Bài 9: Tìm Hiểu Library Form Validation

Trong bài này bạn sẽ được học:

- Sử dụng form validation kiểm tra dữ liệu
- Xuất các thông báo lỗi ra ngoài view

Trong bài này, chúng ta sẽ tìm hiểu việc kiểm soát dữ liệu người dùng nhập vào textbox, tạo ra những tập luật để kiểm duyệt dữ liệu, ví dụ dữ liệu ô nhập bắt buộc phải là con số, email phải đúng định dạng, … Giống như các bài trước, để có thể thao tác với bất kỳ library nào trong CI, chúng ta phải load nó ra.

1. Load library form validation

```
$this->load->library('form_validation')
//gọi phương thức
//$this->form_validation->ten_phuong_thuc()
```

Để có thể nắm bắt vấn đề tốt hơn khi thao tác với **form validation**, chúng ta sẽ liệt kê một số phương thức hay dùng. Đầu tiên là phương thức dùng để tạo ra các luật kiểm tra, đó là **set_rules**. Phương thức này nhận vào 3 giá trị: tên của textbox, tên đối tượng xuất ra thông báo lỗi, và giá trị quan trọng nhất chính là tập luật mà các bạn muốn quy định, các luật cách nhau bằng dấu |.

```
$this->form_validation->set_rules('username', 'Username', 'required')
```

- **required:** Yêu cầu phải nhập liệu không được để trống
- **matches:** Yêu cầu password phải trùng khớp nhau
- **min_length:** Giới hạn bao nhiêu ký tự khi nhập vào
- **max_length:** Giới hạn bao nhiêu ký tự khi nhập vào
- **numeric:** Yêu cầu trong textbox phải nhập liệu là con số
- **valid_email:** Email nhập liệu bắt buộc phải đúng định dạng
- **xss_clean:** Xóa XSS của input, bảo mật

CI cung cấp rất nhiều luật, tuy nhiên, ở trên mình chỉ liệt kê một số luật thường dùng trong thực tế. Ngoài ra, còn một phương thức rất là quan trọng nếu thiếu phương thức này thì xem như các luật phía trên bị vô hiệu hóa, đó là phương thức **run()**. Phương thức **run()** có nhiệm vụ kiểm tra các luật trên có hợp lệ hay không, nếu không hợp lệ thì nó sẽ trả về FALSE, tức là nó sẽ kiểm tra phương thức **set_rules** nếu một trong số các luật mà bạn khai báo bị lỗi thì nó sẽ trả về FALSE đồng thời trả về một cái view nào đó. Chúng ta có thể xuất câu thông báo lỗi ra bằng phương thức **validation_errors()**.

2. Kiểm tra nhập liệu với form validation

Tôi sẽ dùng lại controller & view của bài form helper, trong đó, chúng ta sẽ kiểm tra 3 textbox gồm fullname, password & email.

Controller:

```
<?php
class Form extends CI_Controller{
      public function __construct(){
          parent::__construct();
          $this->load->helper(array('url', 'form'));
      }

     public function index(){
         $this->load->view("form");
     }
}
?>
```

View

```
<?php
$user=array(
    "name" => "username",
    "size" => "25",
);
$pass=array(
    "name" => "pass",
    "size" => "25",
);
$email=array(
    "name" => "email",
    "size" => "25",
);
$gender1=array(
    "name" => "gender",
    "value" => "m",
    "checked" => TRUE,
);
$gender2=array(
    "name" => "gender",
    "value" => "f",
);
$opt=array(
    "1" => "Viet Nam",
    "2" => "Cambodia",
    "3" => "Malaysia",
);
$note=array(
    "name" => "note",
    "cols" => "40",
    "rows" => "5",
```

```
);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>freetuts.net</title>
</head>

<body>
 <?php
    echo form_open(base_url()."index.php/form/index");
    echo form_fieldset("Member Register");
    echo form_label("Fullname: ").form_input($user)."<br />";
    echo form_label("Password: ").form_password($pass)."<br />";
    echo form_label("Email: ").form_input($email)."<br />";
    echo form_label("Gender:
").form_radio($gender1)."Male".form_radio($gender2)."Female<br />";
    echo form_label("Country: ").form_dropdown("Country: ", $opt, 1)."<br
/>";
    echo form_label("Note: ").form_textarea($note)."<br />";
    echo form_label(" ").form_submit("ok",  "Register");
    echo form_fieldset_close();
    echo form_close();
 ?>
</body>
</html>
```

Công việc bây giờ của chúng ta là dùng form validation để check lỗi nhập liệu, nếu người dùng đã nhấn nút submit thì chúng ta sẽ tiến hành yêu cầu họ phải thao tác theo đúng các luật mà chúng ta đã đưa ra. Do ví dụ này chỉ có một action nên việc đầu tiên là chúng ta sẽ load form validation ra, một khi đã load xong library thì chúng ta hoàn toàn có thể sử dụng được toàn bộ các phương thức của nó. Phương thức đầu tiên chúng ta sử dụng sẽ là phương thức **set_rules()**, tên của textbox đầu tiên là fullname và đối tượng xuất ra thông báo lỗi sẽ là label Full Name. Chúng ta muốn người dùng bắt buộc phải nhập tên vào textbox, vì thế chúng ta sẽ dùng luật ***required***. Chúng ta không muốn cho họ nhập vào một cái tên quá dài, chúng ta ép họ phải nhập không được ít hơn 6 ký tự, chúng ta có phương thức ***min_length[6]***. Khi sử dụng nhiều hơn một luật cho cùng một đối tượng, chúng ta phải dùng dấu | để phân cách. Để kiểm tra xem email có đúng định dạng không, chúng ta sẽ dùng luật ***valid_email***. Sau khi khai báo tập luật xong, chúng ta gọi phương thức run() kiểm tra xem các **set_rules** có hợp lệ hay không, nếu một trong 3 luật trên bị lỗi thì ngay lập tức nó sẽ trả về là FALSE và thông báo lỗi sẽ xuất hiện bằng cách gọi phương thức **validation_errors**.

```
<?php
class Form extends CI_Controller{
```

```
      public function __construct(){
          parent::__construct();
          $this->load->helper(array('url', 'form'));
      }

     public function index(){
         $this->load->library('form_validation');
         $this->form_validation->set_rules('username', 'Full Name',
'required|min_length[6]');
         $this->form_validation->set_rules('pass', 'Pass Word',
'required');
         $this->form_validation->set_rules('email', 'Email',
'required|valid_email');

        if($this->form_validation->run() == FALSE){
            $this->load->view("form");
        }
     }
}
?>
<?php
    echo validation_errors();
    echo form_open(base_url()."index.php/form/index");
    echo form_fieldset("Member Register");
    echo form_label("Fullname: ").form_input($user)."<br />";
    echo form_label("Password: ").form_password($pass)."<br />";
    echo form_label("Email: ").form_input($email)."<br />";
    echo form_label("Gender:
").form_radio($gender1)."Male".form_radio($gender2)."Female<br />";
    echo form_label("Country: ").form_dropdown("Country: ", $opt, 1)."<br
/>";
    echo form_label("Note: ").form_textarea($note)."<br />";
    echo form_label(" ").form_submit("ok",  "Register");
    echo form_fieldset_close();
    echo form_close();
 ?>
```

Bây giờ, chúng ta hãy chạy link *localhost/citest/index.php/form* để kiểm tra, nếu chúng ta không nhập bất cứ thứ gì vào textbox mà nhấn nút submit thì ngay lập tức trình duyệt sẽ trả về kết quả sau.

The Full Name field is required.

The Password field is required.

The Email field is required.

Tức là yêu cầu phải nhập liệu, và nếu ở textbox fullname chúng ta nhập ít hơn 6 ký tự và email chúng ta điền không đúng định dạng thì thông báo lỗi sẽ trả về như sau.

The Full Name field must be at least 6 characters in length.

The Email field must contain a valid email address

Việc kiểm tra nhập liệu với form validation thì khá đơn giản, còn một vấn đề đặt ra ở đây là, nếu chúng ta muốn việt hóa các câu thông báo lỗi khi xuất ra thì làm thế nào?

3. Việt hóa language form validation

Bước Việt hóa thật ra là chúng ta sửa code trong phần system của CI. Rất đơn giản, tại folder system, các bạn tìm đến folder language, mở nó ra và tìm file form_validation_lang.php, mở file này ra và dịch các đoạn tiếng Anh ở bên trong bên. Chú ý ký hiệu **%s** chính là tên mà chúng ta đặt khi nó xuất ra thông báo lỗi.

```
<?php
$lang['required']       = "The %s field is required.";
$lang['isset']          = "The %s field must have a value.";
$lang['valid_email'] = "The %s field must contain a valid email address.";
$lang['valid_emails']= "The %s field must contain all valid email
addresses.";
$lang['valid_url']      = "The %s field must contain a valid URL.";
$lang['valid_ip']       = "The %s field must contain a valid IP.";
$lang['min_length']= "The %s field must be at least %s characters in
length.";
$lang['max_length']= "The %s field can not exceed %s characters in length.";
$lang['exact_length']="The %s field must be exactly %s characters in
length.";
$lang['alpha']  = "The %s field may only contain alphabetical characters.";
$lang['alpha_numeric']= "The %s field may only contain alpha-numeric
characters.";
$lang['alpha_dash']     = "The %s field may only contain alpha-numeric
characters, underscores, and dashes.";
$lang['numeric']        = "The %s field must contain only numbers.";
$lang['is_numeric']  = "The %s field must contain only numeric characters.";
$lang['integer']        = "The %s field must contain an integer.";
$lang['regex_match']        = "The %s field is not in the correct format.";
$lang['matches']        = "The %s field does not match the %s field.";
$lang['is_unique']      = "The %s field must contain a unique value.";
$lang['is_natural']     = "The %s field must contain only positive numbers.";
$lang['is_natural_no_zero'] = "The %s field must contain a number greater
than zero.";
$lang['decimal']        = "The %s field must contain a decimal number.";
$lang['less_than']      = "The %s field must contain a number less than %s.";
$lang['greater_than']="The %s field must contain a number greater than %s.";
?>
```