAMANDO O MÉTODO PARA LIGAR A LAMPADA
    $lamp1->Ligar();
    $lamp2->Desligar();

    // CHAMANDO O MÉTODO QUE MOSTRA OS DETALHES INSTANCIADOS NESTE OBJETO
    $lamp1->ListaDetalhes();
    $lamp2->ListaDetalhes();
?>
```



Exercício 01 – **Não faça no PHP! Escreva a mão.**

Defina pelo menos **5 atributos** para as classes a seguir:

- **Cliente**
- **Venda**
- **Produto**
- **Aluno**
- **Curso**

Exercício 02 – Executar no PHP

Crie uma classe **Carro** com os **atributos públicos marca, modelo, combustível, portas e capacidade_tanque**. Depois crie **5 instâncias da classe Carro** e **defina os valores para cada atributo**. Por fim tente **implementar um método** na classe Carro que **exiba todos os detalhes** (atributos) de cada instância criada.