Curso de PHP

Palestrante: Marco Aurélio
Jefson Farias

Título

Agenda

- Propriedades e Métodos Static
- Interface
- Autoload

Título

Propriedades e Métodos Static

- Para acessar qualquer método ou propriedade de uma classe temos que criar uma instância, ou seja usar:
- $objeto = new Classe();
- De outra forma não podemos acessar. Mas existe uma exceção para métodos e propriedades static. Estes podem ser acessados diretamente sem a criação de instância. Um membro static é como um membro global.

Título

Propriedades e Métodos Static

Classe Usuario sem o static

```
class Usuario {
    private $nome;
    private $email;
    private $senha;

    function __construct($nome, $email, $senha) {
        $this->nome = $nome;
        $this->email = $email;
        $this->senha = $senha;
    }

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

Classe Usuario com static

```
class Usuario {

    public static $nome;
    private $email;
    private $senha;

    function __construct($nome, $email, $senha) {
        self::$nome = $nome;
        $this->email = $email;
        $this->senha = $senha;
    }

    function getNome() {
        return self::$nome;
    }
```

self::nome tem o mesmo efeito de Usuario::nome. A palavra reservada self:: é usada para referenciar elementos estáticos da classe.
Enquanto que $this-> é usado para referenciar elementos não estáticos.

Título

Propriedades e Métodos Static

- Acessando métodos fora da classe com o static

Classe Helper

```php
class Helper {

    public static function getString($string) {
        if (is_string($string)) {
            return TRUE;
        } else {
            return FALSE;
        }
    }

}
```

Acessando a função sem instanciar a classse

```php
$usuario = new Usuario("jefson", "marco@gmail.com", "123456");
$nome = $usuario->getNome();

echo Helper::getString($nome);
```


Título

Interface

- Interface é uma classe vazia que contém somente as declarações dos métodos (corpo em branco).
- Qualquer classe que implemente uma interface precisa conter todas as declarações dos métodos. Uma classe usa uma interface passando a palavra reservada "implements".
- Lembrando que nas interfaces podemos apenas declarar métodos mas não podemos escrever o corpo dos métodos, que obrigatoriamente precisam permanecer vazios.

Título

Interface para a Classe Usuario

- Criação da interface

```
interface CRUD {

    public function read(Usuario $u);

    public function create(Usuario $u);

    public function update(Usuario $u);

    public function delete(Usuario $u);
}
```

Implementação da interface

```
class UsuarioDAO implements CRUD {

    public function create(Usuario $u) {
        echo "A pessoa {$u->getNome()} foi cadastrado com sucesso!";
    }

    public function delete(Usuario $u) {

    }

    public function read(Usuario $u) {

    }

    public function update(Usuario $u) {

    }

}
```


Título

Interface Genérica para todas as Classes

- Criação da interface

```
interface CRUD {

    public function read($object);

    public function create($object);

    public function update($object);

    public function delete($object);
}
```

Implementação da interface

```
class NoticiasDAO implements CRUD {

    public function create($object) {
        echo "A noticia {$object->getTitulo()} foi cadastrado com sucesso!";
    }

    public function delete($object) {

    }

    public function read($object) {

    }

    public function update($object) {

    }

}
```

Título

Autoload

- Sempre que desejarmos criar um objeto, temos que ter obrigatoriamente a classe responsável pela criação do objeto carregada na memória, caso contrário o PHP retornará um erro.
- Para carregar classes na memória podemos carregar todas as classes no inicio do documento utilizando o comando *include* ou *include_once* ou com o mesmo comando imediatamente antes de instanciar um objeto.

Título

Autoload

- O grande problema é que o *include* carrega na memória as classes mesmo que você não use.
- Para resolver e simplificar tal procedimento o PHP disponibiliza a função de "carregamento automático", que é a função *autoload*, que carrega a classe apenas quando um objeto dessa classe for criada, tornando o processo de carregamento das classes para a memória dinâmico.

Título

Função autoload

1 – Autoload

```php
function __autoload($classe) {
    if (file_exists("class/" . $classe . ".php")) {
        include_once "class/{$classe}.php";
    } else if (file_exists("DAO/" . $classe . ".php")) {
        include_once "DAO/{$classe}.php";
    } else {
        echo "não possivel incluir a classe!{$classe}";
    }
}
```

2 - Autoload

```php
function __autoload($classe) {
    $Dir = ['class', 'DAO'];

    $dir = null;

    foreach ($Dir as $dirName) {
        if (file_exists($dirName . "/" . $classe . ".php")) {
            include_once $dirName . "/" . $classe . ".php";
            echo "<br>";
            echo "A classe foi incluida: " . $classe . ".php";
            $dir = TRUE;
        }
    }

    if (!$dir) {
        echo '<br>';
        echo "A classe não foi incluida: " . $classe . ".php";
    }
}
```


Título

Obrigado!

Marco Aurélio
marcosilvacosta@gmail.com