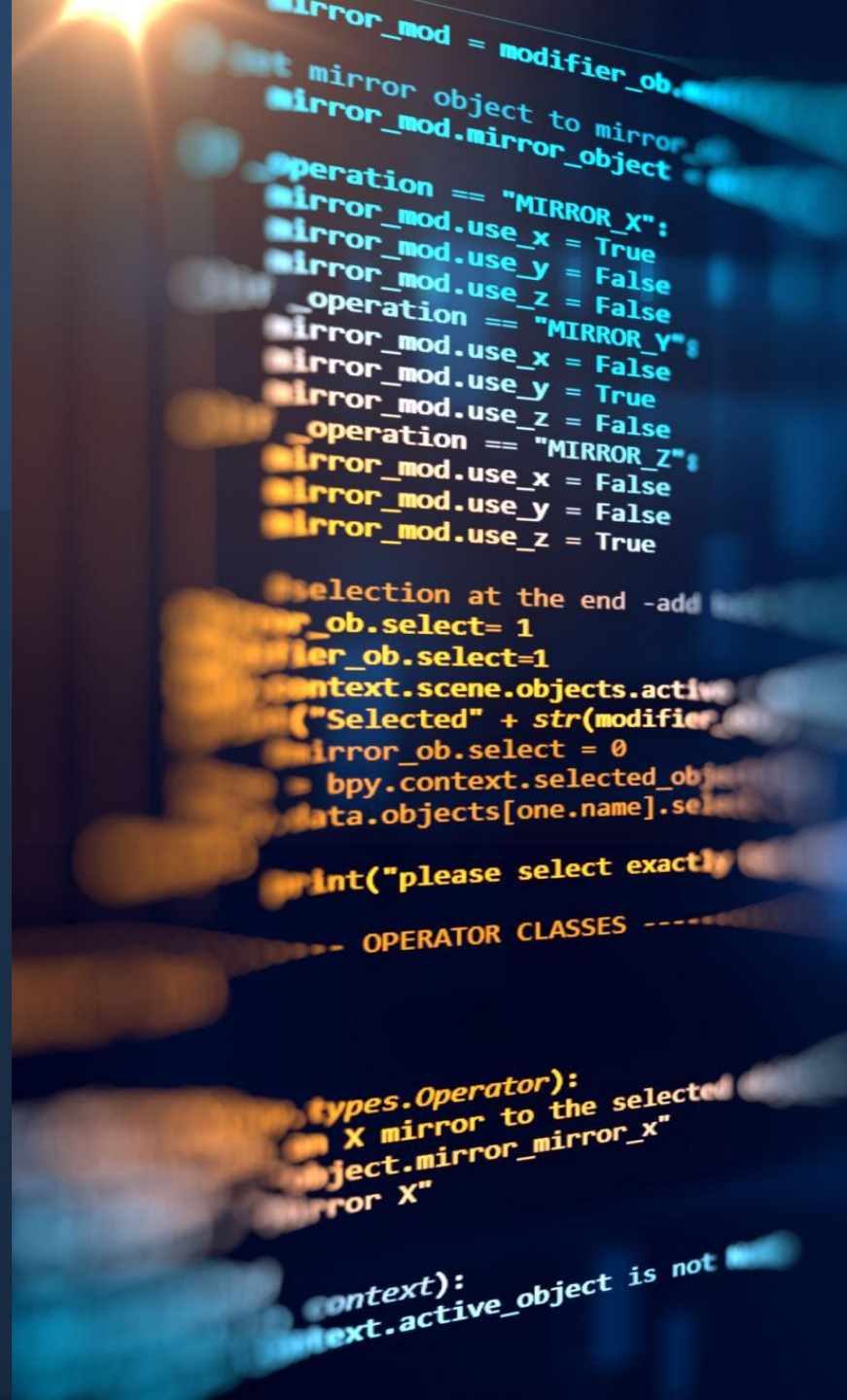
Prof. Ricardo e Prof. Wesley

2º Mtec DS

Programação Web II



PHP Orientação a Objetos

PHP O que é OOP?

OOP significa Programação Orientada a Objetos.

A programação procedural trata de escrever procedimentos ou funções que realizam operações nos dados, enquanto a programação orientada a objetos trata da criação de objetos que contêm dados e funções.

A programação orientada a objetos tem várias vantagens sobre a programação procedural:

- OOP é mais rápido e fácil de executar
- OOP fornece uma estrutura clara para os programas
- OOP ajuda a manter o código PHP DRY "Don't Repeat Yourself" e torna o código mais fácil de manter, modificar e depurar
- OOP torna possível criar aplicativos totalmente reutilizáveis com menos código e menor tempo de desenvolvimento

Dica: O princípio "Don't Repeat Yourself" (DRY) é sobre como reduzir a repetição de código. Você deve extrair os códigos comuns para o aplicativo, colocá-los em um único lugar e reutilizá-los em vez de repeti-los.

PHP - O que são classes e objetos?

Classes e objetos são os dois aspectos principais da programação orientada a objetos.

Observe a ilustração a seguir para ver a diferença entre classe e objetos:

Another example:

Portanto, uma classe é um modelo para objetos e um objeto é uma instância de uma classe.

Quando os objetos individuais são criados, eles herdam todas as propriedades e comportamentos da classe, mas cada objeto terá valores diferentes para as propriedades.

OOP Case
Vamos supor que temos uma classe chamada Fruta. Uma fruta pode ter propriedades como nome, cor, peso, etc. Podemos definir variáveis como $ nome, $ cor e $ peso para conter os valores dessas propriedades.

Quando os objetos individuais (maçã, banana, etc.) são criados, eles herdam todas as propriedades e comportamentos da classe, mas cada objeto terá valores diferentes para as propriedades.

Defina uma classe
Uma classe é definida usando a class palavra - chave, seguida pelo nome da classe e um par de chaves ({}). Todas as suas propriedades e métodos estão entre colchetes:

```php
<?php
class Fruta {
  // aqui vai o código...
}
?>
```

Abaixo, declaramos uma classe chamada Fruit que consiste em duas propriedades ($ name e $ color) e dois métodos set_name () e get_name () para definir e obter a propriedade $ name:

```php
<?php
class Fruta {
  // Propriedades
  public $name;
  public $color;

  // Métodos
  function set_name($name) {
    $this->name = $name;
  }
  function get_name() {
    return $this->name;
  }
}
?>
```

Nota: Em uma classe, as variáveis são chamadas de propriedades e as funções são chamadas de métodos!

Definir objetos

As classes não são nada sem objetos! Podemos criar vários objetos de uma classe. Cada objeto possui todas as propriedades e métodos definidos na classe, mas eles terão diferentes valores de propriedade.

Os objetos de uma classe são criados usando a palavra – chave new.

No exemplo abaixo, $maca e $banana são instâncias da classe Fruta:

```php
<?php
class Fruta {
  // Properties
  public $nome;
  public $cor;

  // Methods
  function set_name($nome) {
    $this->nome = $nome;
  }
  function get_name() {
    return $this->nome;
  }
}

$maca = new Fruta();
$banana = new Fruta();
$maca->set_name('Maçã');
$banana->set_name('Banana');

echo $maca->get_name();
echo "<br>";
echo $banana->get_name();
?>
```

No exemplo abaixo, adicionamos mais dois métodos à classe Fruta, para definir e obter a propriedade $cor:

```php
<?php
class Fruta {
  // Propriedades
  public $nome;
  public $cor;
  // Métodos
  function set_name($nome) {
    $this->nome = $nome;
  }
  function get_name() {
    return $this->nome;
  }
 function set_color($cor) {
    $this->cor = $cor;
  }
  function get_color() {
    return $this->cor;
  }
}
$maca = new Fruta();
$maca->set_name('Maçã');
$maca>set_color('Vermelho');
echo "Nome: " .$maca->get_name();
echo "<br>";
echo "Cor: " .$maca->get_color();
?>
```

PHP - a palavra-chave $ this

A palavra-chave $ this refere-se ao objeto atual e só está disponível dentro de métodos. Veja o seguinte exemplo:

```php
<?php
class Fruta {
  public $name;
}
$maca = new Fruta();
?>
```

Então, onde podemos alterar o valor da propriedade $nome? Existem duas maneiras:
1. Dentro da classe (adicionando um método set_name () e use $ this):

```php
<?php
class Fruta {
  public $nome;
  function set_name($nome) {
    $this->nome = $nome;
  }
}
$maca = new Fruit();
$maca->set_name(“Maçã");
?>
```

2. Fora da classe (alterando diretamente o valor da propriedade):

```php
<?php
class Fruta {
  public $nome;
}
$maca = new Fruta();
$maca->name = "Apple";
?>
```

PHP - instanceof

Você pode usar a palavra-chave instanceof para verificar se um objeto pertence a uma classe específica:

```php
<?php
$maca = new Fruta();
var_dump($apple instanceof Fruta);
?>
```