Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Metodika pre konvencie kódu

Tím DILEMA

Členovia tímu: Bc. Zuzana Bachárová
Bc. Peter Bakoš
Bc. Martin Baláž
Bc. Marek Bernád
Bc. Imrich Nagy
Bc. Peter Tibenský
Bc. Juraj Volko
Vedúci projektu: Ing. Ján Lang, PhD.
Tím: č. 10 [FIIT DU]
Akademický rok: 2017/2018
Dátum poslednej zmeny: 26.2.2018
Vypracoval: Martin Baláž

1 Úvod

Účelom tejto metodiky je opísať spôsob, akým písať dobre čitateľný kód. Uvedenou metodikou sa riadia všetci členovia tímu.

2 Postup

Laravel sám o sebe nevyžaduje striktnú konvenciu písania kódu, ale odporúča dodržiavať PSR štandardy.

2.1 Konvencie (Cheat Sheet)

2.1.1 Pomenovanie

Premenné a funkcie: používať **lowerCamelCase**

```
// Správne
private function doSomething( $userPrefs, $editSummary )

// Nesprávne
private function DoSomething( $UserPrefs, $EditSummary )
```

$wg	globálna premenná
$dbw	databázová premenná na zapisovanie (master connection)
$dbr	databázová premenná na čítanie (non-concurrency-sensitive)
$wc	Cookie premenná, napr. $_COOKIE['wcCookieName']
$ws	Session premenná, napr. $_SESSION['wsSessionName']
$wp	Post premenná, napr. $wgRequest->getText('wpLoginName')

2.1.2 Medzery

```
// 1. Správne
$a = $b + $c;

// 1. Nesprávne
$a=$b+$c;

// 2. Spávne
$a = [ 'foo', 'bar' ];
$c = $a[0];
$x = [];
```

```php
// 2. Nesprávne
$a = ['foo', 'bar'];
$c = a[ 0 ];
$x = [ ];

// Medzera za if
// 3. Správne
if ( isFoo() ) {
    $a = 'foo';
}

// 3. Nesprávne
if( isFoo() ) {
    $a = 'foo';
}

// 4. Správne
(int)$foo;

// 4. Nesprávne
(int) $bar;
( int )$bar;
( int ) $bar;
```

2.2 PSR-1

1. Súbory **MUSIA** používať iba <?php a <?= tagy
2. Súbory **MUSIA** používať **UTF-8** pre PHP kód
3. Namespace a názvy tried **MUSIA** dodržovať „autoloading“ **PSR**
4. Konštanty tried (const) **MUSIA** byť deklarované veľkými písmenami s použitím podtržníkov na oddelene slov
5. Názvy metód **MUSIA** byť deklarované následovne: **camelCase()**

```php
// #3
<?php
namespace Vendor\Model;

class Foo
{
}
```

```php
// #4
<?php
namespace Vendor\Model;

class Foo
{
    const VERSION = '1.0';
    const DATE_APPROVED = '2012-06-01';
}
```

2.3 PSR-2

1. Kód **MUSÍ** používať **4** medzery pre odsek, nie tabulátor
2. Kód **MUSÍ** obsahovať prázdny riadok po Namespace a use
3. Otváracie zložené zátvorky tried, metód a kontrolných jednotiek (napr. if, else), **MUSIA** ísť na nový riadok a zatváracie zložené zátvorky musia ísť na nový riadok po tele triedy alebo metódy
4. Štrukturálne kontrolné jednotky **MUSIA** mať jednu medzeru pred zátvorkou; metódy a funkcie **NIE**

```php
<?php
namespace Vendor\Package;
                                                    // #2
use FooInterface;
use BarClass as Bar;
use OtherVendor\OtherPackage\BazClass;
                                                    // #2
class Foo extends Bar implements FooInterface
{                                                   // #3
  public function sampleMethod($a, $b = null)       // #4
  {
    if ($a === $b) {                                // #3
      bar();                                        // #4
    } elseif ($a > $b) {
      $foo->bar($arg1);
    } else {
      BazClass::bar($arg2, $arg3);
    }                                               // #3
  }                                                 // #3
```

```
    final public static function bar()                  // #4
    {                                                   // #3
        // method body
    }                                                   // #3
}
```

2.4 Komentáre

Pre správne využitie **phpDocumentator** je potrebné dodržiavať správne konvencie písania komentárov.

Jedno riadkový DocComment vyzerá nasledovne:

```
/** This is a single line DocComment. */
```

Viac riadkový DocComment vyzerá nasledovne:

```
/**
 * This is a multi-line DocComment.
 */
```

2.4.1 Zhrnutie

Zhrnutie je krátky ale efektívny opis prvku, porovnateľné so sloganom alebo nadpisom. Nemôže obsahovať špeciálne znaky.

2.4.2 Popis

Popisom môže byť dlhý text s podrobným vysvetlením, čo daný prvok robí. Opis je voliteľný, pretože existuje veľa prvkov, ktoré sú tak jednoduché, že nepotrebujú vysvetlenie dĺžky.

Príklad použitia:

- oddelenie novým riadkom;

```
/**
 * This is a summary
```

```
 *
 * This is a description
 */
```

- oddelenie bodkou.

```
/**
 * This is a summary.
 * This is a description
 */
```

2.4.3 DocBlock

DocBlock vždy začína s /** a je ukončený */. DockBlock obsahuje viac elementov ako je Zhrnutie, Opis, vstupné avystupné parametre.

```
<?php
/**
 * This is a DocBlock.
 */
function associatedFunction()
{
}
```

Treba si dať pozor na viacúrovňové vytváranie DockBlock-u. Kde prvý DockBlock patrí k dokumentu a druhý ku triede a ak je použitý iba jeden tak patrí triede.

Príklad viacúrovňového DockBlock:

```
<?php
/**
 * I belong to a file
 */

/**
 * I belong to a class
 */
class Def
```

```
{
}

<?php
/**
 * I belong to a class
 */

class Def
{
}
```

Príklad úplného DockBlock:

```
<?php
 /**
  * A summary informing the user what the associated element does.
  *
   * A *description*, that can span multiple lines, to go _in-depth_ into the details of this element
  * and to provide some background information or textual references.
  *
  * @param string $myArgument With a *description* of this argument, these may also
  *    span multiple lines.
  *
  * @return void
  */
  function myFunction($myArgument)
  {
  }
```