


Validating Input



Struts University Series




Validating Input

- Do we need to validate input?
 - Can a validation framework help?
 - Does validation affect workflow?
 - Can we code custom Validators?
- 



Validating Input

- Manual Validation
 - Validation Framework
 - Validation Workflow
 - Coding Validators
- 



Register Properties

```
private String username = null;
```

```
public String getUsername() {  
    return username;  
}
```


```
public void setUsername(String value) {  
    username = value;  
}
```

```
private String password = null;
```






Register Properties

- | | |
|------------------|--------------------------|
| ● Username | ● <i>Required</i> |
| ● Password | ● <i>Required</i> |
| ● Password2 | ● <i>== Password</i> |
| ● FullName | ● <i>Required</i> |
| ● FromAddress | ● <i>Required, Email</i> |
| ● ReplyToAddress | ● <i>Email</i> |
- 



Register Form

```
<s:actionerror/>
<s:form>
  <s:textfield label="Username" name="username"/>
  <s:password label="Password" name="password"/>
  <s:password label="(Repeat) Password" name="password2"/>
  <s:textfield label="Full Name" name="fullName"/>
  <s:textfield label="From Address" name="fromAddress"/>
  <s:textfield label="Reply To Address" name="replyToAddress"/>
  <s:submit value="Save" name="Save"/>
</s:form>
```




Register Action

```
public String execute()  
    throws Exception {  
    User user = findUser(getUsername(), getPassword());  
    boolean haveUser = (user != null);  
    if (haveUser) {  
        addActionError(ERROR_USERNAME_UNIQUE);  
        return INPUT;  
    }  
    createUser(getUsername(), getPassword());  
    saveUser();  
    return SUCCESS;  
}
```



MailReaderSupport

```
public User createUser(String username, String password)
    throws Exception {
    UserDatabase database = getDatabase();
    User user = database.findUser(username);
    if (user != null) {
        addActionError(ERROR_USERNAME_UNIQUE);
        return null;
    }
    user = database.createUser(username);
    BeanUtils.setValues(user, this, null);
    return user;
}
```





Manual Validation

- Validating in the execute method
 - Implementing the Validateable Interface
- 

execute method


```
public String execute()  
    throws Exception {  
    String username = getUsername();  
    if ((username==null) ||  
        (username.trim().equals("")) ) {  
        addFieldError("username",  
            "You must enter a first name");  
    }  
    // ... other properties  
    if (hasErrors())  
        return INPUT;  
    createUser(getUsername(), getPassword());  
    saveUser();  
    return Action.SUCCESS;  
}
```



validate method

```
public void validate() throws Exception {
    String username = getUsername();
    if ((username==null) ||
        (username.trim().equals(""))) {
        addFieldError("username",
            "You must enter a first name");
    }
}
```

```
public String execute() throws Exception {
    createUser(getUsername(), getPassword());
    saveUser();
    return SUCCESS;
}
```

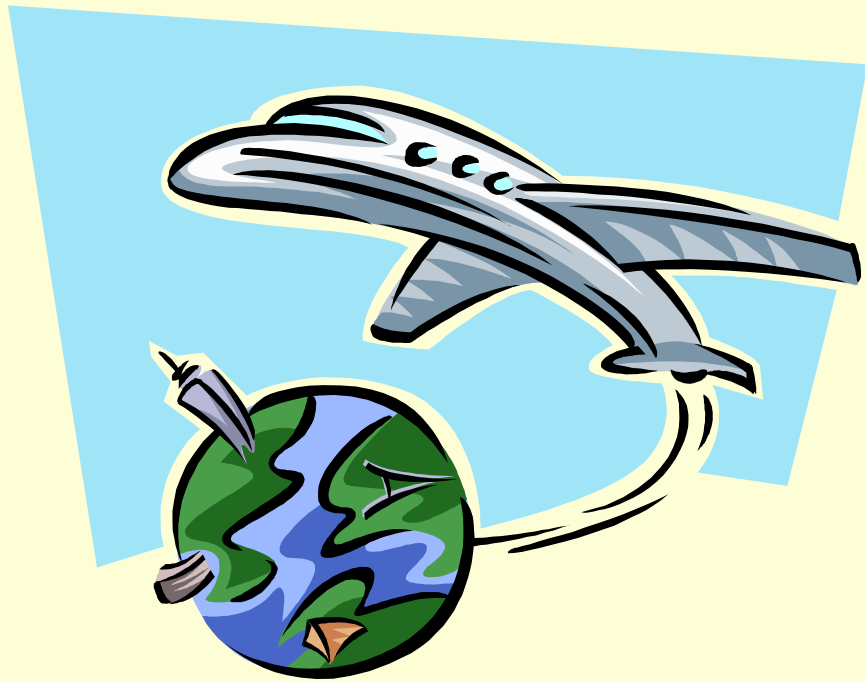




Validateable Logic


- Invoke validate method
 - Has Errors?
 - Yes: return INPUT
 - No: continue ... and ... invoke execute







Review

- Validating in the `*****` method
 - Implementing the `*****` Interface
 - Invoke validate method
 - Has Errors?
 - Yes: return `*****`
 - No: continue ... and ... invoke execute
- 




Review

- Validating in the execute method
- Implementing the Validateable Interface
- Invoke validate method
 - Has Errors?
 - Yes: return INPUT
 - No: continue ... and ... invoke execute






Validating Input

- Manual Validation
 - Validation Framework
 - Validation Workflow
 - Coding Validators
- 



Validation Framework

- The *ClassName-validation.xml* file
 - Bundled Validators
 - Registering Validators
 - Some Validation Examples
- 




Register-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">
      <message>Username is required</message>
    </field-validator>
  </field>

  <field name="fullName">
    <field-validator type="requiredstring">
      <message>Full Name is required</message>
    </field-validator>
  </field>
```






Register-validation.xml

```
<field name="fromAddress">
  <field-validator type="requiredstring">
    <message>From Address is required</message>
  </field-validator>
  <field-validator type="email">
    <message>Invalid format for From Address</message>
  </field-validator>
</field>
```


```
<field name="replyToAddress">
  <field-validator type="email">
    <message>
      Invalid format for Reply To Address
    </message>
  </field-validator>
</field>
```





Register-validation.xml

```
<field name="password">
  <field-validator type="requiredstring">
    <message>Password is required</message>
  </field-validator>
  <field-validator type="stringlength">
    <param name="trim">true</param>
    <param name="minLength">4</param>
    <param name="maxLength">10</param>
    <message>
      Password length is not in the range 4 through 10.
    </message>
  </field-validator>
</field>
```






Register-validation.xml

```
<field name="password2">
  <field-validator type="requiredstring">
    <message>Confirmation password is required</message>
  </field-validator>
</field>


<validator type="expression">
  <param name="expression">password eq password2</param>
  <message>
    Confirmation Password must match Password, please try again
  </message>
</validator>

</validators>
```





OGNL Expressions

- Object Graph Navigation Language
 - An expression language for getting and setting properties of Java objects
 - Normally, the same expression is used for getting and setting
 - `map['foo']`
 - `map['foo'] = 'bar'`
 - Embedded everywhere
 - tags, valuelstack, *.XML
- 




struts-default.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD
Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <package name="struts-default">
    <result-types>
      <result-type name="chain" class=".ActionChainResult"/>
      <result-type name="dispatcher"
        class=".ServletDispatcherResult" default="true"/>
    ...
  <interceptors>
    <interceptor name="alias" class=".AliasInterceptor"/>
  ...

```






Bundled Validators

default.xml

```
<validators>
  <validator name="required" class=".RequiredFieldValidator"/>
  <validator name="requiredstring" class=".RequiredStringValidator"/>
  <validator name="stringlength" class=".StringLengthFieldValidator"/>
  ...
</validators>
```

com.opensymphony.xwork2.validator.validators

To bundle a different set of validators, place a “validators.xml” file at the base of the application's classpath. Include *all* validators to be used by the application. Use default.xml as a guide.





Bundled Validators

 required

 requiredString

 int

 double

 date

 expression

 fieldExpression

 email

 url

 visitor

 conversion

 stringLength

 regex





Some validation examples


```
<field name="birth">  
  <field-validator type="date">  
    <param name="min">01/01/1970</param>  
    <message>You must have been born after 1970</message>  
  </field-validator>  
</field>
```

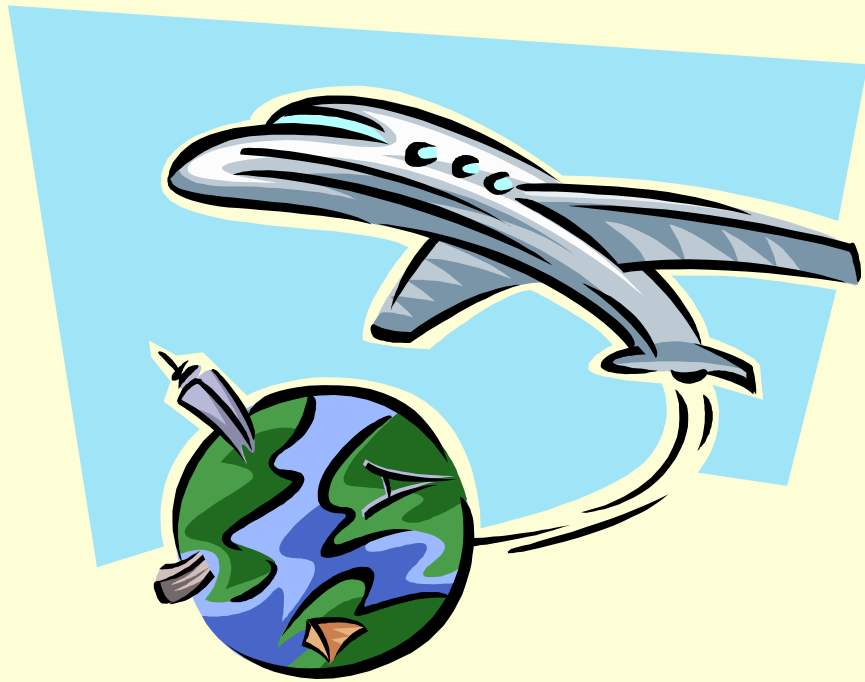




Some validation examples


```
<field name="count">
  <field-validator type="required" short-circuit="true">
    <message>You must enter a value for count.</message>
  </field-validator>
  <field-validator type="int">
    <param name="min">0</param>
    <param name="max">5</param>
    <message>count must be between #{min} and #{max},
      current value is #{count}.</message>
  </field-validator>
</field>
```








Review

- Validations are specified through `*****-validation.xml` files
 - The bundled Validators can be replaced by specifying a `*****.xml` file at the base of the application's classpath.
 - OGNL is used to get and set `*****` of Java objects.
- 




Review

- Validations are specified through ClassName-validation.xml files
 - The bundled Validators can be replaced by placing a validators.xml file at the base of the application's classpath.
 - OGNL is used to get and set properties of Java objects.
- 



Validating Input

- Manual Validation
 - Validation Framework
 - **Validation Workflow**
 - Coding Validators
- 

Shortcircuiting Validators

```
<!DOCTYPE validators PUBLIC "-//OpenSymphony Group//XWork Validator 1.0.2//EN"
"http://www.opensymphony.com/xwork/xwork-validator-1.0.2.dtd">
<validators>
  <!-- Field Validators for email field -->
  <field name="email">
    <field-validator type="required" short-circuit="true">
      <message>You must enter a value for email.</message>
    </field-validator>
    <field-validator type="email">
      <message>Not a valid e-mail.</message>
    </field-validator>
  </field>
  <!-- Field Validators for email2 field -->
  <field name="email2">
    <field-validator type="required">
      <message>You must enter a value for email2.</message>
    </field-validator>
    <field-validator type="email">
      <message>Not a valid e-mail2.</message>
    </field-validator>
  </field>
```




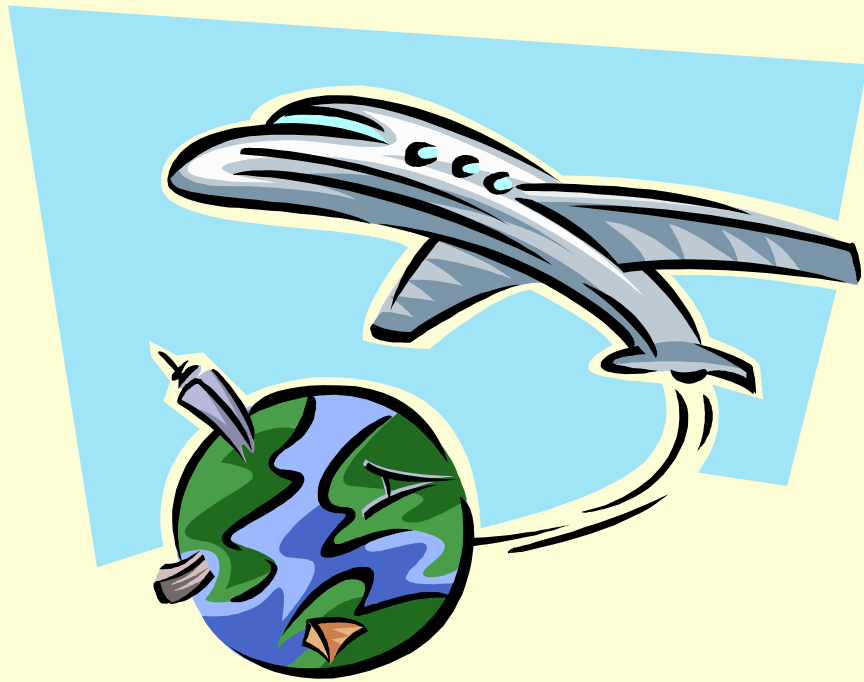
Value Stack

● findValue("max")

- int validator
 - min
 - **max**
- action
 - count

● findValue("count")

- int validator
 - min
 - max
 - action
 - **count**
- 





Review

- To prevent other validations after a certain validator fails, set `*****_*****` to true on that validator.






Review

- To prevent other validations after a certain validator fails, set short-circuit to true on that validator.






Review

- Action and Validator properties be accessed the same way because both are on the `***** *****`.
 - Invalid number or dates value cannot be displayed unless the Action property is a String (True or False).
 - Why?
- 




Review

- Action and Validator properties be accessed the same way because both are on the Value Stack.
 - Invalid number or dates value cannot be displayed unless the Action property is a String (True or False).
 - Why? The invalid String value is pushed onto the stack.
- 

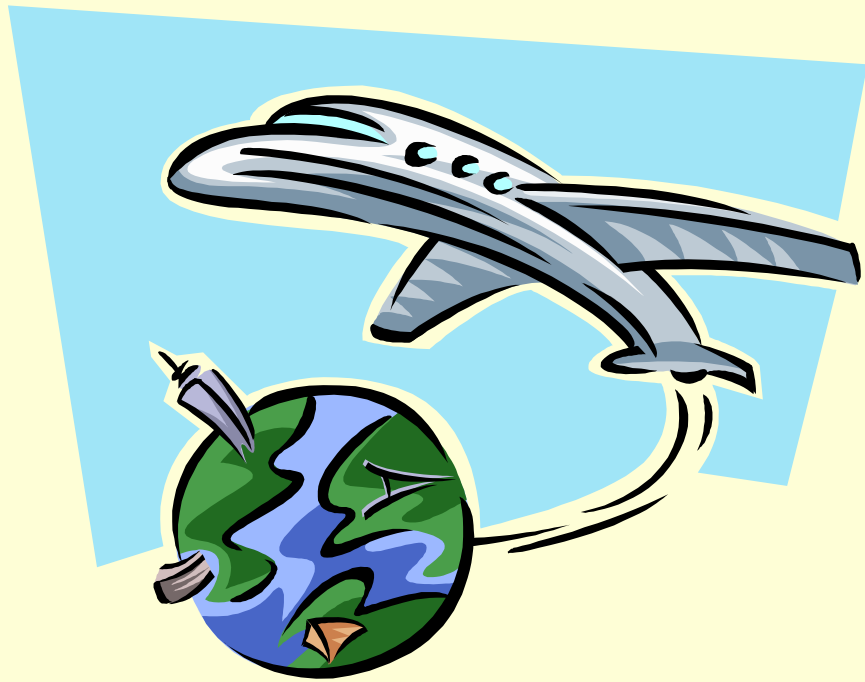


Validating Input

- Manual Validation
 - Validation Framework
 - Validation Workflow
 - Coding Validators
- 


Coding Validators

```
public void validate(Object object)
    throws ValidationException {
    String fieldName = getFieldName();
    Object value = this.getFieldValue(fieldName, object);
    if (!(value instanceof String)) {
        addFieldError(fieldName, object);
    } else {
        String s = (String) value;
        if (doTrim) {
            s = s.trim();
        }
        if (s.length() == 0) {
            addFieldError(fieldName, object);
        }
    }
}
```






Review

- The validate method returns
 - boolean
 - object
 - void
 - If validation fails, a validator
 - Throws an exception
 - Posts an error message
- 




Review

- The validate method returns
 - boolean
 - object
 - void
 - If validation fails, a Validator
 - Throws an exception
 - Posts an error message
- 




Validating Input

- Do we need to validate input?
 - Can a validation framework help?
 - Does validation affect workflow?
 - Can we code custom Validators?
- 




Validating Input

- Do we need to validate input?
 - Garbage in. Garbage out.
 - Can a validation framework help?
 - Does validation affect workflow?
 - Can we code custom Validators?
- 



Validating Input

- Do we need to validate input?
 - Garbage in. Garbage out.
 - Can a validation framework help?
 - Many validations are alike and repetitive and the code is delicate.
 - Does validation affect workflow?
 - Can we code custom Validators?
- 




Validating Input

- Does validation affect workflow?
 - Failed validations create errors to display and correct.
- Can we code custom Validators?





Validating Input

- Does validation affect workflow?
 - Failed validations creates error to display and correct.
 - Can we code custom Validators?
 - Validators are based on an interface and easy to code and plugin.
- 



Struts University Series