

## Apache Struts: Validating Input

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
 More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
 Servlet/JSP/Struts/JSP Training: [courses.coreservlets.com](http://courses.coreservlets.com)  
 Slides © Marty Hall, <http://www.coreservlets.com>, books © Sun Microsystems Press

## Overview

- Distinguishing manual validation from automatic validation
- Distinguishing client-side validation from server-side validation
- Using automatic validation
  - Declare application-wide properties file
  - Add messages to properties file
  - Turn on the automatic validator
  - Put validation rules in validation.xml
  - Put <html:errors/> in input page
  - Enable JavaScript validation

Apache Struts Lecture 3: Using the Validator [www.coreservlets.com](http://www.coreservlets.com)

## Options for Form Field Validation

- Do validation in the Action
  - Most flexible; can use context
  - May require repetition in multiple Actions
  - Must manually map conditions back to input page
  - Must write validation rules yourself
- Do validation in the form bean
  - In individual setter methods
    - Not really validation, but can be used to modify values
  - Using the validate method
    - Not quite as flexible
    - Does not require repetition in multiple Actions
    - Will automatically redisplay input page
    - Still requires you to write validation rules yourself
- Use automatic validator

Apache Struts Lecture 3: Using the Validator [www.coreservlets.com](http://www.coreservlets.com)

## Manual Validation

- Option 1: Put validation code in the Action
  - Return custom conditions from Action
  - Map certain conditions back to the input form
  - Embed the messages in the form bean
- Option 2: Put validation code in bean
  - Insert <html:errors/> in input form

```

public ActionErrors validate(ActionMapping mapping,
    HttpServletRequest request) {
    ActionErrors errors = super.validate(mapping, request);
    if (errors == null) errors = new ActionErrors();
    if (someField == null || someField.trim().equals("")) {
        errors.add("someField",
            new ActionError("errors.someField"));
    }
    return errors;
}

```

Apache Struts Lecture 3: Using the Validator [www.coreservlets.com](http://www.coreservlets.com)

## Manual vs. Automatic Validation

- Manual validation
  - Most flexible
  - Repeats same logic many times
  - Runs on server only if you use existing framework
    - Client-side validation requires writing lots of JavaScript
  - Tedious
  - Embedded in Java code
    - Which violates Struts strategy of having as much as possible in editable XML files
- Automatic validation
  - Consolidates validation code
  - Lets you use standard validation rules
  - Runs on server; can optionally also run on client
  - Described by XML files

Apache Struts Lecture 3: Using the Validator [www.coreservlets.com](http://www.coreservlets.com)

## Client-Side vs. Server-Side Validation

- Client-side validation
  - JavaScript code verifies format of fields
  - Dialog box warns users of illegal values
  - Submission blocked if invalid
  - Pro:
    - Fast
  - Cons:
    - Can be deliberately or accidentally bypassed
    - Cannot do validation that requires much application logic
- Server-side validation
  - Java code on server verifies format of fields
  - Form is redisplayed (with warnings) if illegal values
  - You *must* do this regardless of whether or not you do client-side validation!

Apache Struts Lecture 3: Using the Validator [www.coreservlets.com](http://www.coreservlets.com)

### Using Automatic Validation

- **Declare application-wide properties file**
  - Use message-resources in struts-config.xml. Eg:
    - <message-resources parameter="resources.application" null="false" />
    - Refers to WEB-INF/classes/resources/application.properties
- **Add messages to properties file**
  - Eg: inputForm.firstName=First name
- **Turn on the automatic validator**
  - Use plug-in in struts-config.xml. Eg:
 

```
<plug-in
  className="org.apache.struts.validator.ValidatorPlugIn">
  <set-property property="pathnames"
    value="/WEB-INF/validator-rules.xml,
      /WEB-INF/validation.xml"/>
</plug-in>
```

www.coreservlets.com

### Using Automatic Validation (Continued)

- **Put validation rules in validation.xml**
  - Eg:
 

```
<formset>
  <form name="orderFormBean">
    <field property="firstName"
      depends="required">
      <arg0 key="inputForm.firstName"/>
    </field>
  </form>
  ...
</formset>
```

www.coreservlets.com

### Using Automatic Validation (Continued)


- **Have your form bean extend ValidatorForm, not ActionForm directly**

```
import org.apache.struts.validator.*;

public class OrderFormBean
  extends ValidatorForm { ... }
```
- **Put <html:errors/> in input page**
  - Edit properties file to customize form of error message
- **Enable JavaScript validation**
  - Add <html:javascript> anywhere in form
  - Add onsubmit="return validateBeanName(this);" to html:form

www.coreservlets.com

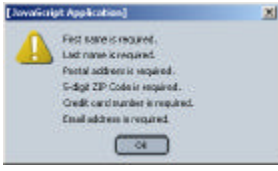
### Using Automatic Validation: Example (Goal)



www.coreservlets.com

### Using Automatic Validation: Example (Goal)


- **JavaScript validation enabled**



www.coreservlets.com

### Using Automatic Validation: Example (Goal)

- **JavaScript validation disabled**



www.coreservlets.com

## Using Automatic Validation: Example (Goal)



13 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Implementing the Example (Background)

```
<form-beans>
  <form-bean name="orderFormBean"
             type="coreservlets.OrderFormBean"/>
</form-beans>

<action-mappings>
  <action path="/actions/order"
         type="coreservlets.Order"
         name="orderFormBean"
         scope="session"
         input="/forms/order-form.jsp">
    <forward name="success"
            path="/WEB-INF/results/order-confirmation.jsp"/>
  </action>
</action-mappings>
```

14 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Implementing the Example (Background)

```
package coreservlets;

import javax.servlet.http.*;
import org.apache.struts.action.*;

public class Order extends Action {
  public ActionForward execute(ActionMapping mapping,
                              ActionForm form,
                              HttpServletRequest request,
                              HttpServletResponse response)
    throws Exception {
    return(mapping.findForward("success"));
  }
}
```

15 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 1: Declare Application- wide Properties File

- In `struts-config.xml` (below `action-mappings`):
  - `<message-resources parameter="resources.application"/>`
  - Refers to `WEB-INF/classes/resources/application.properties`

16 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 2: Add Messages to Properties File

- `WEB-INF/classes/resources/application.properties`

```
# -- Custom messages for this application --
inputForm.firstName=First name
inputForm.lastName=Last name
inputForm.address=Postal address
inputForm.zipCode=5-digit ZIP Code
inputForm.creditCardNumber=Credit card number
inputForm.email=Email address
# -- standard errors --
errors.header=<FONT COLOR=RED><UL>
errors.prefix=<LI>
errors.suffix=</LI>
errors.footer=</UL></FONT>
# -- validator --
...
errors.required={0} is required.
errors.email={0} is an invalid e-mail address.
```

17 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 3: Turn on the Automatic Validator

- At bottom of `struts-config.xml`:
 

```
<plug-in className="org.apache.struts.validator.ValidatorPlugIn">
  <set-property
    property="pathnames"
    value="/WEB-INF/validator-rules.xml,/WEB-INF/validation.xml"/>
</plug-in>
```

18 Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 4: Put Validation Rules in validation.xml

- **<form-validation>** and **<formset>**
  - Main enclosing elements
- **<form name="beanName">**
  - Matches form-bean name from struts-config.xml
- **<field property="firstName" depends="required">**
  - Matches HTML form property name
- **depends="required">**
  - Matches name of predefined validator rule
    - **required:** must be non-empty
    - **mask:** must match a given regular expression
    - **email:** must be an email address
    - **creditCard:** must be a legal credit card number
      - (Use 41111111111111 for testing)
- **<arg0 key="property.subname"/>**
  - Replaces {0} in error message from properties file

16

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 4: Put Validation Rules in validation.xml

```
<form-validation>
  <formset>
    <form name="orderFormBean">
      <field property="firstName"
        depends="required">
        <arg0 key="inputForm.firstName"/>
      </field>
      ...
      <field property="zipCode"
        depends="required,mask">
        <arg0 key="inputForm.zipCode"/>
        <arg1 key="inputForm.zipCode"/>
        <var>
          <var-name>mask</var-name>
          <var-value>^\d{5}\d*</var-value>
        </var>
      </field>
    </form>
  </formset>
</form-validation>
```

20

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Step 5: Have Form Bean Extend ValidatorForm

```
package coreservlets;

import org.apache.struts.validator.*;

public class OrderFormBean extends ValidatorForm {
    private String firstName = "";
    ...
    public String getFirstName() {
        return(firstName);
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    ...
}
```

21

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Steps 6 & 7: Output <html:errors/> and Enable JavaScript validation

```
...
<%@ taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<html:errors/>
<html:form action="/actions/order"
  onsubmit="return validateOrderFormBean(this);">
  First name: <html:text property="firstName"/><BR>
  Last name: <html:text property="lastName"/><BR>
  Mailing address: <html:text property="address"/><BR>
  ZIP Code: <html:text property="zipCode"/><BR>
  Credit Card Number:
  <html:text property="creditCardNumber"/><BR>
  Email address for confirmation:
  <html:text property="email"/><BR>
  <html:submit value="Order Now!"/>
</html:form>
<html:javascript formName="orderFormBean"/>
...
```

22

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Other Validator Capabilities

- **More predefined validators**
  - required
  - mask
  - range
  - maxLength
  - minLength
  - integer, float, double, long, short, byte
  - date
  - creditCard
  - email
- **Rolling your own pluggable validators**
  - For example, one field may depend on another
  - You can assign both server-side and client-side code
  - Declare validators in validator-rules.xml

23

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com

## Summary

- Declare application-wide properties file
- Add messages to properties file
- Turn on the automatic validator
- Put validation rules in validation.xml
- Put <html:errors/> in input page
- Enable JavaScript validation

24

Apache Struts Lecture 3: Using the Validator

www.coreservlets.com



**Questions?**

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
Servlet and JSP Training Courses: [courses.coreservlets.com](http://courses.coreservlets.com)

Slides © Marty Hall, <http://www.coreservlets.com>, books © Sun Microsystems Press