

Аспектно Ориентированное Программирование в RНР

Щеваев “rachanga” Павел (rachanga.shevaev@gmail.com)
BIT, г.Пенза



Как жаль, что мы не живем в
идеальном мире!



Любовь и гармония...

$$\text{Microsoft}^{\text{®}} + \text{Penguin} =$$



Бесплатное
ПИВО...





Поддержка namespaces в PHP...

```
<?php
namespace Foo {
    class Bar {
        ...
    }
}
?>
```





...n closures

```
<?php  
$mult = 2;  
$arr = array_map(function($i){return $mult*$i},  
                 array(1, 2, 3));  
?>
```





Простая и понятная бизнес ЛОГИКА

```
class NewsController extends Controller {  
  function create(){  
    $news = new News();  
    $news->setDate($this->request->get('date'));  
    $news->setContent($this->request->get('content'));  
    $news->save();  
  }  
}
```



Что является помехой?

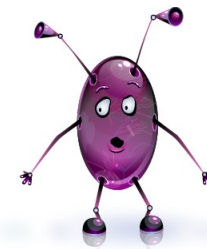
Crosscutting Concerns
(Сквозной

функционал)

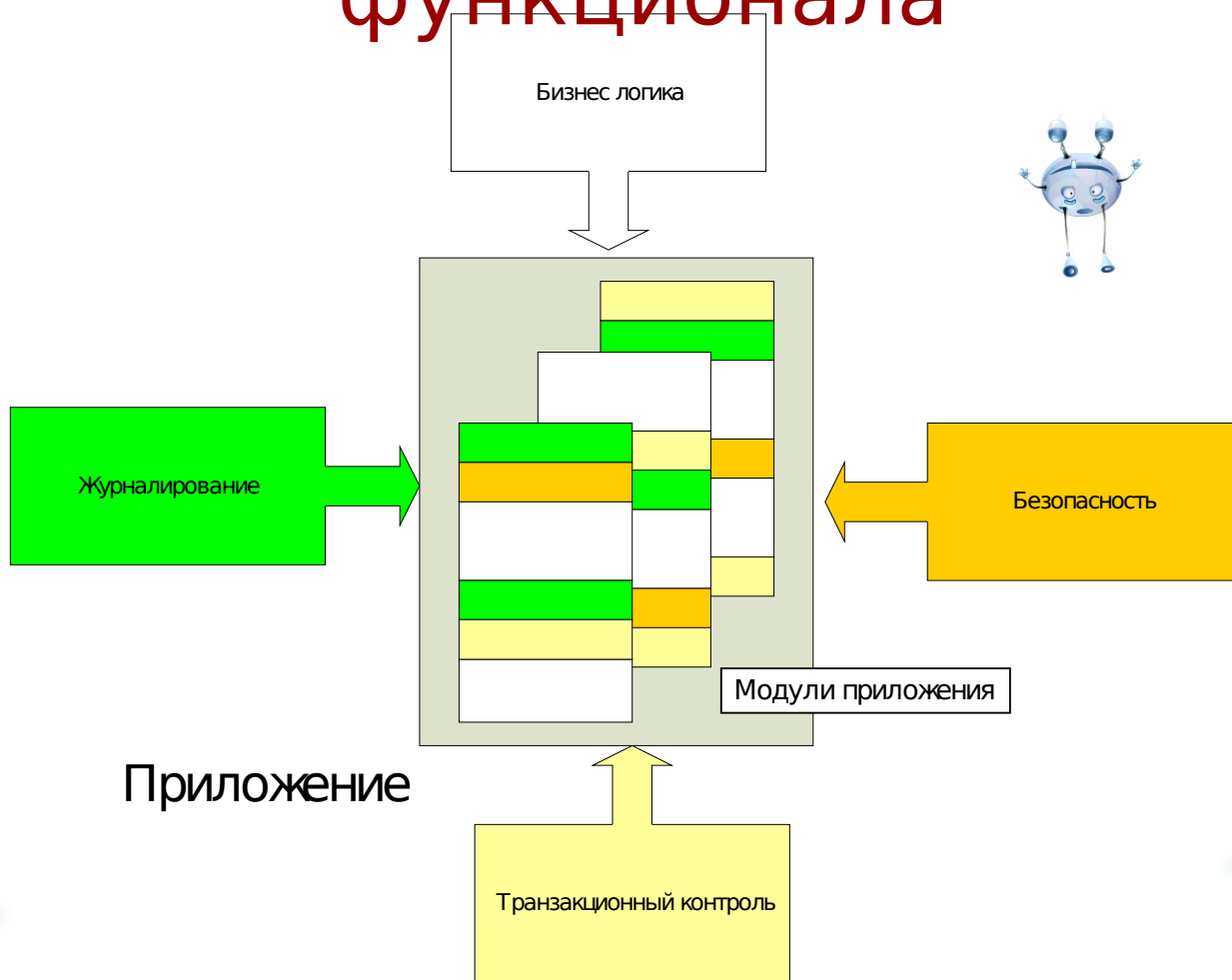




```
class NewsController extends Controller {
function create(){
    $ctx = ApplicationContext :: instance();
    if($ctx->isUserAuthorized())
    {
        $news = new News();
        $news->setDate($this->request->get('date'));
        $news->setContent($this->request->get('content'));
        $ctx->startTransaction();
        try {
            $news->save();
            $ctx->commitTransaction();
            $ctx->log("News created successfully");
        }
        catch(ValidationException $e) {
            $ctx->rollbackTransaction();
            $ctx->log("News validation error");
            throw $e;
        }
    }
    else
    {
        throw new AuthException();
        $ctx->log("Operation is not permitted");
    }
}
}
```



Лоскутное одеяло сквозного функционала



АОП спешит на помощь



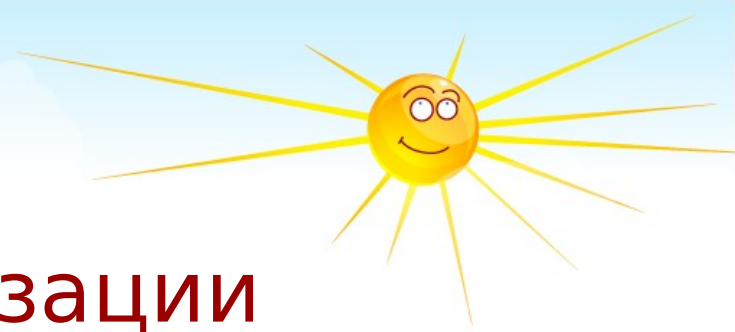


Простая и понятная бизнес логика - 2.0

```
class NewsController extends Controller {  
    function create(){  
        $news = new News();  
        $news->setDate($this->request->get('date'));  
        $news->setContent($this->request->get('content'));  
        $news->save();  
    }  
}
```



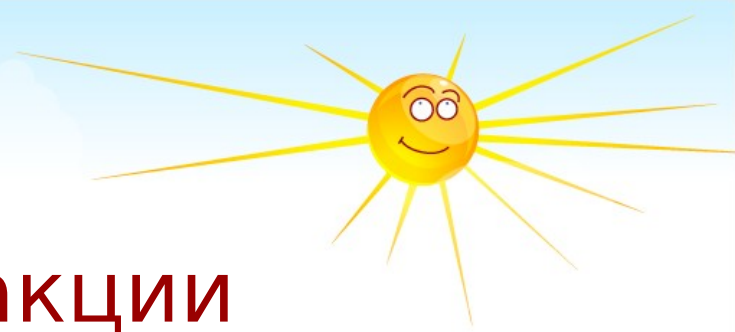
Аспект авторизации



```
aspect Authentication{
    pointcut controllerCreate:exec(public *Controller::create());
    around(): controllerCreate{
        $ctx = ApplicationContext :: instance();
        if($ctx->isUserAuthorized())) {
            proceed();
        } else {
            $ctx->log("Operation is not permitted");
            throw new AuthException();
        }
    }
}
```



Аспект транзакции



```
aspect Transaction{
    pointcut save:call(public News->save());
    around(): save{
        $ctx = ApplicationContext :: instance();
        $ctx->startTransaction();
        try {
            proceed();
        } catch (ValidationException $e) {
            $ctx->rollbackTransaction();
            $ctx->log("News validation error");
            throw $e;
        }
    }
}
```



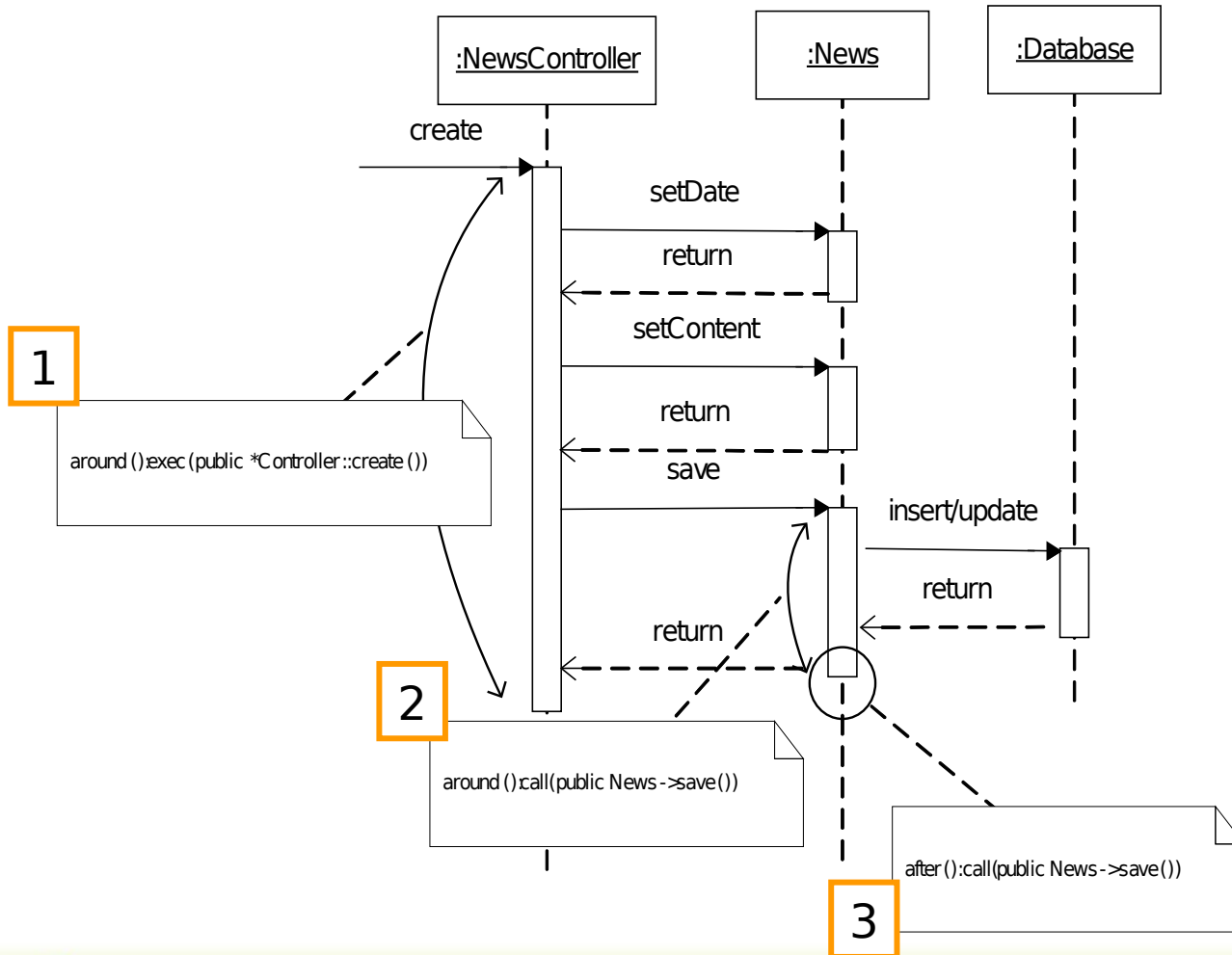


Аспект журналирования

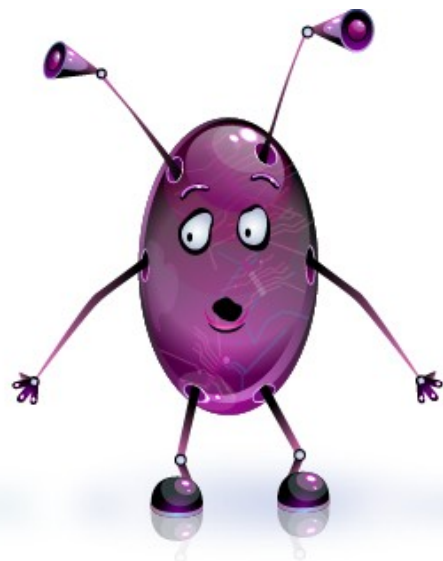
```
aspect Logging{  
    pointcut save:call(public News->save());  
    after(): save{  
        $ctx = ApplicationContext :: instance();  
        $ctx->log("News created successfully");  
    }  
}
```



Диаграмма последовательности



Непонятно? Немного теории (совсем чуть-чуть, честно!)



Введение в АОП

- Gregor Kiczales + команда XEROX PARC + желание облегчить нам жизнь = AspectJ
- AspectJ(<http://aspectj.org>) - “lingua franca” в мире АОП



Базовые понятия

- JoinPoint
- PointCut
- Advice
- Introduction
- Aspect
- Weaving



JoinPoint

- **JoinPoint** - фундаментальное понятие АОП, под которым понимают любую четко идентифицируемую точку исполнения программы
- **JoinPoint** точки являются кандидатами возможной инъекции сквозного функционала

JoinPoint

- Возможные **JoinPoint** точки в примере:

- выполнение метода `credit()`

- доступ к атрибуту `balance`

```
class Account {  
    function credit($amount) {  
        $this->balance += $amount;  
    }  
}
```

1

2

PointCut

- **PointCut** – набор(срез) **JoinPoint** точек, удовлетворяющих определенному условию.
- **PointCut** бывают именованные и анонимные.
- **PointCut** – это некое подобие SQL запроса для **JoinPoint** точек

```
| exec(Account::credit(*))
```

```
    исполнение метода Account :: credit()
```

Advice

- **Advice** - код, выполняемый для каждой **JoinPoint** точки, входящей в определённый срез **PointCut**.
- **Advice** может выполняться до (before), после (after) или вместо (around) **JoinPoint** точки.
- **Advice** схож с традиционным ООП методом

Скука зеленая...



Advice

- Пример Advice метода для анонимного PointCut среза

```
before(): exec(Account::credit(*)) {  
    echo("Сейчас будет выполнен метод credit");  
}
```

- Пример Advice метода для именованного PointCut среза

```
pointcut credit : exec(Account::credit(*))  
before(): credit {  
    echo("Сейчас будет выполнен метод credit");  
}
```

Introduction

- **Introduction** – инструкция для изменения статической структуры классов, интерфейсов и аспектов.
 - Попробуем добавить метод `setLog()` во все классы, которые начинаются со строки “Foo”

```
public function Foo*::setLog(Log $log){  
    $log->setLevel(Log::ALL);  
    $this->log = $log;  
}
```

Aspect

- **Aspect** - модуль в терминах АОП, некоторый аналог класса, который инкапсулирует в себе правила применения сквозного функционала.
- **Aspect** - конечный контейнер для всех АОП элементов: PointCut, Advice Introduction.
- **Aspect** схож с ООП классом (также позволяет объявлять и использовать обычные методы и атрибуты)



Аспект

- Пример аспекта(объединяем все вместе)

```
aspect ExampleAspect {  
    before(): exec(Account::credit(*)) {  
        $this->say("Сейчас будет выполнен метод credit");  
    }  
  
    function say($msg) {  
        echo($msg);  
    }  
}
```

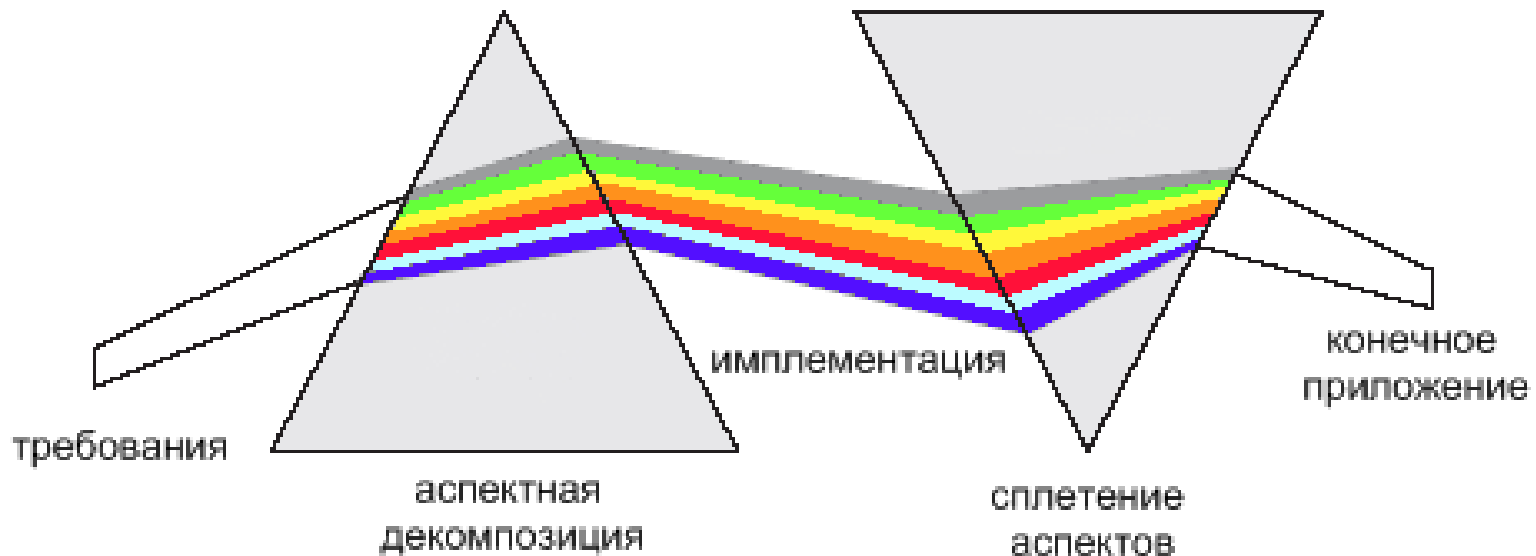


Weaving

- **Weaving** – процесс «вплетения» аспектов в логику приложения.
- **Weaving** процесс может происходить на уровне исходных кодов или же на уровне виртуальной машины (в случае РНР, это уровень исполнения opcode инструкций).

АОП - вид сверху (или сбоку?)

- Процесс разбиения функциональных требований на аспекты с их последующим сплетением в конечный код приложения



Фух...с теорией покончено



Средства АОП для PHP

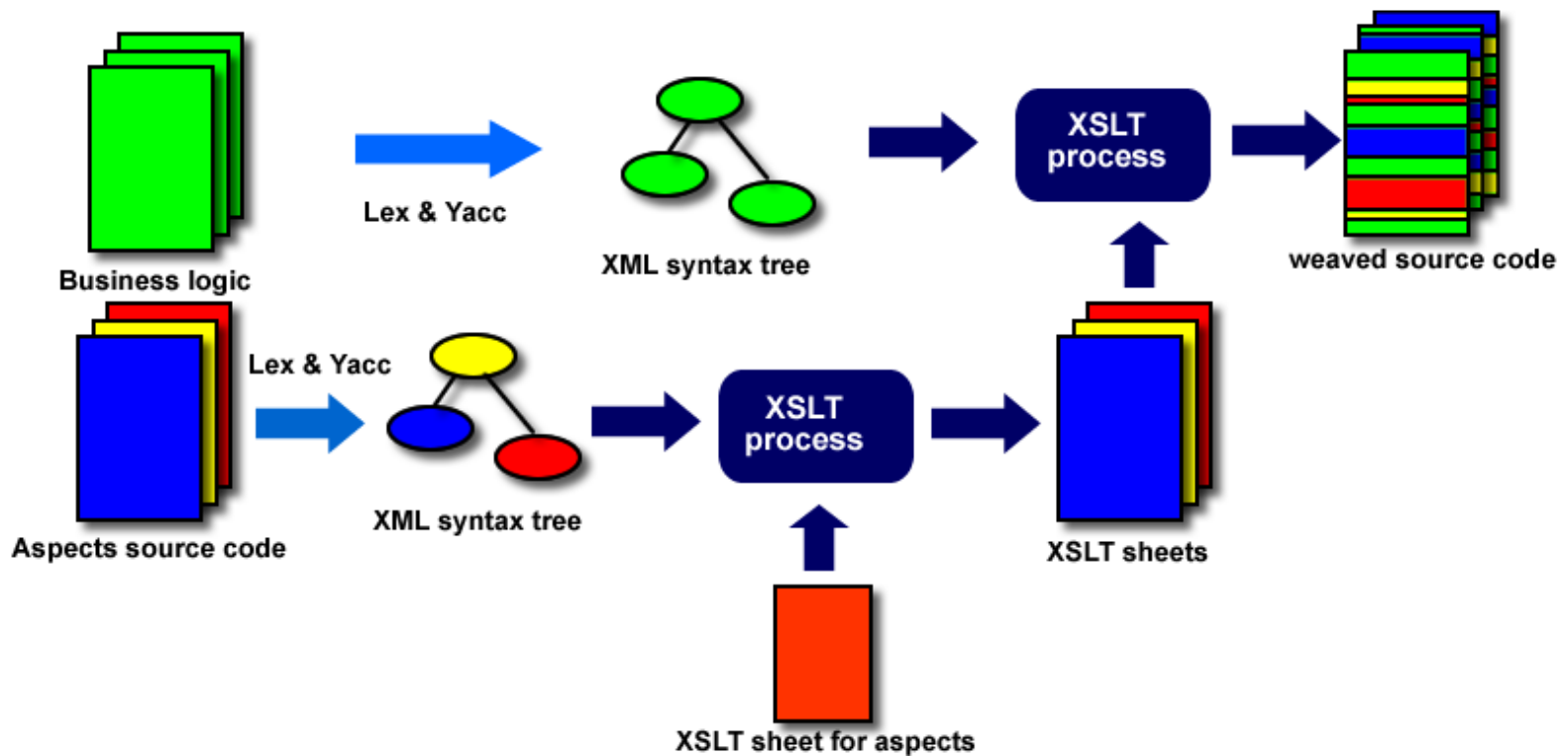
- phpAspect – юная, но наиболее перспективная реализация АОП для PHP
 - Интересная попытка клонировать AspectJ
 - Аспекты вплетаются статически в код
 - Автор William Candillon
- aoPHP – некое подобие АОП
 - Аспекты вплетаются «на лету»
 - Замороженный способ использования: Apache -> mod_rewrite -> aoPHP C++ интерпретатор -> PHP ...бр-р-р)
- aspectPHP – форк aoPHP
 - Не обновлялась с 2005 г
 - Работает только с PHP-4.3.10
- AOP Library for PHP – спорная эмуляция АОП средствами PHP,
 - Автор некто Dmitry Sheiko ☺
- runkit – PECL модуль, предоставляющий AOP Introduction
ВОЗМОЖНОСТИ
 - Переопределение констант, функций, методов, классов, интерфейсов
 - Эх, жаль, что не в core
 - Автор Sara Golemon



• Установка:

```
# pecl install -f Parse_Tree  
# pear install PHP_Beautifier  
# pear install Console_GetOpt
```

• Принцип действия:





- **Базовое использование:**

```
$ php phraspect.php <путь/до/исходников> <путь/до/аспектов>  
<конечная/директория>
```

```
$ php phraspect.php -d src src bin
```



- **АОП поддержка:**
 - PointCut
 - Advice
 - Introduction
 - Aspect
- см. раздаточный материал за подробностями



“HelloWorld”

(ну куда же без него)

- **src/hello.php**

```
<?php
class HelloWorld {
    function say() {
        echo "Hello!\n";
    }
}
$hello = new HelloWorld();
$hello->say();
?>
```

```
$ php hello.php
Hello!
```

Пример аспекта

- **src/trace.aspect.php**

```

<?php
aspect Trace{
    pointcut traceNew:new(*(0));
    pointcut traceSay:call(*->say(0));
    after(): traceNew{
        echo "After a construction of " .
            get_class($thisJoinPoint->getObject()) .
            "\n";
    }
    before(): traceSay{
        echo "Before a saying of " .
            get_class($thisJoinPoint->getTarget()) .
            "\n";
    }
    around(): traceSay{
        echo "Around a saying of " .
            get_class($thisJoinPoint->getTarget()) .
            "\n";
        $res = proceed();
        echo "\nend around\n";
        return $res;
    }
    after(): traceSay{
        echo "After a saying of " .
            get_class($thisJoinPoint->getTarget()) .
            "\n";
    }
}
?>

```

В



ДЕЙСТВИИ

- **Вплетение аспектов**

```
$ php phaspect.php src src bin
```

- **Выполнение переплетенного кода**

```
$ php bin/hello.php  
After a construction of HelloWorld  
Before a saying of HelloWorld  
Around a saying of HelloWorld  
Hello!  
end around  
After a saying of HelloWorld
```





- **bin/hello.php**

```
...
<?php
class HelloWorld {
    function say() {
        $__return_result = $this->__phpaspectsay();
        return $__return_result;
    }
    function __phpaspectsay() {
        echo "Hello!\n";
    }
}
$phpaspect_70 = new HelloWorld();
if (true) {
    $thisJoinPoint = new NewJoinPoint('', __LINE__, __FILE__, array(),
$phpaspect_70);
    function __phpaspectba49ac85769ed0c6e09c7c12487053d2($thisJoinPoint) {
        echo "After a construction of " . get_class($thisJoinPoint-
>getObject()) . "\n";
        unset($thisJoinPoint);
    }
    __phpaspectba49ac85769ed0c6e09c7c12487053d2($thisJoinPoint);
}
...

```

```
$phpaspect_56 = &$hello;
$phpaspect_56 = $phpaspect_70;
$phpaspect_87 = &$hello;
if (isCallType($phpaspect_87, '*', 'say', 'say')) {
    $thisJoinPoint = new CallJoinPoint('', __LINE__, __FILE__, array(),
$phpaspect_87, 'say');
    function __phpaspectfff7205121179f7e637a085e06b4bef62($thisJoinPoint) {
        echo "Before a saying of " . get_class($thisJoinPoint->getTarget()) .
"\n";
        unset($thisJoinPoint);
    }
    __phpaspectfff7205121179f7e637a085e06b4bef62($thisJoinPoint);
}
if (isCallType($phpaspect_87, '*', 'say', 'say')) {
    $thisJoinPoint = new CallJoinPoint('', __LINE__, __FILE__, array(),
$phpaspect_87, 'say');
    echo "Around a saying of " . get_class($thisJoinPoint->getTarget()) .
"\n";
    $res = $phpaspect_87->say();
    echo "\nend around\n";
    $__return_result = $res;
    unset($thisJoinPoint);
} else {
    $phpaspect_104 = &$phpaspect_87->say();
    $__return_result = $phpaspect_104;
}
```

Yikes! 😞

```
$phpaspect_104 = $__return_result;
if (isCallType($phpaspect_87, '*', 'say', 'say')) {
    $thisJoinPoint = new CallJoinPoint('', __LINE__,
__FILE__, array(), $phpaspect_87, 'say');
    function
__phpaspecte2600e1d66b7ca11ec71f56332b62ade($thisJoinPoint)
{
    echo "After a saying of " .
get_class($thisJoinPoint->getTarget()) . "\n";
    unset($thisJoinPoint);
}
__phpaspecte2600e1d66b7ca11ec71f56332b62ade($thisJoinPoint);
}
$phpaspect_104;
?>
```



- Достоинства
 - “Это” сделали для PHP!
 - Наиболее схожая с AspectJ функциональность



Недостатки

- На данный момент не подходит для production – некоторые аспекты вплетаются с ошибками
(хотя чего мы хотим от версии 0.1.1?)
- Функционально неполная реализация АОП
(работа с аннотациями, расстановка приоритетов применения аспектов, наследование аспектов и проч.)
- Генерируемый код накладывает негативный отпечаток на скор



АОП – еще одна «серебряная пуля»?

- Конечно же нет, такой «пули не существует»
- Естественно есть ярые фанаты и отчаянные противники АОП
 - АОП, в самом деле, позволяет посмотреть на проблему сквозного функционала на качественно ином уровне
 - В то же время АОП местами нетривиальная и непрозрачная методология
- Как всегда, истина где-то посередине

АОП - достоинства

- Эффективно адресует проблему сквозного функционала
- Облегчает повторное использование кода - слабо связанные между собой аспекты легко взаимозаменять
- Позволяет отложить принятие спорного решения, касающегося работы всего приложения, на «ПОТОМ»

АОП - недостатки

- Неочевидность происходящего (слишком много “магии”).
- Аспекты сложно (невозможно?) протестировать отдельно от сплетенного кода.
- Требуются отличные от ООП паттерны проектирования аспектов

«А оно вообще надо?» -
решать исключительно
вам :-)

Ссылки по теме

- AspectJ – <http://aspectj.org>
- <http://aspectmentor.com>
- phpAspect – <http://phpaspect.org>
- aoPHP – <http://www.aopphp.net>
- aspectPHP –
<http://www.cs.toronto.edu/~yijun/aspectPHP>
- AOP Library for PHP –
<http://www.phpclasses.org/browse/package/2633.html>
- PECL runkit – <http://pecl.php.net/runkit>

Вопросы?

