Fundamentos da OO com PHP

- Embora a codificação de um sistema não seja uma das fases do projeto, ambos estão muito próximos e guardam uma relação estreita entre si;
- Todos os diagramas de modelagem com UML utilizados, tanto na etapa de análise como na etapa de projeto, servem como base para a criação do código, que é um dos passos finais no processo da elaboração de aplicações;
- Diagramas de classe e diagramas de sequência transmitem, quase que diretamente, suas informações para a fase de codificação, bastando fazer os ajustes adequados à a sintaxe e semântica da linguagem de programação escolhida;
- A seguir, detalharemos as características principais da utilização do paradigma da OO com a linguagem de programação PHP.

Implementação de classes em PHP

- Em PHP, o operador que relaciona um objeto a um atributo (variáveis de instância ou variáveis de classe) ou método é o símbolo **->**;
- Toda classe, em PHP, é dada pela palavra-chave "**class**", seguida pelo nome da classe e um par de **chaves**;
- Dentro das chaves, declaramos os atributos e os métodos que compõem a classe;
- Para acessarmos os atributos de uma classe ou de uma instância de dentro de um método, usamos o operador "**$this**";
- **Atributos** e **métodos** de uma classe podem ter os escopos de visibilidade **public**, **protected** e **private**;
- No eslaide abaixo, criaremos, como exemplo, uma classe chamada Professor, com alguns atributos e métodos. A classe tem três atributos: *nome* do aluno e suas duas últimas *notas* de uma determinada unidade curricular. E apresenta dois métodos: um deles recebe o nome do aluno e suas duas notas como parâmetros, atribui o parâmetro nome ao atributo da classe (por meio de $this) e usa os outros dois parâmetros para calcular e escrever a média. O outro método é responsável por devolver ao código que o invocou, por meio da cláusula **return**, o nome do aluno.

Implementação de classes em PHP - exemplo

```php
<?php
 class Professor
  {
  public    $aluno;
  protected $nota1;
  private   $nota2;

  private function calcularMedia($nota1, $nota2, $nome)
        {
        $media = ($nota1 + $nota2) / 2;
        $this->aluno = $nome;
        echo $media;
        }

  function getNome()
        {
        return $this->aluno;
        }
  }?>
```

Criação de objetos a partir da instanciação de classes em PHP

- Para criarmos um objeto em PHP a partir de determinada classe, usamos o operador **new**. Veja o código abaixo, tomando, como exemplo, a classe definida no eslaide anterior. Nele, criamos um objeto chamado *$prof1* a partir da instanciação da classe dada. Então, uma vez que o objeto foi construído, podemos utilizar seus métodos e atributos para realizarmos as operações necessárias. Observar que não utilizamos o **$** antes dos atributos da classe:

```php
<?php
    //área de instanciação da classe
    $prof1 = new Professor;
    $prof1->calcularMedia(8,6,"Maria");
    $nomeDoAluno = $prof1->getNome();
    echo $nomeDoAluno;
    $prof1->aluno = "Joana";
?>
```

Usando construtor para inicializar os atributos da classe em PHP

- Para criarmos um objeto em PHP e, ao mesmo tempo, inicializarmos seus atributos, usamos o operador **new** com um novo método escrito assim: **__construct().** Veja o código abaixo, que utiliza nome de usuário, senha e e-mail para definição da classe usuário, utilizando um construtor:

```
<?php
    //área de definição da classe
    class Usuario {
        $login; var $senha; var $email;

    function __construct($log, $passw, $mail) {
        $this->login = $log;
        $this->senha = $passw;
        $this->email = $mail;
        }

    function getLogin() {
        return $this->login;
         }
    }
?>
```

Usando construtor para inicializar os atributos da classe em PHP

- A partir do modelo criado no eslaide anterior, podemos utilizar o construtor (new Usuario()) para inicializarmos os dados no momento em que criamos um novo objeto. Veja o código abaixo:

```php
<?php
  //área de instanciação da classe
  $usuario1 = new Usuario("aluno-ifsc", "123", "aluno-
  ifsc@ifsc.edu.br");
  $usuario2 = new Usuario("maria", "123",
  "maria@ifsc.edu.br");

  echo $usuario1->getLogin();
  echo "<br>";
  echo $usuario2->getLogin();
?>
```