Os pilares da programação orientada a objeto

Bruno Henrique

Abstração de dados

- Nesta fase que identificamos nossa identidade e classificamos suas propriedades (características) e seus métodos (funções);
- É comum o uso de uma classe abstrata com propriedades comuns para utilização nas classes filhas;
- Uma classe que contenha pelo menos um método abstrato obrigatoriamente deverá ser abstrata também;
- Uma classe que herda de uma classe abstrata obrigatoriamente deve definir todos os métodos abstratos herdados, fornecendo a mesma visibilidade e a mesma quantidade de argumentos;

```php
abstract class ClasseAbstrata
{
    private $x = 2;
    // Médotos que devem ser definidos na classe filha
    abstract protected function metodoA();
    abstract protected function metodoB($argumentoA);

    // Método comum
    public function metodoC() {
        return $x;
    }
}

class ClasseConcretaA extends ClasseAbstrata
{
    protected function metodoA() {
        return $this->metodoC();
    }

    public function metodoB($argumentoA) {
        return $this->metodoC();
    }
}
```

Encapsulamento

- Encapsulamento é a forma de proteger as propriedades de um objeto;
- A única forma de manipular estes dados é utilizando métodos públicos do próprio objeto. Dessa forma evitamos que essas propriedades recebam valores indesejáveis;
- Para que o encapsulamento ocorra os métodos devem ser privados e os métodos públicos.

Tipos de visibilidade

- Private: atributos ou métodos declarados como private só podem ser utilizados pelo próprio objeto que os criou;
- Protected: atributos ou métodos declarados como protected podem ser utilizados pelo objeto que os criou ou por objetos herdados.
- Public: atributos ou métodos declarados como public podem ser utilizados livremente, pelo objeto que os criou, pelos objetos herdados ou pelo programa que utiliza o objeto. Atributos e métodos declarados sem visibilidade serão considerados como do tipo public.

```php
class ClasseA
{
    private $dia;
    public function setDia($dia) {
        if($dia > 0 && $dia <= 31){
            $this->dia = $dia;
            return true;
        } else {
            return false;
        }
    }
    public function getDia() {
        return 'O dia é: ' . $this->dia;
    }
}

$classeA = new ClasseA();
if($classeA->setDia(28))
    echo $classeA->getDia();
else
    echo 'Erro';
```

Herança

- Este método nos permite agrupar todas as propriedades comuns entre classes em uma classe pai ou também conhecida como superclasse;
- Utilização a abstração para criarmos a superclasse;
- Uma subclasse com herança de uma superclasse herda suas propriedades e seus métodos.

```php
class Carro
{
    private $nome, $ano, $modelo;
    public function setNome($nome){
        $this->nome = $nome;
    }
    public function setAno($ano){
        $this->ano = $ano;
    }
    public function setModelo($modelo){
        $this->modelo = $modelo;
    }
}

class Jaguar extends Carro
{
    private $aquevedor, $tetoSolar;
    public function ligaAquecedor(){
        //Codigo
    }
    public function abreTetoSolar(){
        //Codigo
    }
}
```

Polimorfismo

- A palavra polimorfismo vem do grego e significa muitas formas (poli: muitas, morphos: formas);
- Polimorfismo é reescrever um determinado método de uma superclass em uma subclasse;
- PHP não aceita sobrecarga em seus métodos.

```php
class Produto extends Banco
{
    private $nome, $preco, $peso;
    public function setNome($nome){
        $this->nome = $nome;
    }
    public function getPreco(){
        $sql = 'SELECT preco_brl FROM produtos';
        return mysql_query($sql, $this->connection);
    }
}

class ProdutoUSA extends Produto
{
    public function getPreco(){
        $sql = 'SELECT preco_usd FROM produtos';
        return mysql_query($sql, $this->connection);
    }
}
```

Referências

- Abstração de classes: http://php.net/manual/pt_BR/language.oop5.abstract.php
- Encapsulamento: http://www.kadunew.com/blog/php/encapsulamento-e-visibilidade
- Herança: http://bsideias.wordpress.com/2008/01/29/poo-programacao-orientada-a-objetos/
- Polimorfismo: http://www.tutsup.com/2014/08/11/heranca-e-polimorfismo-em-php-orientado-objetos/