ORM實戰效能

PHPConf Taiwan 2015

@sdpower

WHO AM I

- SDpower info@sd.idv.tw
- 10+ year PHP
- http://blog.sd.idv.tw/
- https://github.com/SDpower
- exosite.com S.R Developer
- Performance Architect / Systems Architect

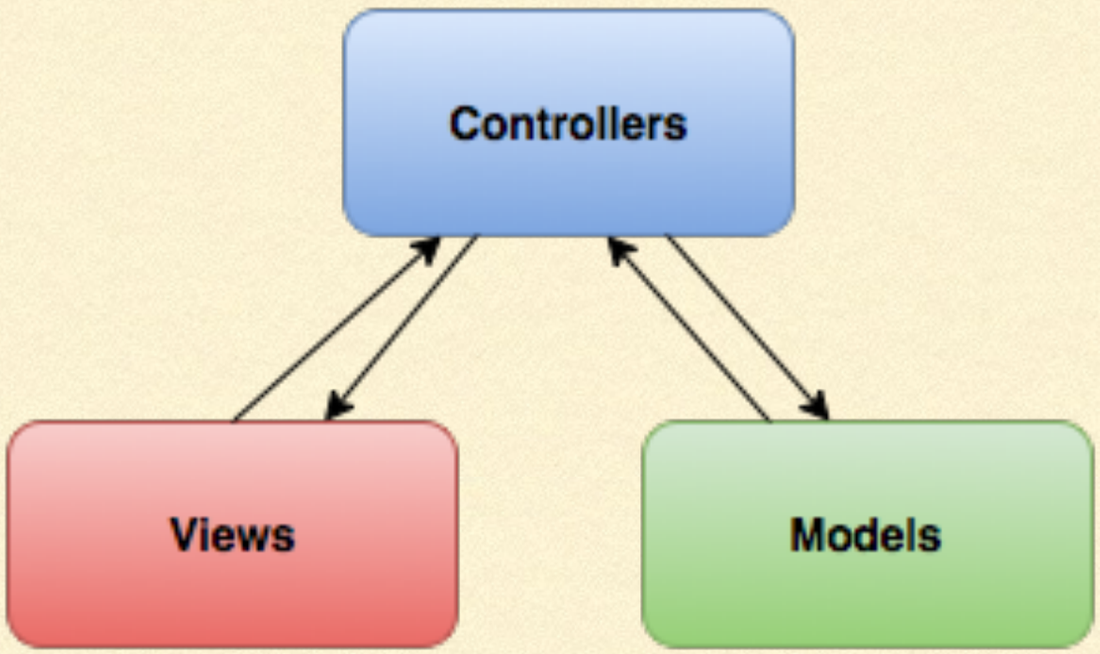
PhalconPHP

freamwork

Object Relational Mapper

O.R.M 物件關聯對映

當我們開始使用M.V.C



- Controller控制:輸入輸出資料或處理
- Models模型:存取資料
- Views:顯示輸出

FIRST MODEL 或許像是這樣

參雜許多SQL 語法並且是寫死的,資料表欄位異動困難維護異動不易。

```
class PostModel
{
    public function GetTotalnNmber()
    {
        $sql = 'SELECT COUNT(post_id) AS num_posts
        FROM ' . POSTS_TABLE . "
        WHERE topic_id = $topic_id
            AND post_time >= $min_post_time
        " . (($auth->acl_get('m_approve', $forum_id)) ? '' : 'AND post_approved = 1');
        $result = $db->sql_query($sql);

        $total_posts = (int) $db->sql_fetchfield('num_posts');
        return $total_posts;
    }
```

MODEL 只後有了分離DB與MODEL層
並且配合加上了一些快取。

```php
    private $_dbAdapter = null;

    /**
     * Config of Zend_Cache_Manager
     *
     * @var Zend_Cache
     */
    private $_CacheDB = null;
    private $_AdvertiseCache = null;

    /**
     * Constructor
     *
     */
    public function __construct()
    {
        $this->_dbAdapter = Zend_Registry::get('db');
        $this->_CacheDB = Zend_Registry::get('CacheDB');
        $this->_Table = DB_Prefix.'latest_news';
        $this->_Colum=array('headline','content');
        $this->_AdvertiseCache = $this->_CacheDB->getCache('AdvertiseCache');
        $this->_total = 40;
        $this->_page_num = 10;
    }
    public function Get_Laste_News()
    {
        if(!$result = $this->_AdvertiseCache->load('Laste_News')) {
            $db_select = $this->_dbAdapter->select();
            $db_select->from(array('n' => $this->_Table), $this->_Colum)
                                ->where('`n`.`status` = 1')
                                ->order('n.date_added DESC')
                               ->limit(MAX_DISPLAY_LATEST_NEWS);
                $result = $this->_dbAdapter->fetchAll($db_select);
                $this->_AdvertiseCache->save($result, 'Laste_News');
            }
            if (is_array($result)){
```

使用 O.R.M 開發MODEL

```
mysql> select * from robots;
+----+------------+------------+------+
| id | name       | type       | year |
+----+------------+------------+------+
|  1 | Robotina   | mechanical | 1972 |
|  2 | Astro Boy  | mechanical | 1952 |
|  3 | Terminator | cyborg     | 2029 |
+----+------------+------------+------+
```

- 預先建立好的資料。

使用 O.R.M 開發MODEL

```php
<?php

use Phalcon\Mvc\Model;

class Robots extends Model
{
    public $id;

    public $name;

    public $price;
}
```

- 預先建立好的資料。
- 簡單描述即可依照物件方式存取資料。

使用 O.R.M 開發MODEL

```php
<?php

// Find record with id = 3
$robot = Robots::findFirst(3);

// Prints "Terminator"
echo $robot->name;

$robot->name = "RoboCop";
$robot->save();
```

- 預先建立好的資料。
- 簡單描述即可依照物件方式存取資料。
- 讀取資料，變更資料物件方式處理都非常直覺物件化。

使用 O.R.M 開發MODEL

```php
// What's the first mechanical robot in robots table?
$robot = Robots::findFirst("type = 'mechanical'");
echo "The first mechanical robot name is ",
    $robot->name, PHP_EOL;

// Get first virtual robot ordered by name
$robot = Robots::findFirst(
    array(
        "type = 'virtual'",
        "order" => "name"
    )
);
echo "The first virtual robot name is ",
    $robot->name, PHP_EOL;
```

- 預先建立好的資料。
- 簡單描述即可依照物件方式存取資料。
- 讀取資料，變更資料物件方式處理都非常直覺物件化。
- 找到你想要的資料是很簡單的。

使用 O.R.M 開發MODEL

```php
$robot = Robots::findFirst(
    array(
        "type = 'virtual'",
        "order" => "name DESC",
        "limit" => 30
    )
);

$robots = Robots::find(
    array(
        "conditions" => "type = ?1",
        "bind"       => array(1 => "virtual")
    )
);
```

- 預先建立好的資料。
- 簡單描述即可依照物件方式存取資料。
- 讀取資料，變更資料物件方式處理都非常直覺物件化。
- 找到你想要的資料是很簡單的。
- 漸漸的你不太需要SQL語法了。

使用 O.R.M 開發MODEL

```
$robots = Robots::query()
    ->where("type = :type:")
    ->andWhere("year < 2000")
    ->bind(array("type" => "mechanical"))
    ->order("name")
    ->execute();
```

- 預先建立好的資料。
- 簡單描述即可依照物件方式存取資料。
- 讀取資料，變更資料物件方式處理都非常直覺物件化。
- 找到你想要的資料是很簡單的。
- 漸漸的你不太需要SQL語法了。
- 新手也可以快速學習與使用。

MODEL 設計關聯式資料庫

```php
<?php

use Phalcon\Mvc\Model;

class Robots extends Model
{
    public $id;

    public $name;

    public function initialize()
    {
        $this->hasMany("id", "RobotsParts", "robots_id");
    }
}
```

- The model “Robots” has many(一對多) “RobotsParts”.

MODEL 設計關聯式資料庫

```php
<?php

use Phalcon\Mvc\Model;

class Parts extends Model
{
    public $id;

    public $name;

    public function initialize()
    {
        $this->hasMany("id", "RobotsParts", "parts_id");
    }
}
```

- The model “Robots” has many(一對多) “RobotsParts”.
- The model “Parts” has many(一對多) “RobotsParts”.

MODEL 設計關聯式資料庫

```php
<?php

use Phalcon\Mvc\Model;

class RobotsParts extends Model
{
    public $id;

    public $robots_id;

    public $parts_id;

    public function initialize()
    {
        $this->belongsTo("robots_id", "Robots", "id");
        $this->belongsTo("parts_id", "Parts", "id");
    }
}
```

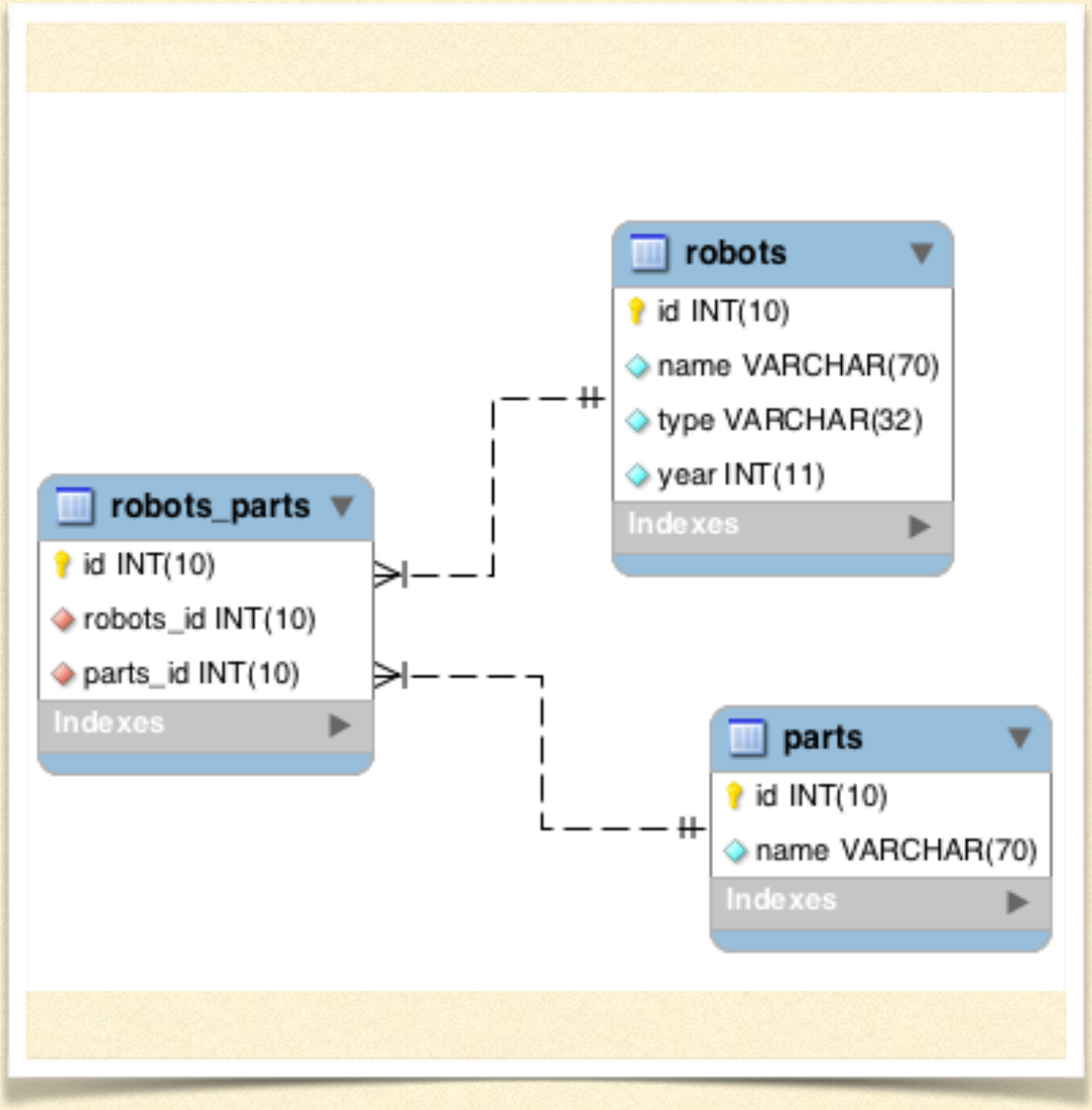
- The model “Robots” has many(一對多) “RobotsParts”.
- The model “Parts” has many(一對多) “RobotsParts”.
- The model “RobotsParts” belongs to both “Robots” and “Parts” models as a many-to-one relation.

MODEL 設計關聯式資料庫

```php
<?php

use Phalcon\Mvc\Model;

class Robots extends Model
{
    public $id;

    public $name;

    public function initialize()
    {
        $this->hasManyToMany(
            "id",
            "RobotsParts",
            "robots_id", "parts_id",
            "Parts",
            "id"
        );
    }
}
```

- The model “Robots” has many(一對多) “RobotsParts”.
- The model “Parts” has many (一對多)“RobotsParts”.
- The model “RobotsParts” belongs to both “Robots” and “Parts” models as a many-to-one relation.
- The model “Robots” has a relation many-to-many to “Parts” through “RobotsParts"

MODEL 設計關聯式資料庫



- The model “Robots” has many (一對多) “RobotsParts”.
- The model “Parts” has many (一對多) “RobotsParts”.
- The model “RobotsParts” belongs to both “Robots” and “Parts” models as a many-to-one relation.
- The model “Robots” has a relation many-to-many to “Parts” through “RobotsParts"
- EER diagram to understand.

O.R.M 與測試

保持簡單KEEP IT SIMPLY

- 如果你有個shop car(購物車)model.

保持簡單KEEP IT SIMPLY

- 如果你有個shop car(購物車)model.
- 購物清單放在SESSION.

保持簡單KEEP IT SIMPLY

- 如果你有個shop car(購物車)model.
- 購物清單放在SESSION.
- 那你是在測試shop car 還是session?

CONTROLLER 測試

- 可以實際hit database做測試。
- 可以善用migration 做測試的database情境模擬。
- 使用Mocking方式來分離依賴做測試。

CONTROLLER 測試

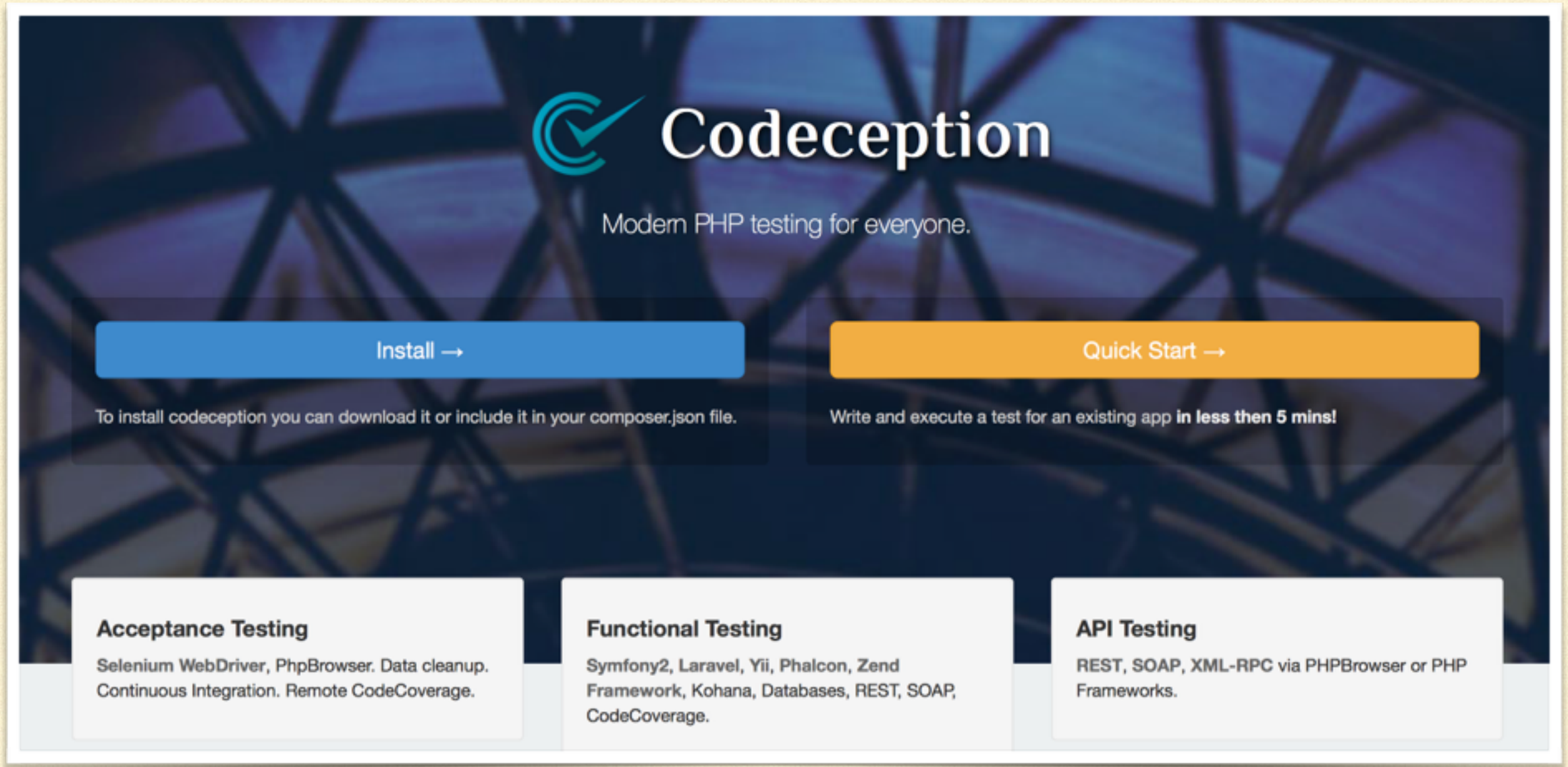
```php
?php

lass UnitTest extends \UnitTestCase


   public function testTestCase()
   {
       $facade = new Facade();
       $facade->setDI($di);
       $actual = $facade->createNewPopup('Test Popup', 'This is a test')
       $this->assertTrue(is_array($actual));
   }
```

- 可以實際hit database做測試。

CONTROLLER 測試

```
{
    protect function setUp()
    {
        parent::setUp();
         // please assume I have setup relative services in below.
        $this->di->set('request', ..);
        $this->di->set('modelsManager', ..);
        $this->di->set('dispatcher', ...)
    }

    public function testGetAction()
    {
        $a_mock = $this->getMockBuilder('A')
                       ->setMethods(array('findFirst'))
                       ->getMock();

        $a_mock->expects($this->once())
               ->method('findFirst')
               ->will($this->returnValue(null));



        $controller = new AController();
        $controller->setAModel($a_mock);
        // do other stuff
        $controller->getAction();
    }
```

- 可以實際hit database做測試。
- 使用Mocking方式來分離依賴做測試。



CODECEPTION IS A BDD-STYLED PHP TESTING FRAMEWORK

try http://codeception.com/

O.R.M 與快取機制

O.R.M 快取

```php
<?php

use Phalcon\Cache\Frontend\Data as FrontendData;
use Phalcon\Cache\Backend\Memcache as BackendMemcache;

// Set the models cache service
$di->set('modelsCache', function () {

    // Cache data for one day by default
    $frontCache = new FrontendData(
        array(
            "lifetime" => 86400
        )
    );

    // Memcached connection settings
    $cache = new BackendMemcache(
        $frontCache,
        array(
            "host" => "localhost",
            "port" => "11211"
        )
    );

    return $cache;
});
```

- 設定好 cache 服務。

O.R.M 快取

```php
<?php

// Get products without caching
$products = Products::find();

// Just cache the resultset. The cache will expire in 1 hou
$products = Products::find(
    array(
        "cache" => array(
            "key" => "my-cache"
        )
    )
);

// Cache the resultset for only for 5 minutes
$products = Products::find(
    array(
        "cache" => array(
            "key"      => "my-cache",
            "lifetime" => 300
        )
    )
);
```

- 設定好 cache 服務。
- find時指定要有快取，life time 預設為1小時。

O.R.M 快取

```php
<?php

use Phalcon\Mvc\Model;

class CacheableModel extends Model
{
    protected static function _createKey($parameters)
    {
        // ... Create a cache key based on the parameters
    }

    public static function find($parameters = null)
    {
        // ... Custom caching strategy
    }

    public static function findFirst($parameters = null)
    {
        // ... Custom caching strategy
    }
}
```

- 設定好 cache 服務。
- find時指定要有快取，life time 預設為1小時。
- 每次都要帶參數有點麻煩，直接繼承物件model 改變自己想要的資料讀取快取方式。

O.R.M 快取

```php
<?php

$phql = "SELECT * FROM Cars WHERE name = :name:";

$query = $this->modelsManager->createQuery($phql);

$query->cache(
    array(
        "key"      => "cars-by-name",
        "lifetime" => 300
    )
);

$cars = $query->execute(
    array(
        'name' => 'Audi'
    )
);
```

- 設定好 cache 服務。
- find時指定要有快取，life time 預設為1小時。
- 每次都要帶參數有點麻煩，直接繼承物件model 改變自己想要的資料讀取快取方式。
- SQL Query PHQL 快取也是沒問題。

O.R.M 與資料優化

O.R.M 你必須了解的事

```php
<?php

class Product extends \Phalcon\Mvc\Model
{
    public $id;
    public $name;
    public $price;
}
```

- 有個產品model.

O.R.M 你必須了解的事

```php
public function CreateProductAction()
{
    $mtime = explode(" ", microtime());
    $startTime = $mtime[1] + $mtime[0];

    $formate = "MyProduct%05d";
    for ($i = 0; $i < 9999; $i++) {
        $product = new Product();
        $name = sprintf($formate, $i);
        $product->assign([
            'name' => $name,
            'price' => rand(5, 999999)
        ]);
        if (!$product->save()) {
            var_dump($product->getMessages());
        }
        echo '.';
    }

    $mtime = explode(" ", microtime());
    $endTime = $mtime[1] + $mtime[0];
    $totalTime = ($endTime - $startTime);
    echo PHP_EOL.$totalTime." seconds".PHP_EOL;
}
```

- 有個產品model.
- 假設我建立資料9999筆。

O.R.M 你必須了解的事

- 有個產品model.
- 假設我建立資料9999筆。

O.R.M 你必須了解的事

```php
public function ProductPriceChangeAction()
{
    $mtime = explode(" ", microtime());
    $startTime = $mtime[1] + $mtime[0];

    $products = Product::find();
    echo "Total Products:".count($products).PHP_EOL;
    foreach ($products as $product) {
        $product->price = $product->price + 5;
        $product->save();
    }

    $mtime = explode(" ", microtime());
    $endTime = $mtime[1] + $mtime[0];
    $totalTime = ($endTime - $startTime);
    echo PHP_EOL.$totalTime." seconds".PHP_EOL;
}
```

- 有個產品model.
- 假設我建立資料9999筆。
- 然後改資料調整價錢。

O.R.M 你必須了解的事

```
-> % php cli.php main ProductPriceChange
Total Products:9999

6.7665848731995 seconds
php cli.php main ProductPriceChange  3.78s user
 0.31s system 60% cpu 6.826 total
```

- 有個產品model.
- 假設我建立資料9999筆。
- 然後改資料調整價錢。
- 這些看起來正常不過但是出了點問題。

「PHP 不是高速運算設計的語言。」

O.R.M 你必須了解的事

```php
public function ProductPriceChange2Action()
{
    $mtime = explode(" ", microtime());
    $startTime = $mtime[1] + $mtime[0];

    echo "Total Products:".Product::count().PHP_EOL;
    $query = "UPDATE Product SET price = price + 5;";
    $this->db->query($query);

    $mtime = explode(" ", microtime());
    $endTime = $mtime[1] + $mtime[0];
    $totalTime = ($endTime - $startTime);
    echo PHP_EOL.$totalTime." seconds".PHP_EOL;
}
```

```
-> % php cli.php main ProductPriceChange2
Total Products:9999

0.048919916152954 seconds
```

- 有個產品model.
- 假設我建立資料10000筆。
- 然後改資料調整價錢。
- 這些看起來正常不過但是出了點問題。
- 請拒絕大量物件操作，不論是不是在使用O.R.M 那樣只會增加你的負載沒有任何價值。

集合處理不能再model以外並且處理集合盡量使用

RowData。

O.D.M (OBJECT-DOCUMENT MAPPER)

O.D.M MONGODB

```php
<?php

use Phalcon\Mvc\Collection;

class Robots extends Collection
{
    public function getSource()
    {
        return "the_robots";
    }
}
```

- 建立一個Robots O.D.M很簡單．

O.D.M MONGODB

```php
// How many robots are there?
$robots = Robots::find();
echo "There are ", count($robots), "\n";

// How many mechanical robots are there?
$robots = Robots::find(
    array(
        array(
            "type" => "mechanical"
        )
    )
);
echo "There are ", count($robots), "\n";

// Get and print mechanical robots ordered by name upward
$robots = Robots::find(
    array(
        array(
            "type" => "mechanical"
        ),
        "sort" => array(
            "name" => 1
        )
    )
);

foreach ($robots as $robot) {
    echo $robot->name, "\n";
}
```

- 建立一個Robots O.D.M很簡單．
- 一樣物間方式的存取．

O.D.M MONGODB

```php
<?php

// First robot where type = "mechanical" and year = "1999"
$robot = Robots::findFirst(
    array(
        "conditions" => array(
            "type" => "mechanical",
            "year" => "1999"
        )
    )
);

// All virtual robots ordered by name downward
$robots = Robots::find(
    array(
        "conditions" => array("type" => "virtual"),
        "sort"       => array("name" => -1)
    )
);
```

- 建立一個Robots O.D.M很簡單．
- 一樣物間方式的存取．
- 但是沒有關聯性的物件可用了．

或許你會有興趣的。

或許你會有興趣的。

- ActiveRecord :
  http://www.phpactiverecord.org/
  http://laravel.com/docs/5.1/eloquent
  http://www.yiiframework.com/doc/guide/1.1/en/database.ar
  http://propelorm.org/

- Sharing：
  http://sysmagazine.com/posts/129780/
  https://github.com/dizeee/yii-sharding
  https://github.com/ienaga/RedisPlugin

歡迎加入EXOSITE

welcome join exosite

THANKS YOU