Hướng dẫn sử dụng

1. Yêu cầu:
   1. Cài đặt wamp, xamp: Bật chức năng rewrite module trong apache
   2. Cấu hình host ảo
   3. Cấu hình httpd.conf
2. pzk\_parse:

Chạy file định dạng xml(đuôi vẫn là .php). Các bước chạy như sau:

$obj = pzk\_parse(‘system/full’);

File system/full.php mẫu

<core.system **id="system"**>

<core.request **id="request"** />

<core.rewrite.request matcher="host" matchMethod="equal" pattern="qlhs.vn"

defaultQueryParams='{"app":"qlhs", "controller": "Student", "action": "index"}' />

<core.loader **id="loader"** />

</core.system>

Chạy <core.system>

Đọc file objects/Core/System.php

Tạo đối tượng PzkCoreSystem, thêm thuộc tính id vào cho đối tượng

Chạy hàm init của PzkCoreSystem

Chạy tiếp <core.request>

Tạo đối tượng PzkCoreRequest, thêm thuộc tính id vào cho đối tượng

Chạy hàm init cho PzkCoreRequest

Chạy hàm finish của PzkCoreRequest

Chạy tiếp <core.rewrite.request>

Tạo đối tượng PzkCoreRewriteRequest, thêm thuộc tính matcher, matchMethod, pattern, defaultQueryParams vào cho đối tượng

Chạy hàm init cho PzkCoreRewriteRequest

Chạy hàm finish của PzkCoreRewriteRequest

Chạy tiếp <core.loader>

Tạo đối tượng PzkCoreLoader, thêm thuộc tính id vào cho đối tượng

Chạy hàm init cho PzkCoreLoader

Chạy hàm finish của PzkCoreLoader

Chạy hàm finish của PzkSystem

Giống như ta đang duyệt cây theo đệ quy: hàm init của một node luôn chạy trước khi các node con của nó init, hàm finish chạy sau khi các node con của nó đã finish

Sau khi chạy xong ta có thể dựa vào id để lấy ra đối tượng được tạo ra:

$requestObject = **pzk\_element(‘request’);**

echo $requestObject->matcher;

echo $requestObject->pattern;

$loaderObject = **pzk\_element(‘loader’);**

$model = $loaderObject->createModel(‘user’);

$user = $model->login(‘admin’, ‘123456’);

if($user) {

echo ‘Login success’;

} else {

echo ‘Login Failed’;

}

1. Cấu hình hệ thống:

File: system/full.php

1. Cấu hình ứng dụng:

File: app/qlhs/application.php

1. Objects

Objects là các đối tượng khai báo để được parse ra từ file XML. Nó có thể được kế thừa từ 3 đối tượng trong hệ thống: /core/Object.php, /core/Object/LightWeight.php, /core/Object/Smarty.php

Một Object kế thừa từ PzkObject trong file /core/Object.php có các thuộc tính cơ bản sau:

layout: giống như khai báo view cho object

scriptable: có javascript file không (mục này liên quan đến javascript object)

scriptTo: đặt script vào đâu(khai báo id của đối tượng đặt script)

cacheable: có cache hay không

cacheParams: các tham số để cache

pzkParentId: id của node cha

cssLink: file css đi kèm với đối tượng

cssExternalLink: file css ngoài

pclass: giống như thuộc tính class của css nhưng dùng để cấu hình mở rộng cho xml

Các Phương Thức Phổ Biến:

display(): hiển thị nội dung

getContent(): lấy nội dung

append($object): cho một node con vào sau danh sách các node con của object đó

prepend($object): cho node con vào đằng trước

insertObject($object, $index): chèn node con vào vị trí index

index(): lấy vị trí của node trong danh sách node

before($obj): chèn vào đằng trước trong danh sách node

after($obj): chèn vào đằng sau trong danh sách node

getParent(): lấy node cha

getChildren($selector): lấy các node con. Selector dạng [thuộc tính=,^=,\*=,!=,giá trị](tương tự như jquery selector)

findElement($selector): tìm phần tử con theo $selector

findElements($selector): tìm các phần tử con theo $selector

findParents($selector): tìm các parents theo $selector

displayChildren($selector): display các children theo selector

matchSelector($selector): kiểm tra xem có $selector không

getProp($prop, $default = NULL): lấy thuộc tính của đối tượng

getModel($model): Lấy model trong thư mục /models

1. Controller

Các controller được tìm ở trong /app/qlhs/controller. Nếu không tìm thấy thì tìm trong thư mục /controller

Trong các controller này ta thường gọi: $page = pzk\_parse(‘index’); $page->display();…..

Các controller được kế thừa từ: PzkController trong /core/Controller.php, hoặc /core/controller/\*

Các phương thức:

getModel($model): trả về model

getStructure($structure): lấy từ /pages ra, vd getStructure(‘index’) thì lấy từ pages/index.php

public $masterStructure = 'masterStructure';

public $masterPosition = 'left';

viewStructure($structure, $useMasterStructure = true) : Hiển thị cấu trúc $structure (lấy ở pages/$struture) có sử dụng masterStructure hay không. Nếu có sử dụng masterStructure thì sẽ gọi masterStructure trước và append $structure vào trong vùng masterPosition

1. Pages

Các page chứa các cấu trúc của trang dạng xml

1. Layouts

Chứa các view

1. Model

Chứa các model

1. Entity

Một dạng model tuy vậy nó gắn với các bản ghi của CSDL

Các phương thức chính:

setData($data): tạo data cho entity

getData(): trả về data cho entity

update($data): update dữ liệu

save(): lưu dữ liệu

delete(): xóa bản ghi khỏi csdl

1. Database Query

Sử dụng hàm \_db() thay cho pzk\_element(‘db’);

Các phương thức:

select($fields): chọn các trường

from($table): từ bảng

where($conds): điều kiện

groupBy($groupBy): nhóm theo

orderBy($orderBy): sắp xếp theo

having($conds): điều kiện having

limit($pageSize, $pageNum): giới hạn số bản ghi cần hiển thị

update($table): update một bảng

set($data): đặt dữ liệu cho lệnh update

insert($table): insert vào bảng

fields($fields): các fields cần insert

values($arr): các bản ghi cần insert

result($entity = false): thực thi query, trả về mảng dạng các đối tượng kiểu $entity, nếu $entity = false thì trả về dạng mảng bình thường

result\_one($entity = false): trả về 1 bản ghi

getQuery(): lấy query dạng string

clear(): reset lại

query($query): thực thi $query

query\_one($query): trả về 1 bản ghi

getFields($table): trả về các trường của 1 bảng

getEntity($entity): lấy đối tượng kiểu entity

useCB(): sử dụng condition builder

khi sử dụng condition builder, các condition có thể được ghi dưới dạng mảng:

$conds = array(‘and’, array(‘equal’, ‘type’, ‘Student’), array(‘like’, ‘name’, ‘%Nguyễn%’));

\_db()->useCB();

\_db()->where($conds);

**Request**

Một số thay đổi mới trong hệ thống

Để đáp ứng yêu cầu phát triển, hệ thống có thêm một vài bổ sung như sau:

**objects/core/Request.php**

*Lấy biến request theo cách mới:*

$request = pzk\_request();

$active = pzk\_request(‘active’); // hoặc $request->get(‘active’) hoặc pzk\_request()->get(‘active’);

Cách thông thường:

$request = pzk\_element(‘request’);

$active = $request->get(‘active’);

*Kiểm tra xem request là post hay get:*

if($request->is(‘post’)) {

            echo ‘Vừa post dữ liệu’;

}

*Lấy segment của uri:*

Một uri có dạng home/category/4 là dạng phân cách nhau bới dấu /. Để lấy đoạn thứ i nào đó ta dùng: $request->getSegment(i);

Trong đoạn uri trên ta echo $request->getSegment(3); sẽ cho kết quả là 4

*Redirect về một url*:

Để redirect về một url ta viết: header(‘Location: ’ . $url); hoặc viết:

$request->redirect($url);

Phần này cần sửa lại sao cho redirect được theo uri như vd: $request->redirect(‘home’); hoặc $request->redirect(‘home/index’);

*Lấy dữ liệu từ request:*

Lấy mảng các dữ liệu theo các trường xác định. VD: post lên url user/registerPost các dữ liệu:

name: John Doe

username: doe

password: 123456

email: [doe@example.com](mailto:doe@example.com)

phone: 01222333444

$data = $request->getFilterData(‘name, username, password, phone, email’);

Trả về mảng $data có dạng:

array(‘name’ => ‘John Doe’, ….., ‘email’ => ‘[doe@example.com](mailto:doe@example.com)’)

*Tạo 1 url hoàn chỉnh build($route, $query = false):*

$categoryPageUrl = $request->build(‘home/category/4’, array(‘page’ => 2));

echo $categoryPageUrl;

Trả về url: <http://ptnn.vn/home/category/4?page=2>

*Tạo url từ url hiện tại buildCurrent($query = false):*

VD đang ở url <http://ptnn.vn/home/category/4?page=2> ta gọi

$categoryPageUrl = $request->buildCurrent( array(‘page’ => 3));

Trả về url: <http://ptnn.vn/home/category/4?page=3>

*Tạo url từ controller hiện tại:buildAction($action = false, $query = false):*

VD đang ở url <http://ptnn.vn/home/category/4?page=2> controller hiện tại là *home* ta gọi

$articlePageUrl = $request->buildAction (‘article’, array(‘id’ => 6));

Trả về url: <http://ptnn.vn/home/article?id=6>

**objects/core/Notifier.php**

Thêm thông báo hệ thống: pzk\_notifier\_add\_message($message, $type);

Các $type là : active, success, info, warning, danger

Vd: pzk\_notifier\_add\_message(‘Tên đăng nhập hoặc mật khẩu sai’, ‘danger’);

**objects/core/Validator.php**

Validate các input: sử dụng gần tương tự như jqueryvalidation

$validator = pzk\_element(‘validator’); hoặc $validator = pzk\_validator();

*public function required($str): chuỗi nhập vào không được trống*:

vd:

$name = pzk\_request(‘name’);

if($validator->required($name) == false) {

pzk\_notifier\_add\_message(‘Tên không được để trống’, ‘danger’);

}

Tương tự với các hàm khác

            public function email($str)

            public function url($str)

            public function dateNormal($str)

            public function number($str)

            public function digits($str)

            public function maxlength($str, $len)

            public function minlength($str, $len)

            public function rangeLength($str, $min, $max)

            public function max($str, $max)

            public function min($str, $min)

            public function range($str, $min, $max)

            public function equalTo($str, $value)

*function validate($data, $validator);*

Để validate cả mảng:

$data = array(‘name’ => ‘A’, … , ‘phone’ => ‘0122334455’);

Với validator như sau:

$v = array(

            ‘rules’ => array(

            ‘name’ => array(

                        ‘required’ => true,

                        ‘maxlength’ => 50,

                        ‘minlength’ => 5

),

…..

),

‘messages’ => array(

            ‘name’ => array(

            ‘required’ => ‘Bạn phải nhập tên’,

            ‘maxlength’ => ‘Tên không dài quá 50 ký tự’,

            ‘minlength’ => ‘Tên không được ngắn hơn 5 ký tự’

),

….

)

);

Kết quả validate:

$result = $validator->validate($data, $v);

Nếu $result trả về true

if($result === true) {

            // không gặp lỗi

} else {

            // xử lý $result; $result là danh sách các lỗi gặp phải; var\_dump($result)

            // $result có dạng: array(‘name’ => array(‘required’ => ‘Bạn phải nhập tên’, ‘maxlength’ => ‘Tên không dài quá 50 ký tự’, ))

            // Cho vào lỗi hệ thống:

            foreach($result as $field => $messages) {

            foreach($messages as $rule => $message) {

                        pzk\_notifier\_add\_message($message, ‘danger’);

}

}

}

**objects/core/Controller.php**

Để lấy xem một vài biến đổi ta thử sửa đổi lại code của một action như sau:

// Hiển thị thông báo sau khi đăng ký tài khoản

            public function showregisterAction()

            {

                        $this->layout();

                        $pageUri = $this->getApp()->getPageUri('user/showregister');

                        $page = PzkParser::parse($pageUri);

                        $left = pzk\_element('left');

                        $left->append($page);

                        $this->page->display();

            }

Đầu tiên là gọi $this->layout(); hàm này lấy layout từ file pages/index.php

Thay vì vậy ta khai báo trong controller:

$masterPage = ‘index’;

$masterPosition = ‘left’;

public function showregisterAction() {

            $this->initPage(); // tương đương với gọi $this->layout();

            $this->append('user/showregister'); // append vào ‘left’ là masterPosition mặc định

// để append vào vị trí khác ta viết $this->append('user/showregister', ‘right’);

            $this->display();

}

Hoặc viết liền 1 mạch:

public function showregisterAction() {

            $this->initPage()->append('user/showregister') ->display();

}

Ta có thể append thêm tùy ý:

$this->initPage()->append('user/showregister') ->append('user/support’, ‘right’) ->append('user/avatar’, ‘right’) ->display();

Hàm append($page/\* hoặc $obj\*/, $position): nhận vào đối số là tên page hoặc là một đối tượng dạng PzkObject cho vào vị trí $position

*Hàm validate($data, $validator):* tương tự như hàm validate của pzk\_validator() tuy vậy tự động thêm cảnh báo(pzk\_notifier\_add\_message) vào cho hệ thống, và trả về giá trị true hoặc false;