Developers Day

Объектная гимнастика в PHP

Денис Потапов

DenysPotapov.com

Презентация

bit.ly/php-webcamp

Код

bit.ly/php-webcamp-git

Объектная гимнастика

англ. Object Calisthenics. Calisthenics — зарядка, гимнастика.

- читаемость
- тестируемость
- понятность
- поддерживаемость

Важно

Объектная гимнастика — это упражнения, я не правила

#1

Только один уровень отступа в методе

- цикломатическая сложность
- один уровень абстракции

#2

Не используйте else

- читаемость
- предсказуемость

синтетический пример

```
public function createPost($entity){
        $form = new MyForm($entity);
        if ($form->isValid()) {
            if (!$this->posts->exists($entity)) {
                $this->posts->save($entity);
                return $this->redirect('create_ok');
            } else {
                $error = "Post Title already exists";
                return array(form => $form, error => $error);
            }
        } else {
            $error = "Invalid fields";
            return array(form => $form, error => $error);
        }
    }
```

синтетический пример

```
public function createPost($entity){
        $form = new MyForm($entity);
        if ($form->isValid()) {
            $error = "Invalid fields";
            return array(form => $form, error => $error);
        }
        if ($this->posts->exists($entity)) {
            $error = "Post Title already exists";
            return array(form => $form, error => $error);
        }

        $this->posts->save($entity);
        return $this->redirect('create_ok');
    }
```

пример из Monolog

```php
public function __construct($serverData = null)
    {
        if (null === $serverData) {
            $this->serverData =& $_SERVER;
        } elseif (is_array($serverData)) {
            $this->serverData = $serverData;
        } else {
            throw new \UnexpectedValueException('$serverData ....');
        }
    }
```

пример из Monolog

```
public function __construct($serverData = null)
    {
        if (null === $serverData) {
            $serverData =& $_SERVER;
        }
        if (!is_array($serverData)) {
            throw new \UnexpectedValueException('$serverData ....');
        }
        $this->serverData =& $serverData;
    }
```

#3

Оберните все примитивные типы

- предсказуемость
- инкапсуляция операций
- если у объекта есть поведение[адаптировано для PHP]

синтетический пример

```
//...
$component->repaint(false);
//...

class UIComponent{
    //...
    public function repaint($animate = true){
        //...
    }
}
```

синтетический пример

```
//...
$component->repaint(new Animate(false));
//...

class UIComponent{
    public function repaint(Animate $animate){
        //...
    }
}

class Animate{
    public $animate;

    public function __construct($animate = true){
        $this->animate = $animate;
    }
}
```

#4

Коллекции первого класса

- инкапсуляция поведений, относящихся к коллекции:
  - поиск
  - фильтрация

#5

Одна точка на строку* **

* Два -> на строку в PHP

** кроме текучих интерфейсов

- тестируемость (mock-объекты)
- закон Деметры

#6

Не используйте сокращения

- слишком длинное название метода?
- слишком часто приходиться вызывать метод?
- читаемость

#7

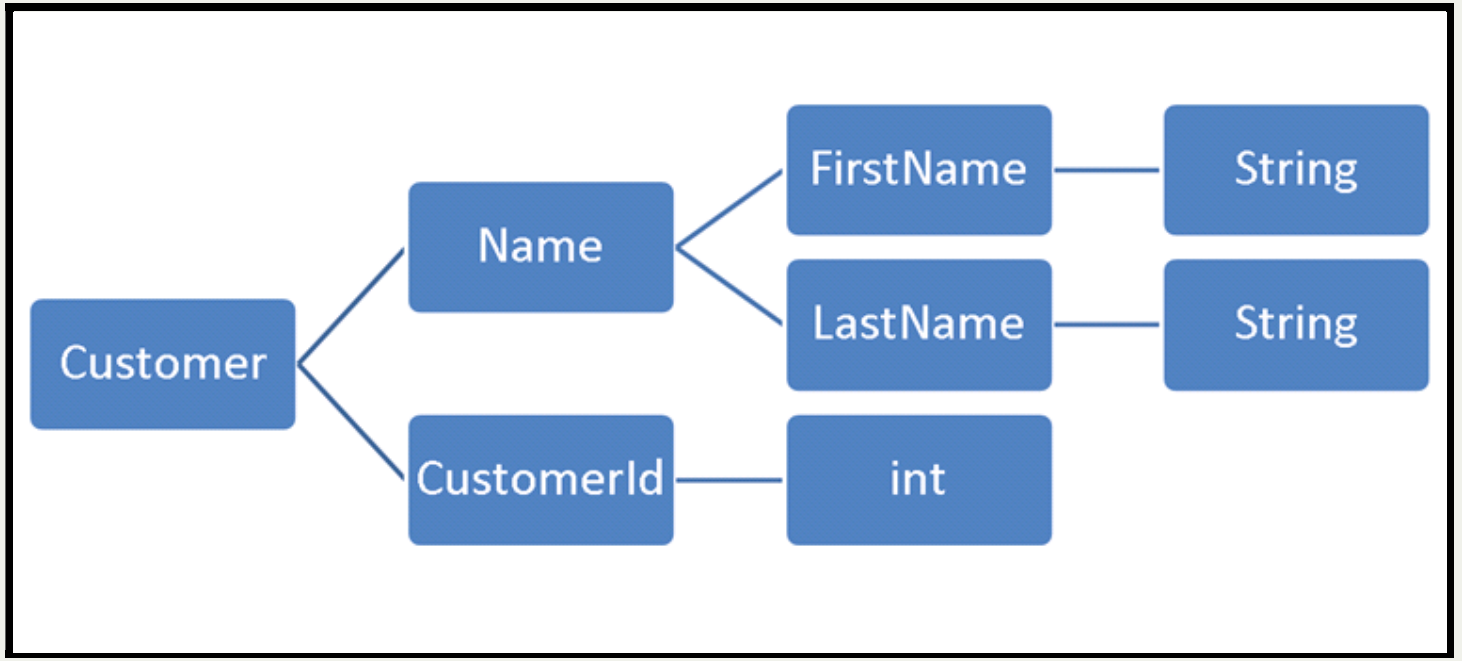
Сохраняйте сущности короткими

- 200 сток в файле (включая docblock)
- 10 методов в классе
- 15 классов в пространстве имен

#8

Никаких классов с более чем 2 атрибутами

- это не шутка, а упражнение
- высокая связность
- инкапсуляция


Customer
Name
FirstName
String
LastName
String
CustomerId
int

Почему два?

- обслуживают состояние одного атрибута
- координируют две отдельные переменные

#9

~~геттеры, сеттеры и свойста~~

- указывай, а не спрашивай
- принцип открытости/закрытости

синтетический пример

```
class Game
{
    private $score;

    public function setScore($score) {
        $this->score = $score;
    }

    public function getScore() {
        return $this->score;
    }
}

//..
$game->setScore($game->getScore() + ENEMY_DESTROYED_SCORE);
```

синтетический пример

```
class Game
{
    private $score;

    public function addScore($delta) {
        $this->score += $delta;
    }
}

//..
$game->addScore(ENEMY_DESTROYED_SCORE);
```

Еще раз

1. Только один уровень отступа в методе
2. Не используйте else
3. Оберните все примитивные типы
4. Коллекции первого класса
5. Одна точка на строку
6. Не используйте сокращения
7. Сохраняйте сущности короткими
8. Никаких классов с более чем 2 атрибутами
9. Никаких геттеров, сеттеров и свойств

Если вам не нравятся эти правила — это нормально

Вопросы?