PHP Class

kurz Webové technológie

Eduard Kuric

STU FIIT

Trieda

```
class MyRectangle {}


class MyRectangle
{
// atributy musia mat explicitne uvedenu uroven pristupu
  public $x, $y;
// metody nie, ak nie je uvedene, metody maju uroven
// pristupu public
  function newArea($a, $b) { return $a * $b; }
}
```

Prístup k členom z vnútra triedy

- vo vnútri triedy sa k jej členom pristupuje cez inštanciu triedy `$this` a operátor `->`

```
class MyRectangle {
      public $x, $y;
      function newArea($a, $b) {
            return $a * $b;
      }
      function getArea() {
            return $this->newArea($this->x,
                                  $this->y);
      }
}
```

Vytvorenie inštancie

```
$r = new MyRectangle();

// inštancia môže byť vytvorená predtým,
// ako je trieda definovaná
$r = new MyDummy();
class MyDummy {};
```

Prístup k členom z vonku

```
$r = new MyRectangle();

$r->x = 5;
$r->y = 10;
$r->getArea();
```

Inicializácia atribútov triedy

```
class MyRectangle
{
    // inicilizácia pri vytvorení inštancie,
    // hodnotou musí byť konštanta, nesmie byť
    // napr. premenná, alebo matematický výraz
    public $x = 5, $y = 10;
}
```

Konštruktor

- inicializácia inštancie/objektu, pri inicializácii atribútov sa nemusíme limitovať na konštanty

```
class MyRectangle
{
   public $x, $y;

   function __construct($x, $y) {
      $this->x = $x + 5;
      $this->y = $this->y + $this->x;
      echo "Constructed";
   }
}
```

Konštruktor /2

```
// pri vytváraní inštancie je zavolaný konštruktor
// vypise Constructed
$r = new MyRectangle();


// ak nema konstruktor argumenty,
// zátvorky môžeme vynechať
$r = new MyRectangle;
```

Deštruktor

- PHP garbage collecor uvoľní objekt, ak neexistuje na neho žiadna referencia
- pred uvoľnením je zavolaná metóda `__destruct`

```
class MyRectangle {
      function __destruct() {
            echo "Destructed";
      }
}

// odstrani referenciu na objekt,
// vypise sa Destructed
unset($r);
```

Názvy tried sú case-insensitive

- názvy premenných sú case-sensitive, ale **názvy tried** - rovnako ako názvy funkcií, kľúčových slov, built-in funkcií (`echo`) - **sú case-insensitive**

```
class MyClass {}
$o1 = new myclass(); // ok
$o2 = new MYCLASS(); // ok
```

Porovnanie objektov

- operátor ==, objekty sú rovnaké, ak majú **rovnaký typ** a ich **atribúty rovnaké hodnoty**
- operátor ===, ak premenné odkazujú na tú istú inštanciu

```
class Flag {
      public $flag = true;
}

$a = new Flag();
$b = new Flag();
$c = ($a == $b); // true
$d = ($a === $b); // false
```

Anonymné triedy

- podpora od PHP 7

```
$obj = new class {};

$o = new class('Hi') {
     public $x;
     public function __construct($a) {
          $this->x = $a;
     }
};
echo $o->x; // Hi;
```

Closure

- anonymné funkcie – sú funkcie bez názvu
- closure – je anonymná funkcia, ale reprezentovaná objektom
- umožňujú `lazy loading` – inicializované, až keď sú použité
- toto nám umožňuje pristupovať k atribútom triedy, ktoré sú `private`
- môžeme tiež pridať nové správanie do triedy bez toho, aby sme menili jej definíciu

Closure /2

```
class SimpleClass {
    private $privateData = 2;
}

$simpleClosure = function() {
    return $this->privateData;
};

$resultClosure =
   Closure::bind($simpleClosure,
    new SimpleClass(), 'SimpleClass');

echo $resultClosure();
```