Internationalization and Struts

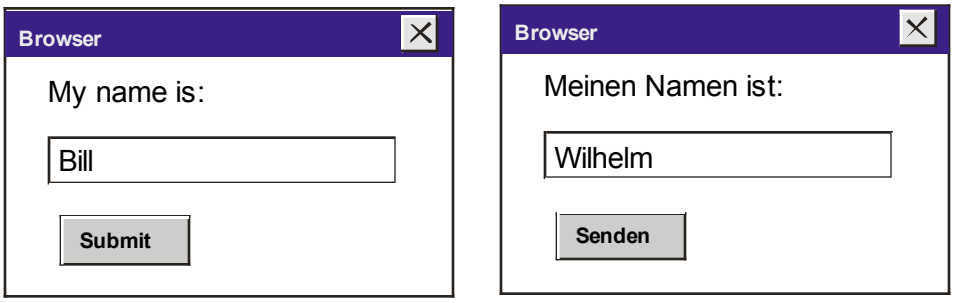
- What are I18N and L9N?
- Struts and JSTL Support
- Message Bundles

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

What is Internationalization?

- Internationalization, often abbreviated as I18N, is the act of writing a program so that it can adapt to different national languages and cultures
- Localization (L9N), is the act of actually adapting the program to a particular locale



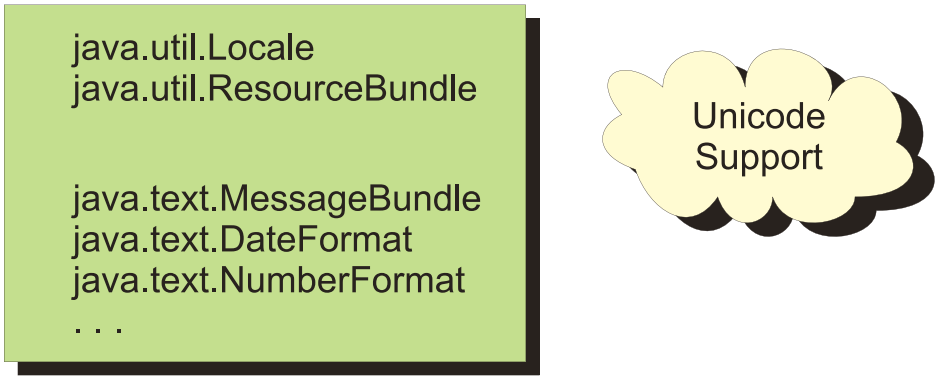
It's a cliche, but the world really is getting smaller. It's likely that Struts applications will be used by people who do not use English as their everday language. Luckily, Struts makes it fairly easy to adapt your program for users in other locales.

Note that localization means more than just translating messages -- it also means using local customs for dates, times and currency.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

J2SE Support for I18N

- The base Java platform provides considerable support for I18N
- Struts builds on and extends that support



Java itself was designed from the very beginning to support I18N. The util and text packages especially provide a wealth of function. The basic class is the Locale class, which gathers together all of the information about a given locale. The ResourceBundle class provides a mechanism to support translatable strings (we've used it indirectly with the ApplicationResources.properties file). Finally, the text package contains several classes that let you format dates, currency and so forth in a locale-specific fashion.

Though the basic Java support is not that difficult to use, the good news is that Struts handles most of this for you!

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Language and Country Codes

- The International Organization for Standards (ISO) has defined codes for both languages and countries
- The combination of a language code and a country code specifies both a national language and local customs (e.g. How to format dates and currency)

Language Codes		Country Codes	
en	English	us	United States
de	German	gb	United Kingdom
fr	French	ca	Canada
es	Spanish	. . .	
ru	Russian		
. . .			

Java uses ISO's codes as a way to categorize resource bundles containing translatable strings. For more information on the ISO codes, see http://www.iso.org/iso/en/prods-services/popstds/popstdsindex.html.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Message Bundle Property Files

- If you hard-code displayable text in your program (e.g. In a System.out.println()), then to change national language, you must recompile
- You can avoid this by defining text in a message bundle and then loading the text at runtime
- J2SE has a standard naming convention for message bundles that specify the language and country codes

ApplicationResources.properties

```
my.msg=Hello
```

ApplicationResources_de.properties

```
my.msg=Guten Tag
```

ApplicationResources_en_GB.properties

```
my.msg=Halloo
```

The trick here is in how you name the resource-bundle files. The "base" file contains all of the translatable strings for the server's default locale. Then you create additional files for each of the locales you wish to support, adding the language code and optionally the country code to the file name. Then at runtime, the system will choose the appropriate file based on the user's locale.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Struts Support for I18N

- Struts maintains a Locale object in the user's session - the Locale is initially set to the server's default
- Struts supports message bundles (e.g. ApplicationResources.properties)
- Struts provides custom JSP actions that are sensitive to the Locale
- Struts applications can use JSTL, which provides additional support, especially for formatting dates and so forth

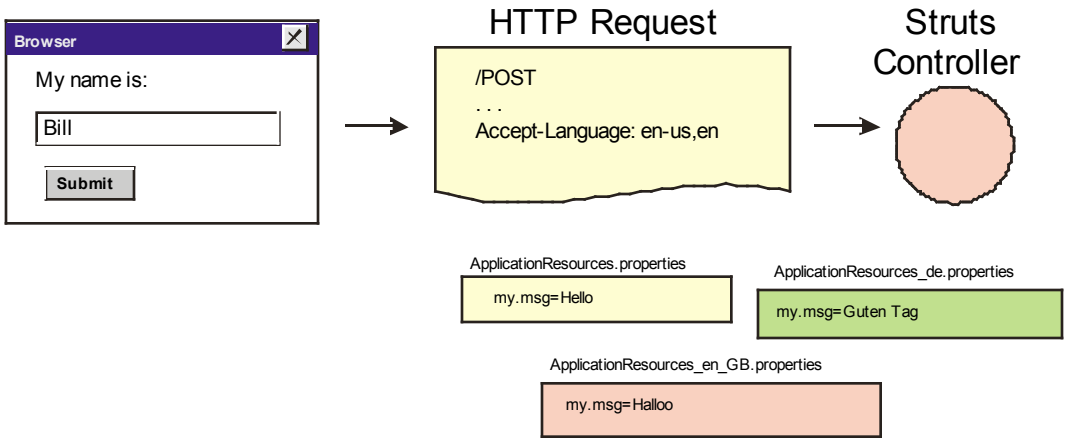
As mentioned earlier, Struts provides some nice features for I18N which makes it fairly easy to write localizable Web applications.

We should note, however, that it's best to write your application using the support from the beginning -- adding I18N capabilities after the fact is a bit tedious.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

How Does Struts Determine Locale?

- Web browsers typically send an HTTP header that specifies the user's preferred language and country
- Struts extracts the Accept-Language header and uses it to store a Locale object in the session



When the first request for a Struts application arrives, Struts examines the HTTP Accept-Language header and uses it to determine the user's preferred locale. Struts then creates a java.util.Locale object and stores its reference under the key of "Action.LOCALE_KEY". Struts then determines if a resource bundle for the user's local is available in the application, and selects one if so, based on the name of the resource bundle.

Note that all of this happens automatically. If you wish, your application can change locales under program control by creating a new Locale object and storing it the session using the standard key.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Using the bean:message Struts Action

- You can use this custom action to display localizable text

my.jsp

```
<%@ taglib
  uri="http://jakarta.apache.org/struts/tags-bean"
  prefix="bean" %>
. . .
<bean:message key="my.msg"/>
```

ApplicationResources.properties

```
my.msg=Hello
```

ApplicationResources_de.properties

```
my.msg=Guten Tag
```


This is the workhorse custom action -- you can use it to display any chunk of translatable text in a Struts view.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

bean:message Struts Action, cont'd

- This tag is also useful to display the text of button widgets

```
<%@ taglib
  uri="http://jakarta.apache.org/struts/tags-bean"
  prefix="bean" %>

<html:form . . .>
. . .
  <html:submit>
    <bean:message key="msg.mybutton"/>
  </html:submit>

</html:form>
```


Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

JSTL Support for I18N

- You can use JSTL tags to format numbers, dates and so forth according to the current locale

```
<%@ taglib prefix="fmt"
   uri="http://java.sun.com/jsp/jstl/fmt" %>
. . .

<fmt:formatNumber
     maxFractionDigits="2" value="${10 div 3}"/>

<jsp:useBean id="now" class="java.util.Date"/>
<fmt:formatDate dateStyle="short" value="${now}"/>
. . .
```


For localization beyond simple substitution of translatable strings, you can use JSTL custom actions. These work using the current default locale to format numbers with the property separator (e.g. period or comma), dates according to local custom and even currency using the local currency symbol.

Each of the JSTL actions accepts attributes that let you customize its output. See the JSTL documentation for a complete listing.

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum

Chapter Summary

In this chapter, you learned:

- How to use Struts and JSTL to create localizable Web applications

Copyright © Descriptor Systems, 2001-2014. Course materials may not be reproduced in whole or in part without prior written consent of Joel Barnum