

Extracted from:

Stripes

... and Java Web Development Is Fun Again

This PDF file contains pages extracted from Stripes, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit

<http://www.pragprog.com>.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printer versions; the content is otherwise identical.

Copyright © 2008 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.



Figure 6.3: Customizing the display of information messages

6.2 Customizing Error Messages

Error messages can be customized in the same way as information messages, but they also support additional features. They can be displayed in a group or individually next to the input field associated with the error. The labels and input fields that are in error can be highlighted. The message text can be modified. With all these features, we can display error messages so that they fit in well with the look and feel of our web application.

Error Messages in a Group

Much like the `<s:messages/>` tag, the `<s:errors/>` tag generates HTML code using the values defined in `StripesResources.properties`. The keys start with `stripes.errors` and have the default values shown in the following code. This displays error messages as in the example shown in Figure 6.4, on the next page.

[Download](#) email_07/res/StripesResources.properties

```
stripes.errors.header=<div style="color:#b72222; font-weight: bold">\
  Please fix the following errors:</div><ol>
stripes.errors.beforeError=<li style="color: #b72222;">
stripes.errors.afterError=</li>
stripes.errors.footer=</ol>
```

Contact Information

Please fix the following errors:

1. The value (1982-04) entered in field Contact Birth Date must be a valid date
2. Contact Email is a required field
3. 555 is not a valid Contact Phone Number

Email:	<input type="text"/>
First name:	<input type="text"/>
Last name:	<input type="text"/>
Phone number:	<input type="text" value="555"/>
Birth date:	<input type="text" value="1982-04"/>
	<input type="button" value="Save"/> <input type="button" value="Cancel"/>

Figure 6.4: The default display of error messages

Let's modify these values to display error messages in a box with an error icon, as illustrated in Figure 6.5, on the following page:

[Download](#) email_08/res/StripesResources.properties

```
stripes.errors.header=<div class="errors">\
  
stripes.errors.beforeError=<p>
stripes.errors.afterError=</p>
stripes.errors.footer=</div>
```

[Download](#) email_08/web/css/style.css

```
div.errors {
  display: block;
  border: 2px solid #880000;
  margin-bottom: 8px;
  background-color: #FFDDDD;
}
div.errors p {
  font-weight: bold;
  color: #880000;
  margin: 0;
}
```

As we can see, we did that much in the same manner as we changed the display of information messages.

Contact Information

✖ The value (1982-04) entered in field Contact Birth Date must be a valid date
Contact Email is a required field
555 is not a valid Contact Phone Number

Email:

First name:

Last name:

Phone number:

Birth date:

Figure 6.5: Customizing the display of error messages

Error Messages Next to Fields

Stripes makes it easy to display error messages individually, next to each corresponding field, as in Figure 6.6, on the next page. This is nice because the user doesn't have to read the error messages at the top and then scan down the form to figure out to which field each message refers.

If you indicate the name of a field in the `field=` attribute of the `<s:errors>` tag, only the error messages for that field will be displayed. The value for `field=` must match the `name=` attribute of the corresponding input field. For example, this would display error messages concerning the contact's email next to the email field:

```
<td>Email:</td>
<td><s:text name="contact.email"/></td>
<td><s:errors field="contact.email"/></td>
```

After adding `<s:errors>` tags with the `field=` attribute next to each input field, we can remove the `<s:errors/>` tag at the top. Now, to display the messages with the error icon, the entries that start with `stripes.fieldErrors` must be modified:

[Download](#) email_09/res/StripesResources.properties

```
stripes.fieldErrors.header=
stripes.fieldErrors.beforeError=\
  <span class="error">
stripes.fieldErrors.afterError=</span><br/>
stripes.fieldErrors.footer=
```

The screenshot shows a contact form with the following fields and error messages:

- Email:** An empty input field with the error message: **Contact Email is a required field**
- First name:** An empty input field.
- Last name:** An empty input field.
- Phone number:** An input field containing "555" with the error message: **555 is not a valid Contact Phone Number**
- Birth date:** An input field containing "1982-04" with the error message: **The value (1982-04) entered in field Contact Birth Date must be a valid date**

At the bottom of the form are two buttons: "Save" and "Cancel".

Figure 6.6: Displaying error messages next to input fields

We can use the error class on the `` tag to display the error text in bold and red:

[Download](#) email_09/web/css/style.css

```
span.error {
  font-weight: bold;
  color: #880000;
  padding: 8px;
}
```

This will display error messages as in Figure 6.6.

Highlighting Errors

Stripes automatically adds `class="error"` to labels and input fields that are in error, as long as they are created with Stripes tags. We're already using the `<s:text>` tag for the text fields; we need to use an `<s:label>` tag to take advantage of this feature for labels. To associate a label to an input field, place the name of the field in the `for=` attribute of `<s:label>`. For example:

[Download](#) email_10/web/WEB-INF/jsp/contact_form.jsp

```
<tr>
  <td><s:label for="contact.email">Email:</s:label></td>
  <td>
    <s:text name="contact.email" class="required"/>
  </td>
  <td><s:errors field="contact.email"/></td>
</tr>
<!-- same for other fields -->
```



Joe Asks...

How Can I Display Error Messages in More Than One Way?

Changing the values in the `StripesResources.properties` file sets the display of error messages for the whole application. We can override these settings in a page by nesting the `<s:errors-header>`, `<s:individual-error>`, and `<s:errors-footer>` tags within `<s:errors>`:

```
<s:errors>
  <s:errors-header>code for header goes here</s:errors-header>
  code before each message goes here
  <s:individual-error/>
  code after each message goes here
  <s:errors-footer>code for footer goes here</s:errors-footer>
</s:errors>
```

For example, if we wanted to keep the Stripes defaults in `StripesResources.properties` and use the error box just for the contact form, we would have replaced the `<s:errors/>` tag in `contact_form.jsp` with this:

```
<s:errors>
  <s:errors-header>
    <div class="errors">
      
    </s:errors-header>
  <p>
    <s:individual-error/>
  </p>
  <s:errors-footer>
    </div>
  </s:errors-footer>
</s:errors>
```

We can customize field-specific error messages in the same way—just specify the `field=` attribute in the `<s:errors>` tag.

This gives us the possibility of having the most-often used error message display configured in `StripesResources.properties` and still have as many different ways of displaying error messages as we need.

Combining Global and Field-Specific Errors

You can create error messages that are not associated with a specific field with the `addGlobalError()` method of the `ValidationErrors` class. These global errors will not be displayed if you have only `<s:errors field="..." />` tags. Adding the plain `<s:errors />` tag displays global errors but duplicates the field-specific error messages. To combine the display of global and field errors, add the `globalErrorsOnly="true"` attribute to the `<s:errors />` tag. This way, you can display global errors in a group and field-specific errors next to fields:

```
<s:form ...>
  <s:errors globalErrorsOnly="true"/>
  ...
  <td>Email:</td>
  <td><s:text name="contact.email"/></td>
  <td><s:errors field="contact.email"/></td>
  ...
</s:form>
```

Highlighting the labels and text fields that are in error is now a simple matter of some CSS code:

[Download](#) email_10/web/css/style.css

```
input.error {
  border: 2px solid #880000;
  background-color: #FFDDDD;
}
label.error {
  color: #880000;
  font-weight: bold;
  text-decoration: underline;
}
```

This will highlight errors as shown in Figure 6.7, on the next page. Notice that both the labels and the fields that are in error are highlighted.

If the error class is not enough to highlight tags as we require, we can take full control of how tags are rendered when they are in error by implementing the `TagErrorRenderer` interface:

```
public interface TagErrorRenderer {
  void init(InputTagSupport tag);
  void doBeforeStartTag();
  void doAfterEndTag();
}
```

Contact Information

Email: ❌ Contact Email is a required field

First name:

Last name:

Phone number: ❌ 555 is not a valid Contact Phone Number

Birth date: ❌ The value (1982-04) entered in field Contact Birth Date must be a valid date

Figure 6.7: Highlighting labels and input fields for errors

The `DefaultTagErrorRenderer` adds the `class="error"` attribute to tags that are in error. If the tag already had another `class=` defined, such as `"myClass"`, the renderer produces `class="error myClass"` to preserve any previously specified CSS classes.

Suppose we want to display `**` after tags that are in error, as illustrated in Figure 6.8, on the following page. We can do this with a simple implementation of `TagErrorRenderer`:

[Download](#) `email_11/src/stripesbook/ext/MyTagErrorRenderer.java`

```
package stripesbook.ext;
public class MyTagErrorRenderer implements TagErrorRenderer {
    private InputTagSupport tag;
    public void init(InputTagSupport atag) { tag = atag; }
    public void doBeforeStartTag() { }
    public void doAfterEndTag() {
        try { tag.getPageContext().getOut().write("**"); }
        catch (IOException exc)
            { throw new StripesRuntimeException(exc); }
    }
}
```

`TagErrorRenderer` implementations are Stripes extensions, so having the `MyTagErrorRenderer` class in the `stripesbook.ext` package is enough to have it automatically loaded by Stripes. Remember that on page 117 we configured `stripesbook.ext` in `web.xml` as an extension package with the `Extension.Packages` parameter.

The screenshot shows a form titled "Contact Information" with the following fields and error messages:

- Email:** Required field, empty. Error: "The email address is required."
- First name:** Required field, empty.
- Last name:** Required field, empty.
- Phone number:** Required field, contains "555". Error: "The phone number is not valid."
- Birth date:** Required field, contains "1982-04". Error: "The birth date is not valid."

At the bottom of the form are "Save" and "Cancel" buttons.

Figure 6.8: Using a tag error renderer

6.3 Changing the Text of Error Messages

When a validation error occurs, Stripes constructs an error message based on the type of validation that failed, the name of the field, and the value entered by the user. Although this gives messages that are quite reasonable, we can change the text in two ways: by changing the field label and keeping the rest of the text or by changing the text completely. Let's start with using different field labels.

Changing Field Labels

Stripes constructs a field label by taking the name of the field and separating words based on dots (.) and uppercase letters. For example, "contact.phoneNumber" becomes "Contact Phone Number". This label replaces the {0} token in an error message, while {1} is replaced by the value entered by the user. So if the user enters 555 in the contact.phoneNumber field, the following message:

{1} is not a valid {0}

becomes the following:

555 is not a valid Contact Phone Number.

The label you want to appear in an error message may not correspond to the name of the property. I'm sure you don't want to change property names just for labeling purposes. Instead, you can use the label= attribute of @Validate, or you can define the label in the StripesResources.properties file. Let's see how each technique works by changing the labels for the fields of the Contact class.

The Pragmatic Bookshelf

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help you stay on top of your game.

Visit Us Online

Stripes...and Java Web Development Is Fun Again's Home Page

<http://pragprog.com/titles/fdstr>

Source code from this book, errata, and other resources. Come give us feedback, too!

Register for Updates

<http://pragprog.com/updates>

Be notified when updates and new books become available.

Join the Community

<http://pragprog.com/community>

Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy

<http://pragprog.com/news>

Check out the latest pragmatic developments in the news.

Buy the Book

If you liked this PDF, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragprog.com/titles/fdstr.

Contact Us

Phone Orders:	1-800-699-PROG (+1 919 847 3884)
Online Orders:	www.pragprog.com/catalog
Customer Service:	orders@pragprog.com
Non-English Versions:	translations@pragprog.com
Pragmatic Teaching:	academic@pragprog.com
Author Proposals:	proposals@pragprog.com