JSF – properties files (resource bundles)

The .properties file is placed inside WEB-INF/classes. It will consist of name/value pairs.

The content of messages.properties is:

```
title=Registration
firstNamePrompt=First name
lastNamePrompt=Last name
emailAddressPrompt=Email Address
buttonLabel=Register Me
```

index.jsp consists of a redirect:

```
<% response.sendRedirect("form1.faces"); %>
```

form1.jsp contains an <f:loadBundle> tag. Since the basename attribute is messages, that means that messages.properties is the properties file that will be used.

Since the var attribute is msgs, that means that the variable msgs references the values in the properties file.

Then we use the properties themselves relevant to the msgs variable. We'll use <h:outputText > tags when we want to display these properties.

form1.jsp:

```
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<f:view>
<f:loadBundle basename="messages" var="msgs"/>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD><TITLE>
  <h:outputText value="#{msgs.title}"/>
</TITLE>
<LINK REL="STYLESHEET"
      HREF="./css/styles.css"
      TYPE="text/css">
</HEAD>
<BODY>
<CENTER>
<TABLE BORDER=5>
  <TR><TH CLASS="TITLE">
      <h:outputText value="#{msgs.title}"/></TH></TR>
</TABLE>
<P>
<h:form>
<h:outputText value="#{msgs.firstNamePrompt}"/>:
<h:inputText/>
<BR>
<h:outputText value="#{msgs.lastNamePrompt}"/>:
<h:inputText/>
<BR>
<h:outputText value="#{msgs.emailAddressPrompt}"/>:
<h:inputText/>
<BR>
<h:commandButton
  value="#{msgs.buttonLabel}"
  action="nothing"/>
</h:form>

</CENTER></BODY></HTML>
</f:view>
```

We can also do parameterized messages. This is messages2.properties:

```
title=Registration
firstName=first name
lastName=last name
emailAddress=email address
text=Please enter your {0}, {1}, and {2}.
prompt=Enter {0}
buttonLabel=Register Me
```

text uses three parameters and prompt uses one parameter.

The h:outputFormat tag is used to display data from parameterized messages.

```
<h:outputFormat value="#{msgs.someName}">
     <f:param value="value for 0th entry"/>
     <f:param value="value for 1st entry"/>
</h:outputFormat>
```

form2.jsp (but note that the action is to a non-existent bean method:

```
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<f:view locale="#{facesContext.externalContext.requestLocale}">
<f:loadBundle basename="messages2" var="msgs"/>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD><TITLE>
  <h:outputText value="#{msgs.title}"/>
</TITLE>
<LINK REL="STYLESHEET" HREF="./css/styles.css" TYPE="text/css">
</HEAD>
<BODY>
<CENTER>
<TABLE BORDER=5>
<TR><TH CLASS="TITLE">
<h:outputText value="#{msgs.title}"/></TH></TR>
</TABLE>
<BR>
<h:outputFormat value="#{msgs.text}">
      <f:param value="#{msgs.firstName}"/>
      <f:param value="#{msgs.lastName}"/>
      <f:param value="#{msgs.emailAddress}"/>
</h:outputFormat>
<P>
<h:form>
<h:outputFormat value="#{msgs.prompt}">
      <f:param value="#{msgs.firstName}"/>
</h:outputFormat>:
<h:inputText value="#{person.firstName}"/>
<BR>
<h:outputFormat value="#{msgs.prompt}">
      <f:param value="#{msgs.lastName}"/>
</h:outputFormat>:
<h:inputText value="#{person.lastName}"/>
<BR>
<h:outputFormat value="#{msgs.prompt}">
      <f:param value="#{msgs.emailAddress}"/>
</h:outputFormat>:
<h:inputText value="#{person.emailAddress}"/>
<BR>
<h:commandButton
value="#{msgs.buttonLabel}"
action="#{person.doRegistration}"/>
</h:form>
</CENTER></BODY></HTML>
</f:view>
```

Notice one new line in form2.jsp:

```
<f:view locale="#{facesContext.externalContext.requestLocale}">
```

This means that the broswer’s language setting will be used to determine which messages file will be used.

Here, we have messages2_fr.properties and messages2_es.properties.

You can also set the Locale based on user input:

```
locale="#{settings.selectedLocale}"
```