
<?php
echo 'Hello world';

Выжить с ООП

Макс Гопей


CGI

Почему?


24 HR
CAR

Объектно-

Ориентированное

Программирование

Почему ООП?

- Это удобно
- Это расширяемо
- Адекватная модель мира
- Позволяет строить асбтракции
- ~~*Мне так сказали*~~

Объект олицетворяет объект

в реальном мире.

Объект олицетворяет объект

в ~~реальном мире~~ вашем домене.

Класс — это описание объекта

Класс — это не область имен

Пожалуйста, не делайте так:

```
01. use Spatie\Regex\Regex;
02.
03. // Using `match`
04. Regex::match('/a/', 'abc'); // `MatchResult` object
05. Regex::match('/a/', 'abc')->hasMatch(); // true
06. Regex::match('/a/', 'abc')->result(); // 'a'
```

Вместо:

```
01. Regex::match('/a/', 'abc'); // `MatchResult` object
02. Regex::match('/a/', 'abc')->hasMatch(); // true
03. Regex::match('/a/', 'abc')->result(); // 'a'
```

лучше:

```
01. $pattern = new Regex('/a/');
02. $matchingResult = $pattern->match('abc'); // `MatchResult` object
03. $matchingResult->hasMatch(); // true
04. $matchingResult->result(); // 'a'
```

Вместо:

```
01. Assertion::nullOrMax(null, 42); // success
02. Assertion::nullOrMax(1, 42);    // success
03. Assertion::nullOrMax(1337, 42); // exception
```

лучше:

```
01. // since PHP 5.6
02. Assertion\nullOrMax(null, 42); // success
03. Assertion\nullOrMax(1, 42);    // success
04. Assertion\nullOrMax(1337, 42); // exception
```

Инкапсуляция

Реализация меняется

- разные алгоритмы
- разные хранилища
- разные протоколы

Абстракция не меняется


PROMISE

Наследование

Что у нас есть?

Наследование!

Для чего?

Писать абстракции!

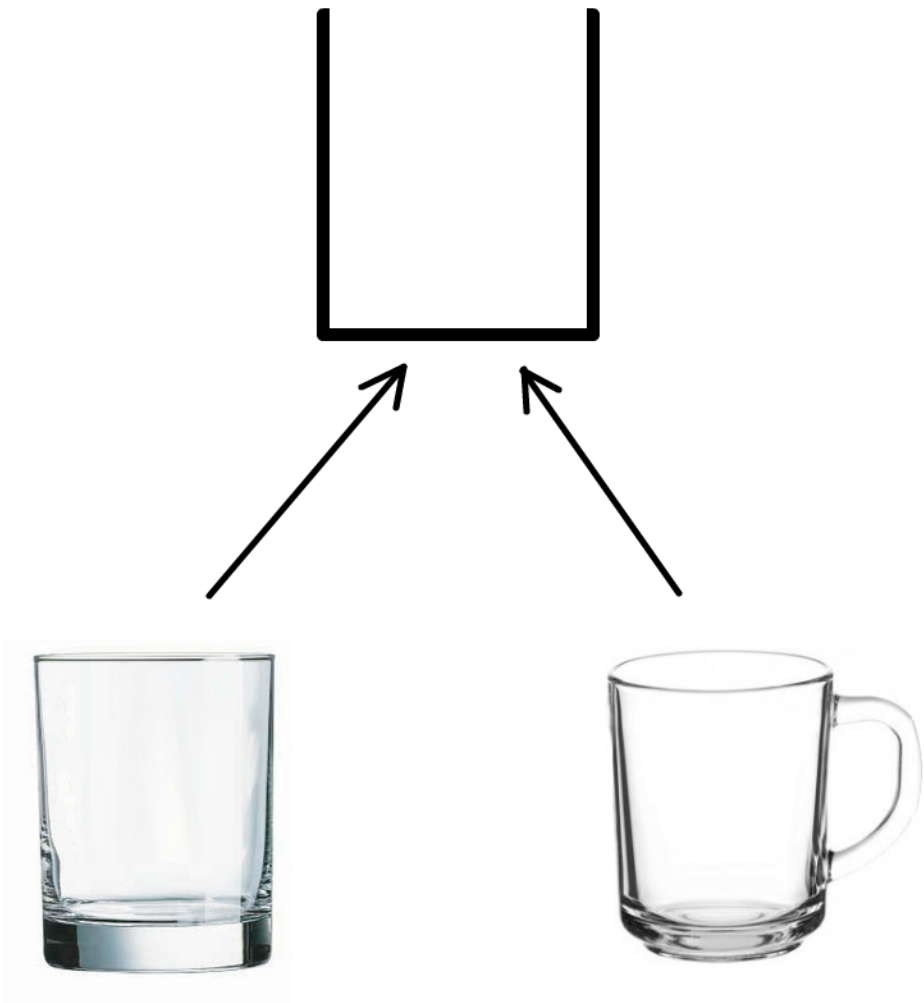
А мы что делаем?

Меняем существующие классы!

Стакан

Чашка

Динозаврик с ручкой



Проектируйте для наследования.

Наследуйте, если спроектировали для этого.

Полиморфизм

Полиморфизм

Один интерфейс — множество реализаций.

И не важно, какая используется сейчас.

```php
interface Container {
    public function drop();
}
class Glass implements Container {
    public function drop() { /* well, crash */ }
}
class Cup implements Container {
    public function drop() { /* well, crash, and throw the handle out */  }
}
class Cat {
    public function dropContainer(Container $container) {
        $container->drop();
    }
}
```

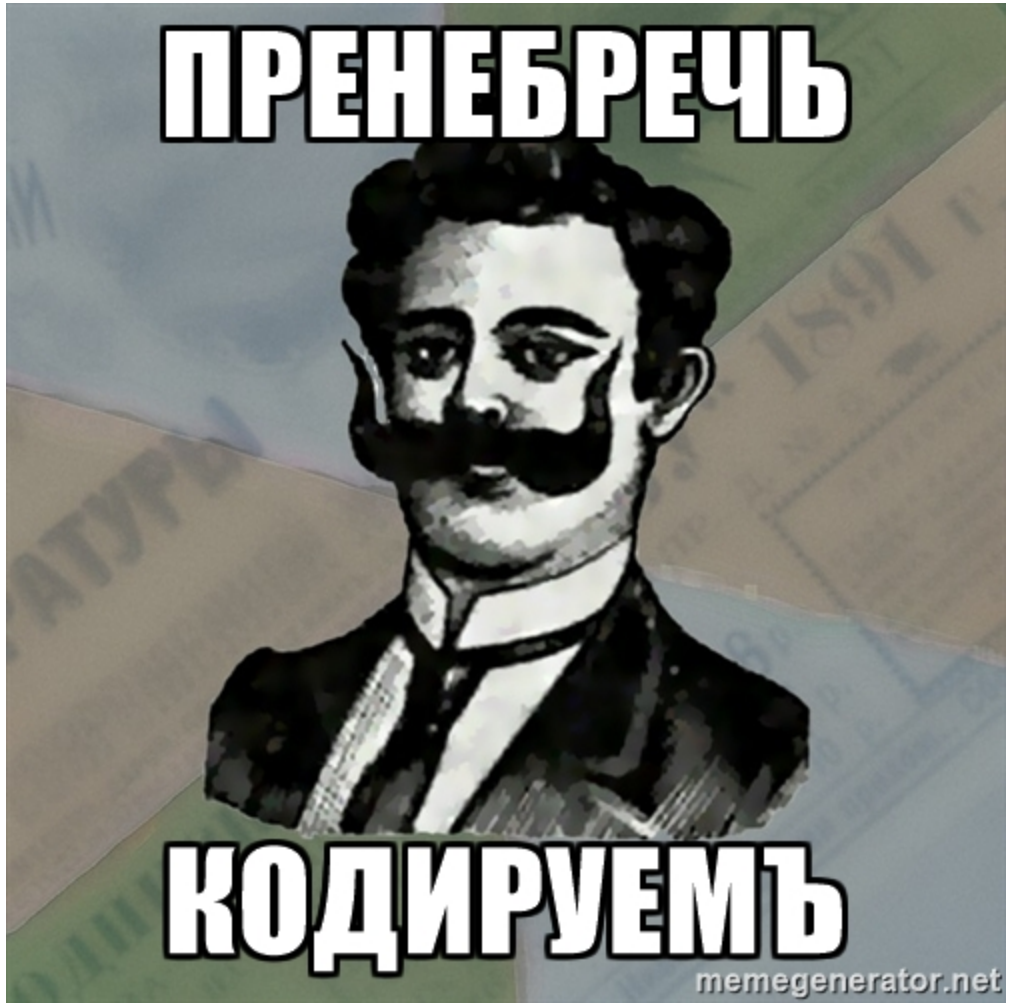
```php
$cat = new Cat();
$cat->dropContainer(new Glass());
$cat->dropContainer(new Cup());
$cat->runAway();
```

```php
class ContainerCollection implements Iterator {
    public function current() : Container { /* ... */ };
    // ...
}

$containersOnTable = new ContainerCollection();

// Your mom fills the collection here:
$eventManager->dispatch('serve_table', $containers);

array_walk($containersOnTable,
    function(Container $container) use ($cat) {
        $cat->dropContainer($container);
    }
);
```

Инкапсуляция помогает скрыть реализацию за **абстракицей**.

Наследование помогает строить **абстракции**.

Полиморфизм помогает писать код на основе **абстракций**.


ПРЕНЕБРЕЧЬ
КОДИРУЕМЪ
memegenerator.net

Методы классов

```php
class SearchEngine {
    public function indexProduct(Product $product) {
        $this->productIndex->add($product);
        $indexIsUpToDate = $this->productIndex->isUpToDate();
        //..
    }
}
```

Три вида сообщений:

- Команда
- Запрос
- Документ

Метод-команда

- принимает запрос на **изменение состояния объекта**,
- ничего не возвращает (void),
- выполняется успешно, либо бросает исключение.

Метод-запрос

- принимает запрос на получение информации,
- возвращает значение указанного типа,
- если это невозможно, возвращает NULL*,
- * или бросает исключение,
- **никогда не меняет наблюдаемое состояние объекта**.

Документ

- результат работы метода-запроса.

Принцип:
Command Query Separation (CQS)

Избегайте сеттеров

```php
class Person {
    private $firstName, $lastName, $gender, $email;
    // __constructor()
    // getters
    // setters
}

$person = new Person('Sheldon', 'Cooper', 'M', 'shelly@gmail.com');
render($person);
```

```php
class Person {
    private $firstName, $lastName, $gender, $email;
    // __constructor()
    // getters
    // setters
}

$person = new Person('Sheldon', 'Cooper', 'M', 'shelly@gmail.com');

$person->setFirstName('Penny');
$person->setGender('F');

render($person);
```

Отдавайте предпочтение неизменяемым объектам (immutables)

```php
class Person {
    private $firstName, $lastName, $email, $gender;

    public function rename(NameChangingRequest $request) {
        // change first/last/... names depending on request
        // throw exception if name is not male, for instance
    }

    public function changeGender(GenderChangingRequest $request) {
        // A request which contains also the new name,
        // maybe the reason or whatever is needed.
    }
}

$person->changeGender(new GenderChangingRequest('M', 'New Name'));
```

Метод — это транзакция

```
01. class Product {
02.     public function reduceQuantity($deltaQuantity) {
03.         $this->quantity -= $deltaQuantity;
04.     }
05.     public function verifyStockAvailability() {
06.         if ($this->quantity == 0) {
07.             $this->removeFromStock();
08.         }
09.     }
10. }
11.
12. $product->reduceQuantity($orderedQuantity);
13. $product->verifyStockAvailability();
```

```php
class Product {
    private function reduceQuantity($deltaQuantity) { /*...*/ }
    private function verifyStockAvailability() { /*...*/ }

    public function takeFromStock($quantity) {
        try {
            $this->reduceQuantity($orderedQuantity);
            $this->verifyStockAvailability();
        } catch() {
            // ...
        }
    }
}

$product->takeFromStock($orderedQuantity);
```

Название — это интерфейс

```
01. function getProductUrl($product) {
02.     return '/'
03.         . str_replace(' ', '-', strtolower($product));
04. }
```

```php
$product = 'Ski Boots'; // product name
$product = 456; // product id
$product = [ // product data
    'id' => 456,
    'name' => 'Ski Boots'
];
```

Name things by their real value

- **$product** — object
- **$productName** — string
- **$productId** — integer / hash
- **$productData** — array / structure

Объекты-значения (value-objects)

```
01. $person->addContactInformation(
02.     new EmailAddress('max.gopey@gmail.com')
03. );
04. $person->addContactInformation(
05.     new LinkedInProfileUrl('@max.gopey')
06. );
07. $this->redirect(new Url('https://stackoverflow.com'));
```

Объекты-значения не изменяются.

Немного практики

Интернет-аптека для ветеринаров.

Можно покупать товар:

- для клиники (clinic),
- для клиента (pet owner).

От этого зависит процесс заказа. Например, при заказе для клиента можно оформить доставку в клинику или на дом.

```
01. class Cgi_Nda_Model_Order_Mode
02. {
03.     const ORDER_MODE_CLINIC = 1;
04.     const ORDER_MODE_PET_OWNER = 2;
05. }
```

```
01. class Cgi_Nda_Model_Session
02. {
03.     public function getOrderMode() : int {
04.         return $this->getSessionValue('order_mode');
05.     }
06. }
```

```
01.  class Cgi_Nda_Block_Order_Mode_Info
02.  {
03.      public function getOrderMode() {
04.          $orderMode = $this->_getSession()->getOrderMode();
05.          if ($orderMode) {
06.              if ($orderMode == Cgi_Nda_Model_Order_Mode::ORDER_MODE_PET_OWNER) {
07.                  return 'For pet owner';
08.              } elseif ($orderMode == Cgi_Nda_Model_Order_Mode::ORDER_MODE_CLINIC) {
09.                  return 'For clinic';
10.              } else {
11.                  return false;
12.              }
13.          } else {
14.              return false;
15.          }
16.      }
17.
18.      public function isSeparateShippingAddressAllowed ()
19.      {
20.          $orderMode = $this->_getSession()->getOrderMode();
21.          return $orderMode &&
22.              $orderMode == Cgi_Nda_Model_Order_Mode::ORDER_MODE_PET_OWNER
23.      }
24.  }
```

```
01. interface Cgi_Nda_Model_Order_Mode_Interface
02. {
03.     public function getCode() : int;
04.
05.     public function getTitle() : string;
06.
07.     public function isSeparateShippingAddressAllowed() : bool;
08. }
```

```
01.  class Cgi_Nda_Model_Order_Mode_Clinicimplements Cgi_Nda_Model_Order_Mode_Interface
02.  {
03.      public function getCode() : int {
04.          return 1;
05.      }
06.      public function getTitle() : string {
07.          return 'For Clinic';
08.      }
09.      public function isSeparateShippingAddressAllowed() : bool {
10.          return false;
11.      }
12.  }
13.
14.  class Cgi_Nda_Model_Order_Mode_PetOwnerimplements Cgi_Nda_Model_Order_Mode_Interface
15.  {
16.      public function getCode() : int {
17.          return 2;
18.      }
19.      public function getTitle() : string {
20.          return 'For Pet Owner';
21.      }
22.      public function isSeparateShippingAddressAllowed() : bool {
23.          return true;
24.      }
25.  }
```

```
01. class Cgi_Nda_Model_Session
02. {
03.     public function getOrderMode()
04.         : Cgi_Nda_Model_Order_Mode_Interface {
05.         return $this->getSessionValue('order_mode');
06.     }
07. }
```

```php
class Cgi_Nda_Block_Order_Mode_Info
{
    private $orderMode;

    public function __construct(
        Cgi_Nda_Model_Order_Mode_Interface $orderMode
    ) {
        $this->orderMode = $orderMode;
    }

    public function getOrderModeTitle() {
        return $this->orderMode->getTitle();
    }

    public function isSeparateShippingAddressAllowed() {
        return $this->orderMode->isSeparateShippingAddressAllowed();
    }
}
```


That's all Folks!
SUPERB wallpapers