D S T Q Q S S

PHP + OO

• Classes, atributos e metodos

```
Class Pessoa {
    public $nome;
    public $idade;
    public function falar(){
        echo $this->nome . " de " . $this->idade . " anos";
    }
}
```

$nome; $idade; } atributos

public → podem ser acessados de qualquer lugar.

$this->nome → atributo DESSA classe

Obj com as caracs da classe pessoa

```
$pedro = new Pessoa();
$pedro->nome = "Pedro";
$pedro->idade = "18";
$pedro->falar();
```

$pedro->nome = "Pedro"; $pedro->idade = "18"; } setando os atribs.

falar(); → método

• Setters e Getters

private $nome
private $idade
↳ só podem ser acessados dentro da classe

continua

Digitalizado com CamScanner

Para resolver isso, fazemos duas func. pra cada atrib.
↳ pode ser acessada de fora da classe

```
public function setNome($nome){
    $this->nome = $nome
}
```
↳ seta o nome

```
public function getNome(){
    return $this->nome;
}
```
↳ devolve o nome.

Dessa forma, podemos acessar esses valores de fora da classe.

- Método construtor
  - Método mágico que é chamado assim que nós instanciamos uma classe.

```
public function __construct(*){
    echo "Obj instanciado";
}
```
↳ nome obrigatório
↳ aceita params

- Herança: Permite que outras classes herdem seus métodos.

```
Class Veiculo {
    public function andar(){};
}
```
↳ classe pai

Digitalizado com CamScanner

↳ diz que herda a classe

Class Carro extends veiculo(){}

Na pag:

$carro = new Carro();

$carro->andar();

↳ mesmo que Carro não tenha esse método, podemos utiliza-lo por carro extender Veiculo.

- Modificadores de acesso
  - Public Os mais permissivo de todos
  - Protected Pode ser acessado de dentro de uma classe e suas classes filhas
  - Private Só pode ser acessado de dentro da própria classe.

- Abstração

Permite a criação de classes e funções "vazias" que só serão definidas in suas "filhas"

```
abstract Class Banco(){
    abstract protected function sacar();
    abstract protected function depositar($v);
}
```

↳ só foi declarada não faz nada

```
Class Itau extends banco{
    public function depositar($v){
        ... Aqui sim é definido oq a func faz
```


Digitalizado com CamScanner

D S T Q Q S S

- Constante

```
Class Pessoa {
    const nome = "Pedro";

    public function exibirNome(){
        echo self::nome;
    }
}
```

↳ self para acessar pred da propria classe

```
$nome = Pessoa::nome;
```

↳ acessa a const.

- Metodos e atributos estáticos

```
Class Pessoa {
    public static function falar();
}

Pessoa::falar();
```

↳ Podemos acessar um metodo estatico sem instanciar a classe.

GENIBAT
Feira Nacional de Baterias de Chumbo + Conferência

Digitalizado com CamScanner

D S T Q Q S S

- Polimorfismo

Substituir ou reescrever metodos da classe pai

```
Class Animal {
    public function Andar(){
        echo "O animal andou";
    }
}

Class Cavalo extends Animal {
    public function Andar(){
        echo "O cavalo andou";
    }
}

$cavalo = new Cavalo();
$cavalo->andar();
```

↳ reescreve a func. da classe mãe

↳ saida: O cavalo andou

SENAI BAT Soluções Nacional e Internacional de Baterias de Chumbo + Conferência

Digitalizado com CamScanner

• Interfaces

Definem o modelo a ser usado por outra classe.

```
Interface Crud {  → contrato
    public function create();
    public function read();
      .     .     .
      .     .     .
}
```

↳ torna todos os métodos da interface obrigatórios aqui.

```
Class noticias implements Crud {
    public function create() {
        /// ⇝ a responsabilidade de criar a lógica
    }do mét. fica por conta da classe
    public function read() {
        ///
    }
```

FENIBAT 5ª Feira Nacional e Internacional de Baterias de Chumbo + Conferência

• Namespaces

Partições lógicas para o sist de arquivos das classes.

• na model:

```
namespace App\Model;
Class Carro {...
```

• na Ctrl

```
namespace App\Controller
Class Carro {...
```

na pag: $carroCtrl = new \App\Ctrl\Carro;

⇝ continua

Digitalizado com CamScanner

D S T Q Q S S

use App\Models\Carro as carroModel
use App\Ctrl\Carro as CarroCtrl
} apelidos.

$model = new CarroModel
$ctrl = new CarroCtrl
} basta chamar o apelido

• Exceptions

```
Class newsletter {
    public function cadastrarEmail($email) {
        if (self::validaEmail) {
            self::cadastraEmail();
        } else {
            throw new Exception("Email inv", 400)
        }
}
```

↳ msg de erro
↳ Codigo proprio

$newsletter = new newsletter();

```
try {
    $newsletter->cadastrarEmail($email);
} catch (Exception $exc) {
    echo "Msg" . $exc->getMessage();
    echo "Cod" . $exc->getCode();
    echo "Linha" . $exc->getLine();
    echo "Arquivo" . $exc->getFile();
}
```

FENIBAT 5ª Feira Nacional e Internacional de Baterias de Chumbo + Conferência

Digitalizado com CamScanner

9/1/22

D S T Q Q S S

• Invoke

Chamado quando tentamos utilizar o obj como função.

```
public function __invoke(){
    echo "Obj como função";
}
```

• Composer

. Gerenciador de dependencias php.

. necessita de download

. Tem o packagist como central de bibliotecas.

≠ Composer init

↳ Cria um composer.json para configuração

. a partir daqui basta configurar o arquivo.json de acordo com o nosso projeto

. necessita de um require no autoload para que todas as bibliotecas da vendor estejam disponiveis no projeto.

. Para uma conf. manual, basta escrever o json e rodar composer install.

Digitalizado com CamScanner