**Chương 8. Input validation**

1. **Giới thiệu**

* Một ứng dụng web tốt phải đảm bảo dữ liệu người dùng nhập vào là hợp lệ. Ví dụ, bạn muốn kiểm tra thông tin người dùng được nhập vào ở form và thông tin này sẽ lưu xuống cơ sở dữ liệu, kiểm tra password nhập vào ít hơn n kí tự và ngày sinh phải là ngày nhỏ hơn ngày hiện tại.
* Trong Struts thì kiểm tra dữ liệu nhập sẽ dễ dàng với built-in validators, dựa trên Xwork Validation Framework. Bạn phải khai báo một file XML để kiểm tra dữ liệu nhập, phải khai báo rõ rãng những dữ liệu nhập kiểm tra, và gửi message đến browser nếu sai dữ liệu nhập.
* Có 2 loại validator: field validators và plain validators (non-field validators).
* Field validators is associated trên 1 form và hoạt động bằng cách kiểm tra lại dữ liệu trước khi dữ liệu này tác động vào thuộc tính action. Hầu hết các validators được dùng là field validators.
* Plain validators thì không associated một vùng mà nó được dùng để kiểm tra một điều kiện khi gặp.
* Sử dụng validator gồm 3 bước:
  + 1. Determine action mà có dữ liệu nhập cần kiểm tra (validator)
  + 2. Viết một file cấu hình validator, file này phải theo một trong 2 dạng sau:
    - ActionClass-validation.xml
    - ActionClass-alias-validation.xml
  + 3. Determine nơi mà user được trỏ đến khi validation thất bại bằng cách định nghĩa một <result name=”input”> trong struts.xml. Thông thường, giá trị của thẻ result là file JSP có xài validation form.

Chú ý trong việc đăng kí Validator:

Tất cả bundled validators được đăng kí bằng cách mặc định. Việc đăng kí giống như viết một Issue, nếu bạn xài các loại custom validator.

1. **Cấu hình validator**

* Để cấu hình validators, bạn phải viết một file cấu hình, là dạng XML theo chuẩn Xwork validator DTD
* Một file cấu hình validator luôn bắt đấu với trạng thái DOCTYPE

<!DOCTYPE validators PUBLIC

“-//OpenSymphony Group//Xwork Validator 1.0.2//EN”

“http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>

* Thẻ đầu tiên trong file cấu hình validator là <validators>, nó có thể có bất kì các thẻ field và các thẻ validator con. Một thẻ field đại diện một form field mà có một hay nhiều trường validators cần kiểm tra. Một thẻ validator đại diện một plain validator. Sau đây là cấu trúc cơ bản của một file cấu hình validator

<!DOCTYPE validators PUBLIC

“-//OpenSymphony Group//Xwork Validator 1.0.2//EN”

“http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>

<validators>

<field name =”…”>

…

</field>

<field name =”…”>

…

</field>

….

<validator type=”…”>

…

</validator>

<validator type=”…”>

…

</validator>

….

</validators>

* Thuộc tính name trong thẻ field chỉ rõ form field được validated
* Bạn có thể áp dụng bất kì nhiều điều kiện kiểm tra (validators) cho một form field bằng cách sử dụng các thẻ field-validator. Ví dụ, thẻ field chỉ rõ cái field userEmail phải được kiểm tra là bắt buộc có và là dạng email

<field name=”userEmail”>

<field-validator type=”required”>

</field-validator>

<field-validator type=”email”>

</field-validator>

</field>

* Thẻ field-validator phải có thuộc tính type, là thuộc tính chỉ rõ validator. Ngoài ra, thẻ này còn có thuộc tính short-circuit, thuộc tính này mang giá trị true hoặc false. Giá trị true chỉ rõ là nếu validator hiện thời thất bại, thì validators kế tiếp sẽ không thực thi. Ví dụ, nếu validator yêu cầu phải có bị thất bại, thì validator email sẽ được thực thi

<field name=”userEmail”>

<field-validator type=”required” short-circuit=”true”>

</field-validator>

<field-validator type=”email”>

</field-validator>

</field>

* Bạn có thể truyền thêm tham số vào một validator bằng thẻ param. Bạn cũng có thể định nghĩa một message error validation bằng cách sử dụng thẻ message. Ví dụ sau đây, sẽ kiểm tra độ dài chuỗi của field, gồm 2 tham số minLength và maxlength, và message error phải được hiện ra khi validation thất bại.

<field-validator type=”stringlength”>

<param name=”minLength>6</param>

<param name=”maxLength>14</param>

<message>

User name must be between 6 and 14 character long

</message>

</field-validator>

* Thẻ field-validator có thể có không hoặc nhiều thẻ param và co duy nhất một thẻ message
* Thẻ validator dùng để đại diện một plain validator. Nó có thể chứa nhiều thẻ param, và một thẻ message. Ví dụ, thẻ validator sau đây chỉ rõ max field phải lớn hơn min field

<validator type=”expression”>

<param name=”expression”>

max > min

</param>

<message>

Maximum temperature must be greater than Minimum temperature

</message>

</validator>

* Cũng như thẻ field validator, thẻ validator phải có thuộc tính type và có thể có thuộc tính short circuit attribute

1. **Bundleds validators – Các gói validators**

* Struts tích hợp các validators sau:
  + requrired validator
  + requiredstring validator
  + int validator
  + date validator
  + expression validator
  + fieldexpression validator
  + email validator
  + url validator
  + visitor validator
  + conversion validator
  + stringlength validator
  + regex validator

1. **required validator**

* validator này giúp đảm bảo giá trị trong field không null.
* Ví dụ, class RequiredTestAction ở Listing 8.1 có 2 thuộc tính, là userName và password, và cấu hình validator ở Listing 8.2

**Listing 8.1 class RequiredTestAction**

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class RequiredTestAction extends ActionSupport {  private String userName;  private String password;  //Các phương thức getter và setter không được hiển thị ở đây  } |

**Listing 8.2. RequiredTestAction-validation.xml**

|  |
| --- |
| <!DOCTYPE validators PUBLIC  “-//OpenSymphony Group//Xwork Validator 1.0.2//EN”  “http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>  <validators>  <field name=”userName”>  <field-validator type=”required”>  <message>Please enter a user name</message>  </field-validator>  </field name=”password”>  <field-validator type=”required”>  <message>Please enter a password</message>  </field-validator>  <field name=”password”>  </field>  </validators> |

**Listing 8.3 Trang Required.jsp**

|  |
| --- |
| <%@ taglib prefix=”s” uri=”/struts-tags” %>  <html>  <head>  <title>required Validator Example</title>  <style type=”text/css”>@import url(css/main.css);</style>  </head>  <style>.errorMessage{color:redl}</style>  <body>  <div id=”global” style=”width:350px”>  <h3>Enter user name and password</h3>  <s:fielderror/>  <s:form action=”Required2”>  <%-- <s:textfield name=”userName” label=”User name”/>  ---%>  <s:password name=”password” label=”Passord”/>  <s:submit/>  </s:form>  </div>  </body>  </html> |

* Hình 8.1 thể hiện form sau validation thất bại khi bỏ trống ô userName



1. **requiredstring validator**

* requriedstring validator đảm bảo giá trị một field không null hoặc trống. Nó có tham số trim, mặc định là true. Nếu trim là true, thì vùng validate sẽ cắt bỏ các khoảng trắng trước khi validation. Nếu trim là false, giá trị validate sẽ không cắt bỏ khoảng trắng. Tham số trim được mô tả trong bảng 8.1
* Ví dụ, class RequiredStringTestAction ở Listing 8.4 có thuộc tính cần validate là userName và password. File cấu hình validation ở Listing 8.5 sẽ yêu cầu kiểm tra requiredstring với field userName và password

**Listing 8.4 Class RequiredStringTestAction**

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class StringLengthTestAction extends ActionSupport {  private String userName;  private String password;  //Các phương thức getter và setter không hiển thị ở đây  } |

**Listing 8.5 RequiredStringTestAction-validation.xml**

|  |
| --- |
| <!DOCTYPE validators PUBLIC  “-//OpenSymphony Group//Xwork Validator 1.0.2//EN”  “http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>  <validators>  <field name=”userName”>  <field-validator type=”requiredstring”>  <param name=”trim”>true</param>  <message>Please enter a user name</message>  </field-validator>  </field name=”password”>  <field-validator type=”requiredstring”>  <param name=”trim”>false</param>  <message>Please enter a password</message>  </field-validator>  <field name=”password”>  </field>  </validators> |

* Khi đó, kết quả ở trang JSP ta sẽ được khi gõ space vào ô UserName và bỏ trống ô password



1. **stringlength validator**

* stringlength được dùng để validate độ dài giá trị của một field không trống, bạn có thể khai báo độ dài minimum và maximum thông qua 2 tham số minLength và maxLength.
* Ví dụ, ở class StringLengthTestAction ở Listing 8.7 định nghĩa 2 thuộc tính là userName và password. Một user name phải có độ dài kí tự từ 6 đến 14. File cấu hình validator cho ví dụ trên ở Listing 8.8. Và ở trang StringLength.jsp ở Listing 8.9 thể hiện form có field được map với thuộc tính userName

Listing8.7 class StringLengthTestAction

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class RequiredStringTestAction extends ActionSupport {  private String userName;  private String password;  //Các phương thức getter và setter không hiển thị ở đây  } |

Listing 8.8 File StringLengthTestAction-validation.xml

|  |
| --- |
| <!DOCTYPE validators PUBLIC  “-//OpenSymphony Group//Xwork Validator 1.0.2//EN”  “http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>  <validators>  <field name=”userName”>  <field-validator type=”stringlength”>  <param name=”minLength”>4</param>  <param name=”maxLength”>16</param>  <message>  User name must be between 6 and 14 characters long  </message>  </field-validator>  </field>  </validators> |

Listing 8.9 Trang StringLength.jsp

|  |
| --- |
| <%@ taglib prefix=”s” uri=”/struts-tags” %>  <html>  <head>  <title>stringlength Validator Example</title>  <style type=”text/css”>@import url(css/main.css);</style>  </head>  <style>.errorMessage{color:redl}</style>  <body>  <div id=”global” style=”width:480px”>  <h3>Select a user name</h3>  <s:form action=”StringLength2”>  <s:textfield name=”userName”  label=”User name (6-14 characters)”/>  <s:submit/>  </s:form>  </div>  </body>  </html> |

* Khi đó ta được kết quả:



1. **int validator**

* int validator dùng để kiểm tra giá trị một field có thể chuyển qua thành kiểu int, hoặc kiểm tra giá trị này với 2 tham số min và max.
* Ví dụ, class IntTestAction ở Listing 8.10, có một thuộc tính là year

Listing 8.10 Class IntTestAction

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class IntTestAction extends ActionSupport {  private int year;  // Các phương thức getter và setter không thể hiện ở đây  } |

* File cấu hình validator ở Listing 8.11 sẽ kiểm tra giá trị year được submit lên phải có giá trị từ 1990 đến 2009.

Listing 8.11 File IntTestAction-validation.xml

|  |
| --- |
| <!DOCTYPE validators PUBLIC  “-//OpenSymphony Group//Xwork Validator 1.0.2//EN”  “http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>  <validators>  <field-validator type=”int”>  <param name=”min”>1990</param>  <param name=”max”>2009</param>  <message>Year must be between 1990 and 2009</message>  </field-validator>  </validators> |

* Trang Int.jsp ở Listing 8.12 thể hiện một form với một tag textfield đặt tên là year, và khi form được submit, validator sẽ kiểm tra đảm bảo giá trị year thuộc phạm vi 1990 đến 2009

Listing 8.12 Trang In.jsp

|  |
| --- |
| …  <div id=”global” style=”width:350px”>  <h3>Enter a year</h3>  <s:form action=”Int2”>  <s:textfield name=”year” label=”Year (1990-2009)”/>  <s:submit/>  </s:form>  </div>  … |

* Khi đó, ở trang Int.jsp ta chỉ nhập được giá trị của year từ 1990-2009



1. **date validator**

* validator này sẽ kiểm tra field date có nằm trong một phạm vi nhất định hay không, thông qua 2 tham số min và max
  + Chú ý: định dạng ngày tháng được dùng để kiểm tra ngày tháng phụ thuộc vào vị trí hiện tại (current locale).
* Ví dụ, class DateTestAction, ở Listing 8.13, dùng date validator trong file cấu hình DateTestAction-validation.xml ở Listing 8.14. Và Listing 8.15 sẽ có field là birthday dùng để submit date nhập vào.

Listing 8.13 class DateTestAction

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class DateTestAction extends ActionSupport {  private Date birthDate;  // Các phương thức getter và setter không thể hiện ở đây  } |

Listing 8.14 File DateTestAction-validation.xml

|  |
| --- |
| <!DOCTYPE validators PUBLIC  “-//OpenSymphony Group//Xwork Validator 1.0.2//EN”  “http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd”>  <validators>  <field name=”birthDate”>  <field-validator type=”date”>  <param name=”max”>1/1/2000</param>  <message>  You must have been born before the year 200 to register  </message>  </field-validator>  </field>  </validators> |

* File cấu hình đã chỉ rõ giá trị của năm phải trước ngày 1 tháng 1 năm 2000. Định dạng date được sữ dụng ở đây là US\_en.

Listing 8.15 Trang Date.jsp

|  |
| --- |
| …  <div id=”global” style=”width:350px”>  <h3>Enter your bithdate</h3>  <s:form action=”Date2”>  <s:textfield name=”birthDate” label=”Bith Date”/>  <s:submit/>  </s:form>  </div>  … |

* Khi đó, ta sẽ được một form cho nhập ngày tháng và kiểm tra ngày tháng đó phải trước ngày 1-1-2000.



1. **email validator**

* email validator dùng để kiểm tra một chuỗi có phải là một địa chỉ email hay không. Validator này dùng Java regular Expression API và theo mẫu sau:

"\\b(^[\_A-Za-z0-9-]+(\\.[\_A-Za-z0-9-]+)\*@([A-Za-z0-9-])+(\\.[A-Za-z0-9-]+)\*((\\.[A-Za-z0-9]{2,})|(\\.[A-Za-z0-9]{2,}\\.[A-Za-z0-9]{2,}))$)\\b"

* Cấu trúc trên có nghĩa một email có thể bắt đầu bằng sự kết hợp của kí tự và số, và phải có kí tự @ để xác định rõ hostname
* Ví dụ, class EmailTestAction ở Listing 8.16 có thuộc tính email. File cấu hình validator ở Listing 8.17 sử dụng email validator để kiểm tra thuộc tính này.

Listing 8.16. The EmailTestAction class

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class EmailTestAction extends ActionSupport  {  private String email;  //getter and setter not shown  } |

Listing 8.17. The EmailTestAction-validation.xml file

|  |
| --- |
| <!DOCTYPE validators PUBLIC "-//OpenSymphony Group//XWork Validator 1.0.2//EN" "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">  <validators>  <field name="email">  <field-validator type="email">  <message>Invalid email</message>  </field-validator> \  </field>  </validators> |

Listing 8.18. The Email.jsp page

|  |
| --- |
| …  <div id="global" style="width:350px">  <h3>Enter your email</h3>  <s:form action="Email2">  <s:textfield name="email" label="Email"/>  <s:submit/>  </s:form>  </div>  … |

* Sau đó ta được kết quả như sau:



1. **url validator**

* url validator được dùng để kiểm tra một chuỗi có phải là URL hay không. Validator này hoạt động dựa trên việc tạo một đối tượng java.net.URL sử dụng chuỗi này. Nếu không xảy ra lỗi gì khi xử lý, validation là thành công.
* Các URL hợp lệ:
  + http://www.google.com
  + <https://hotmail.com>
  + <ftp://yahoo.com>
  + <file:///C:/data/V3.doc>
* Một trường hợp không hợp lệ vì không có protocol:
  + java.com
* Ví dụ, class UrlTestAction ở Listing 8.19 có một thuộc tính url. File cấu hình validation ở Listing 8.20

Listing 8.19. The UrlTestAction class

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class UrlTestAction extends ActionSupport  {  private String url;  // getter and setter not shown  } |

Listing 8.20. The UrlTestAction-validation.xml file

|  |
| --- |
| <!DOCTYPE validators PUBLIC "-  //OpenSymphony Group//XWork Validator 1.0.2//EN" "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">  <validators>  <field name="url">  <field-validator type="url">  <message>Invalid URL</message>  </field-validator>  </field>  </validators> |

Listing 8.21. The Url.jsp page

|  |
| --- |
| …  <div id="global" style="width:350px">  <h3>What is your website?</h3>  <s:form action="Url2">  <s:textfield name="url" label="URL" size="40"/>  <s:submit/>  </s:form>  </div>  … |

* Kết quả ta được như sau



1. **regex validator**

* Validator này kiểm tra một giá trị với dạng mẫu đã được khai báo. Nó gồm các parameter được thống kê trong Table 8.5.

Table 8.5

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| expression\* | String | Dạng mẫu để so sánh |
| caseSensitive | boolean | Indicates whether or not the matching should be done in a case sensitive way. The default value is true. |
| trim | boolean | Có cắt bỏ khoảng trắng trước khi validation hay không. Mặc định là true |

1. **expression and fieldexpression Validators**

* The expression và fieldexpression validators đều được dùng để validate lại một field thông qua biểu thức OGNL.
* Cả 2 validators này tương tự nhau, chỉ khác là expression validator sẽ sinh ra một action error, còn fieldexpression sẽ sinh ra một field error khi validation thất bại. Tham số của cả 2 validation này ở Table 8.6

Table 8.6

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Description |
| expression\* | String | Biểu thức OGNL chỉ rõ cách xử lý validation |

* Ví dụ 1: The expression Validator Example
* Class ExpressionTestAction ở Listing 8.22 có 2 thuộc tính min và max. Listing 8.23 là file cấu hình validator có sử dụng expression validator và chỉ rõ giá trị thuộc tính max phải lơn hơn min. Listing 8.24 thể hiện JSP gồm form có 2 field.

Listing 8.22. The ExpressionTestAction class

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class ExpressionTestAction extends ActionSupport  {  private int min;  private int max;  // getters and setters not shown  } |

Listing 8.23. The ExpressionTestAction-validation.xml file

|  |
| --- |
| <!DOCTYPE validators PUBLIC  "-//OpenSymphony Group//XWork Validator 1.0.2//EN" "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd"> <validators>  <validator type="expression">  <param name="expression"> max > min </param>  <message>  Maximum temperature must be greater than Minimum temperature  </message>  </validator>  </validators> |

Listing 8.24. The Expression.jsp page

|  |
| --- |
| …  <div id="global" style="width:400px">  <s:actionerror/>  <h3>Enter the minimum and maximum temperatures</h3>  <s:form action="Expression2">  <s:textfield name="min" label="Minimum temperature"/>  <s:textfield name="max" label="Maximum temperature"/>  <s:submit/>  </s:form>  </div>  … |

* Kết quả ta được như sau:



* Ví dụ 2: The fieldexpression Validator Example
* Tương tự Class FieldExpressionTestAction ở Listing 8.25 có 2 thuộc tính là min và max. File cấu hình validator ở Listing 8.26 chỉ rõ biểu thức OGNL cho fieldexpression validator. Listing 8.27 thể hiện trang JSP trong ví dụ này.

Listing 8.25. The FieldExpressionTestAction class

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class FieldExpressionTestAction extends ActionSupport {  private int min;  private int max;  // getters and setters not shown  } |

Listing 8.26. The FieldExpressionTestAction-validation.xml file

|  |
| --- |
| <!DOCTYPE validators PUBLIC  "-//OpenSymphony Group//XWork Validator 1.0.2//EN" "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">  <validators>  <field name="max">  <field-validator type="fieldexpression">  <param name="expression"> max > min </param>  <message>  Maximum temperature must be greater than Minimum temperature  </message>  </field-validator>  </field>  </validators> |

. Listing 8.27. The FieldExpression.jsp page

|  |
| --- |
| …  <div id="global" style="width:400px">  <h3>Enter the minimum and maximum temperatures</h3>  <s:form action="FieldExpression2">  <s:textfield name="min" label="Minimum temperature"/>  <s:textfield name="max" label="Maximum temperature"/>  <s:submit/>  </s:form>  </div>  … |

* Kết quả ta được:



1. **conversion validator**

* conversion validator sẽ thông báo một conversion error message cho bạn khi chuyển đổi dạng một thuộc tính từ form về action sai kiểu. Validator này cũng cho phép bạn thêm một message ở conversion error message. Sau đây là message mặc định của một conversion error:

Invalid field value for field "**fieldName**".

* Bạn có thể add thêm message của mình:

Invalid field value for field "**fieldName**".

[**Your custom message**]

* Ví dụ, class ConversionTestAction ở Listing 8.28 có mộ thuộc tính age dạng integer. File cấu hình validator ở Listing 8.29 sử dụng conversion validator cho age field và add error message khi conversion thất bại.

Listing 8.28. The ConversionTestAction class

|  |
| --- |
| package app08a;  import com.opensymphony.xwork2.ActionSupport;  public class ConversionTestAction extends ActionSupport  {  private int age;  // getter and setter deleted  } |

Listing 8.29. The ConversionTestAction-validation.xml file

|  |
| --- |
| <!DOCTYPE validators PUBLIC  "-//OpenSymphony Group//XWork Validator 1.0.2//EN"  "http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">  <validators>  <field name="age">  <field-validator type="conversion">  <message>  An age must be an integer.  </message>  </field-validator>  </field>  </validators> |

Listing 8.30. The Conversion.jsp page

|  |
| --- |
| …  <div id="global" style="width:350px">  <h3>Enter your age</h3>  <s:form action="Conversion2">  <s:textfield name="age" label="Age"/>  <s:submit/>  </s:form>  </div>  … |

* Kết quả ta được:



1. **visitor validator**

* visitor validator được tái sử dụng nhiều cấp bậc, cho phép bạn sử dụng file cấu hình validator cho nhiều action khác. Ví dụ một trường hợp sau đây:
  + Giả sử bạn có một class action là Customer, có một thuộc tính là address kiểu Address.
  + Kiểu Address gồm 5 thuộc tính sau: streetName, streetNumber, city, state, và zipCode.
  + Để validate thuộc tính zipCode ở trong đối tượng Address, mà address là thuộc tính của Customer, bạn phải viết thẻ field trong file Customer-validation.xml

<field name="address.zipCode">

<field-validator type="requiredstring">

<message>Zip Code must not be empty</message>

</field-validator>

</field>

* + Giả sử bạn có class Employee action cũng sử dụng thuộc tính address dạng Address. Nếu thuộc tính address của Employee yêu cầu các validation giống address của Customer, bạn có thể có một file Employee-validation.xml giống hệt như Customer-validation.xml
  + Điều này không cần thiết, visitor validator giúp bạn phân chia lại các validation có qui tắc giống nhau vào một file. Mỗi khi bạn cần sử dụng các qui tắc validation này, bạn chỉ cần gọi file một cách đơn giản. trong ví dụ này, bạn cần phân chia các qui tắc validation của Address clashh thành Address-validation.xml. Sau đó, ở file Customer-validation.xml bạn viết như sau:

<field name="address">

<field-validator type="visitor">

<message>Address: </message>

</field-validator>

</field>

* + Thẻ field có name là address, sử dụng file validation

This **field** element says, for the **address** property, use the validation file that comes with the property type (**Address**). In other words, Struts would use the **Address-validation.xml** file for validating the **address** property. If you use **Address** in multiple action classes, you don't need to write the same validation rules in every validator configuration file for each action.