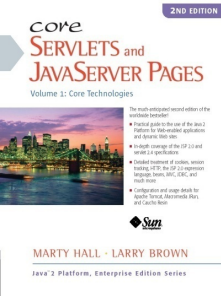
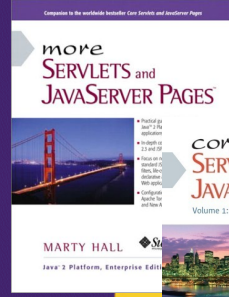




# JSF 2.0: Properties Files, Messages, and I18N

Originals of Slides and Source Code for Examples:  
<http://www.coreservlets.com/JSF-Tutorial/jsf2/>

**Customized Java EE Training:** <http://courses.coreservlets.com/>  
Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



**For live training on JSF 2.x, please see courses at <http://courses.coreservlets.com/>.**



**Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization.**

- Courses developed and taught by Marty Hall
    - JSF 2, PrimeFaces, servlets/JSP, Ajax, jQuery, Android development, Java 6 or 7 programming, custom mix of topics
    - Ajax courses can concentrate on 1 library (jQuery, Prototype/Scriptaculous, Ext-JS, Dojo, etc.) or survey several
  - Courses developed and taught by coreservlets.com experts (edited by Marty)
    - Spring, Hibernate/JPA, EJB3, GWT, Hadoop, SOAP-based and RESTful Web Services
- Contact [hall@coreservlets.com](mailto:hall@coreservlets.com) for details**

# Agenda

- **Creating properties files**
- **Declaring properties files in faces-config.xml**
- **Simple messages**
- **Parameterized messages**
- **Internationalized messages**

5

© 2012 Marty Hall



## Simple Messages

**Customized Java EE Training: <http://courses.coreservlets.com/>**

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

6

# Motivation

- **Idea**
  - Store some fixed strings in a simple plain-text file. Load that file and refer to the strings by the names given in the file.
- **Purpose**
  - Reuse same strings in multiple pages.
  - Update in one fell swoop.
- **Notes**
  - Bean properties are for values that change at runtime.
  - Entries in properties files are much simpler strings that are constant for the life of the application, but either appear multiple places or might change some time in the future.

7

# Displaying Fixed Strings (From Top-Level Folder)

- 1. Create a .properties file**
  - Contains simple keyName=value pairs
  - Must be deployed to WEB-INF/classes
    - In Eclipse, this means you put it in “src” folder
- 2. Declare with resource-bundle in faces-config**
  - base-name gives base file name
  - var gives scoped variable (Map) that will hold results
    - Relative to WEB-INF/classes, “.properties” assumed
    - E.g., for WEB-INF/classes/messages.properties

```
<resource-bundle>
  <base-name>messages</base-name>
  <var>msgs</var>
</resource-bundle>
```
- 3. Output messages using normal EL**
  - `#{msgs.keyName}`

8

# Displaying Fixed Strings (Minor Variation: Subfolders)

## 1. Create a .properties file

- Contains simple keyName=value pairs
- Deployed to WEB-INF/classes/**resources** (or other name)
  - In Eclipse, this means you create package called “**resources**” and put the properties file in the **resources** package directory

## 2. Declare with resource-bundle in faces-config

- base-name gives **package name**, dot, base file name
- var gives scoped variable (Map) that will hold results
  - E.g., for WEB-INF/classes/resources/messages.properties

```
<resource-bundle>
  <base-name>resources.messages</base-name>
  <var>msgs</var>
</resource-bundle>
```

<resource-bundle> goes inside <application>. See later slide for details.

## 3. Output messages using normal EL

- #{msgs.keyName}

9

# More on resource-bundle Element

## • Inside the “application” element

```
<?xml version="1.0"?>
<faces-config ...>
  <application>
    <resource-bundle>
      <base-name>messages</base-name>
      <var>msgs</var>
    </resource-bundle>
  </application> ...
</faces-config>
```

## • Mapping to actual file

- The file extension (.properties) is assumed
- Relative to WEB-INF/classes (“src” in Eclipse)

## • Note for JSF 1.x users

- Do not use old-style f:loadBundle tag

10

# Copying Properties Files From One Project to Another

- **Issue**

- Eclipse won't let you directly drag/drop files onto "src". So, it is tricky to copy .properties files into new projects
  - Creating a *new* properties file is no problem: just R-click "src" and do "new File"

- **Solution 1**

- Use Eclipse Navigator, not Package Explorer
  - Window → Show View → Navigator (close when done!)
  - You could also find folder on file system, copy/paste there, then return to Eclipse, R-click project, and do "Refresh"

- **Solution 2**

- Create a file in default package, then drop there.
  - R-click on "src", then New → Class. Make up a class name. This class is placed in default package. Drag/drop .properties files into that folder or R-click and New → File and make a new .properties file. Then delete temp class.

11

## src/messages1.properties

```
registrationTitle=Registration
registrationText=Please enter your first name,
    last name, and email address.
firstNamePrompt=Enter first name
lastNamePrompt=Enter last name
emailAddressPrompt=Enter email address
buttonLabel=Register Me
successTitle=Success
successText=You registered successfully.
```

This is a single line in actual file. You can break long lines into multiple lines by putting \ at the end of a line.

- At runtime, this will be  
.../WEB-INF/classes/messages1.properties
- faces-config.xml will load this with

```
<resource-bundle>
  <base-name>messages1</base-name>
  <var>msgs1</var>
</resource-bundle>
```
- Facelets page will output messages with
  - `#{msgs1.firstNamePrompt}`

12

# faces-config.xml

```
<?xml version="1.0"?>
<faces-config ...>
  <application>
    <resource-bundle>
      <base-name>messages1</base-name>
      <var>msgs1</var>
    </resource-bundle>
  </application>
</faces-config>
```

13

# simple-messages.xhtml (Top)

```
<!DOCTYPE ... >
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html">
<h:head><title>#{msgs1.registrationTitle}</title>
<link href="./css/styles.css"
      rel="stylesheet" type="text/css"/>
</h:head>
<h:body>
<table border="5" align="center">
  <tr>
    <th class="title">#{msgs1.registrationTitle}</th></tr>
</table>
<h3>#{msgs1.registrationText}</h3>
```

14

## simple-messages.xhtml (Bottom)

```
<h:form>
  #{msgs1.firstNamePrompt}:
  <h:inputText value="#{person1.firstName}"/>
  <br/>
  #{msgs1.lastNamePrompt}:
  <h:inputText value="#{person1.lastName}"/>
  <br/>
  #{msgs1.emailAddressPrompt}:
  <h:inputText value="#{person1.emailAddress}"/>
  <br/>
  <h:commandButton value="#{msgs1.buttonLabel}"
                    action="#{person1.doRegistration}"/>
</h:form>
</h:body></html>
```

15

## Person.java

```
public abstract class Person {
    private String firstName, lastName, emailAddress;

    public String getFirstName() {
        return(firstName);
    }

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

    ... // get/setLastName, get/setEmailAddress

    public abstract String doRegistration();
}
```

16

# Person1.java

```
@ManagedBean
public class Person1 extends Person {
    public String doRegistration() {
        return("success1");
    }
}
```

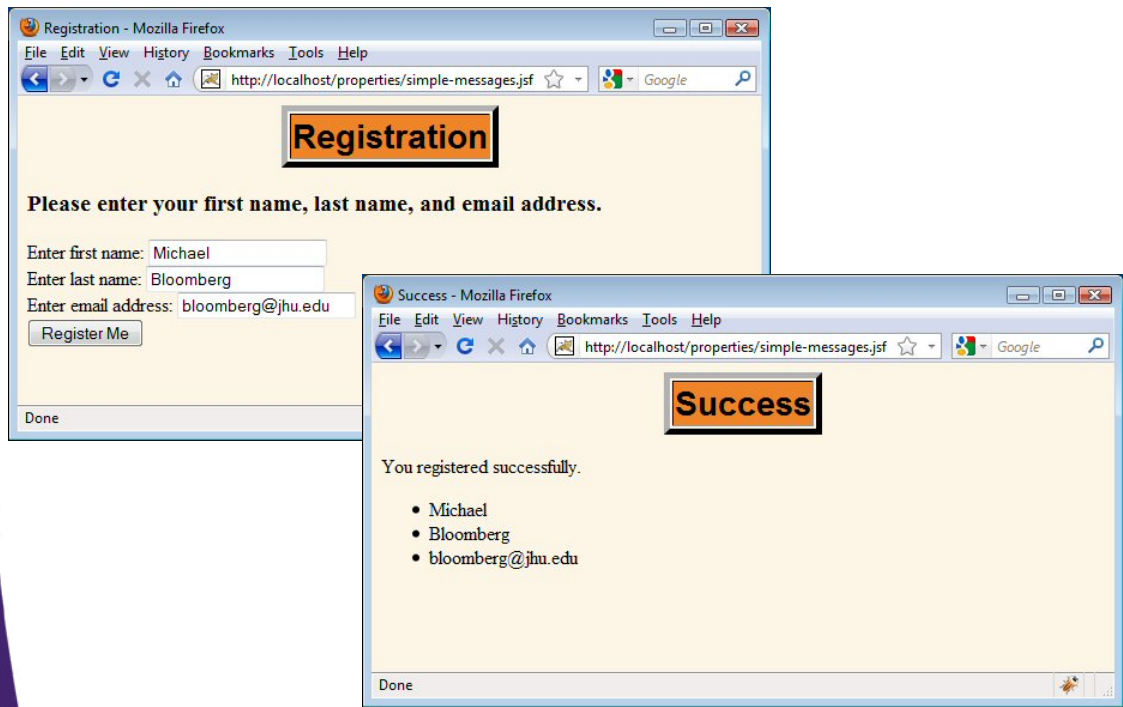
17

# success1.xhtml

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html">
<h:head><title>#{msgs1.successTitle}</title>
...
</h:head>
<h:body>
<table border="5" align="center">
  <tr><th class="title">#{msgs1.successTitle}</th></tr>
</table>
<p>#{msgs1.successText}</p>
<ul>
  <li>#{person1.firstName}</li>
  <li>#{person1.lastName}</li>
  <li>#{person1.emailAddress}</li>
</ul>
</h:body></html>
```

18

# Results



19

© 2012 Marty Hall



# Parameterized Messages

Customized Java EE Training: <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

20

# Motivation

- **Idea**
  - Store some strings in a simple plain-text file. Load that file and refer to the strings by the names given in the file. Allow portions of the strings to be replaced.
- **Purpose**
  - Make strings more flexible by having one string refer to another.
  - Allow runtime values to be inserted into strings.
    - Particularly useful for prompts and error messages
- **Notes**
  - These are no longer purely static strings. However, the basic outline of the string is still fixed, so they are different from bean properties, which are very dynamic.

21

# Approach: Parameterizing Strings

- 1. Create a .properties file in src**
  - Values contain {0}, {1}, {2}, etc.
  - E.g., someName=blah {0} blah {1}
    - Reminder: “src” in Eclipse becomes WEB-INF/classes when deployed
- 2. Declare file with resource-bundle as before**
  - base-name gives base file name
  - var gives scoped variable (Map) that will hold results
- 3. Output messages using h:outputFormat**
  - value gives base message
  - Nested f:param gives substitution values. These can be literal strings or runtime values
  - E.g.:

```
<h:outputFormat value="#{msgs.someName}">
  <f:param value="Literal value for 0th entry"/>
  <f:param value="#{someBean.calculatedValForEntry1}"/>
</h:outputFormat>
```

22

## More on f:param

- **You must define f: namespace**
  - Since the tag uses a different prefix (f:), you must define the f: namespace in the <html...> tag at the top
  - We did this in some earlier sections when we used f:selectItems inside of h:selectOneMenu

- **Example**

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
<h:head>...</h:head>
<h:body>...</h:body>
</html>
```

23

## messages2.properties

```
registrationTitle=Registration
firstName=First Name
lastName=Last Name
emailAddress=Email Address
registrationText=Please Enter Your {0}, {1}, and {2}.
prompt=Enter {0}
buttonLabel=Register Me
successTitle=Success
successText=You Registered Successfully.
```

24

# faces-config.xml

```
<?xml version="1.0"?>
<faces-config ...>
  <application>
    <resource-bundle>
      <base-name>messages1</base-name>
      <var>msgs1</var>
    </resource-bundle>
    <resource-bundle>
      <base-name>messages2</base-name>
      <var>msgs2</var>
    </resource-bundle>
  </application>
</faces-config>
```

25

# parameterized-messages.xhtml (Snippet)

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
  <h:head><title>#{msgs2.registrationTitle}</title>
  ...
  <h3>
    <h:outputFormat value="#{msgs2.registrationText}">
      <f:param value="#{msgs2.firstName}" /> Replaces {0} in registrationText
      <f:param value="#{msgs2.lastName}" /> Replaces {1} in registrationText
      <f:param value="#{msgs2.emailAddress}" /> Replaces {2} in registrationText
    </h:outputFormat>
  ...
  ...
```

26

# Person2.java

```
@ManagedBean
public class Person2 extends Person {
    public String doRegistration() {
        return("success2");
    }
}
```

27

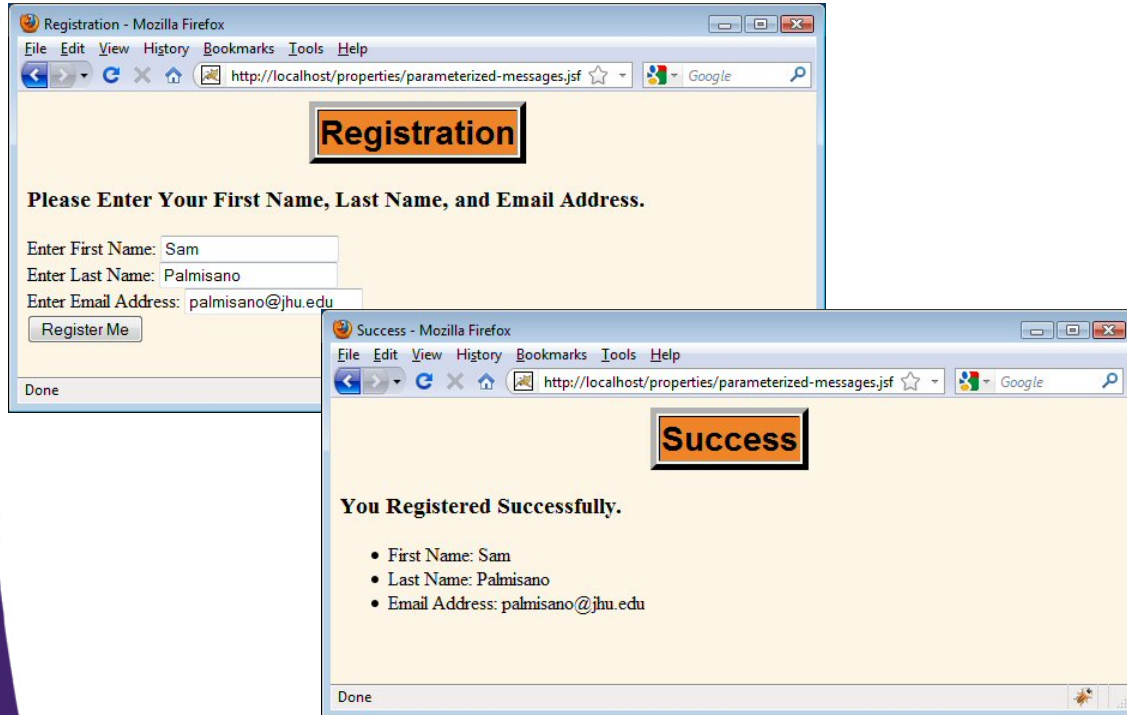
# success2.xhtml

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html">
<h:head><title>#{msgs2.successTitle}</title>
...
</h:head>
<h:body>
<table border="5" align="center">
  <tr><th class="title">#{msgs2.successTitle}</th></tr>
</table>
<h3>#{msgs2.successText}</h3>
<ul>
  <li>#{msgs2.firstName}: #{person2.firstName}</li>
  <li>#{msgs2.lastName}: #{person2.lastName}</li>
  <li>#{msgs2.emailAddress}: #{person2.emailAddress}</li>
</ul>
</h:body></html>
```

Since "First Name" (etc.) were substituted into the prompts on input page (instead of being hardcoded into "Enter First Name"), then those phrases are reusable in other pages without repetition in properties file.

28

# Results



29

© 2012 Marty Hall



# Internationalization and Messages

Customized Java EE Training: <http://courses.coreservlets.com/>

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

30

# Motivation

- **Idea**

- Store some multiple versions of properties file (msg.properties, msg\_es.properties, msg\_es\_mx.properties, etc.)
  - Base properties file gets loaded (e.g., msg.properties). Then one most loosely matching current Locale, if any (e.g., msg\_es.properties), then one more specifically matching, if any (e.g., msg\_es\_mx.properties). If same name occurs in more than one file, later file wins.

- **Purpose**

- Let page be displayed in multiple languages

- **Notes**

- This example will show Locale selected based on browser language settings. See tutorial section on event handling to see Locale selected based on user choices.

31

# Approach: Localizing Strings

1. **Create multiple similarly named .properties files**

- blah.properties, blah\_es.properties, blah\_es\_mx.properties

2. **Use f:view and locale attribute**

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

- Determines locale from browser language settings
- Note: can also set the Locale based on user input
  - See event handling section for best approach

3. **Declare file with resource-bundle as before**

- base-name gives base file name
  - Version matching Locale will be used *automatically*
- var gives scoped variable (Map) that will hold results

4. **Output using h:outputFormat or normal EL**

- Same as before

32

## Quick Example: Properties Files

- **messages.properties**
  - company=JsfResort.com
  - feature=Our {0}:
  - pool=swimming pool
- **messages\_es.properties**
  - feature=Nuestra {0}:
  - pool=piscina
- **messages\_es\_mx.properties**
  - pool=alberca

33

## Quick Example: faces-config.xml

```
<?xml version="1.0"?>
<faces-config ...>
  <application>
    <resource-bundle>
      <base-name>messages</base-name>
      <var>msgs</var>
    </resource-bundle>
  </application>
</faces-config>
```

34

# Quick Example: Facelets

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
  <f:view
    locale="#{facesContext.externalContext.requestLocale}">
    ...
    <h1>#{msgs.company}</h1>
    <h2>
    <h:outputFormat value="#{msgs.feature}">
      <f:param value="#{msgs.pool}"/>
    </h:outputFormat>
    </h2>
    
    ...
  </f:view></html>
```

35

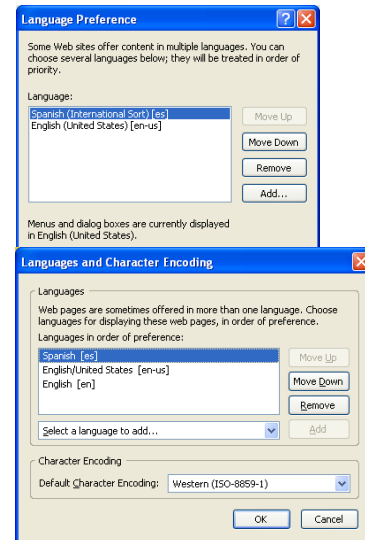
# Quick Example: Results

- **English (or any language except Spanish)**  
**JsfResort.com**  
Our swimming pool:  
*(Picture of pool)*
- **Non-Mexican Spanish**  
**JsfResort.com**  
Nuestra piscina:  
*(Picture of pool)*
- **Mexican Spanish**  
**JsfResort.com**  
Nuestra alberca:  
*(Picture of pool)*

36

# Setting Language Preferences in Browsers

- **Internet Explorer**
  - Tools, Internet Options, General, Languages
  - Click Add, select language, OK
  - Move to top of list using “Move Up”
- **Firefox 3**
  - Enter “about:config” as URL
  - Scroll down to entry named “general.useragent.locale”
  - Double click it and enter Locale by hand (es, es-mx), etc.
- **Firefox 4 through Firefox 6**
  - Tools, Options, Content tab, Languages, click Choose, move to top



37

# messages2.properties

```
registrationTitle=Registration
firstName=First Name
lastName=Last Name
emailAddress=Email Address
registrationText=Please Enter Your {0}, {1}, and {2}.
prompt=Enter {0}
buttonLabel=Register Me
successTitle=Success
successText=You Registered Successfully.
```

38

## messages2\_es.properties

```
registrationTitle=Registro
firstName=Primer Nombre
lastName=Apellido
emailAddress=Dirección de Email
registrationText=Incorpore Por This is a single line in actual file.
                    Favor su {0}, {1}, y {2}.
prompt=Incorpore {0}
buttonLabel=Coloquéme
successTitle=Éxito
successText=Se Registró con Éxito.
```

## messages2\_fr.properties

```
registrationTitle=Enregistrement
firstName=Prénom
lastName=Nom
emailAddress=Adresse électronique
registrationText=Merci de Entrer This is a single line in actual file.
                    Votre {0}, {1}, et {2}.
prompt=Entrez Votre {0}
buttonLabel=Enregistrez Moi
successTitle=Succès
successText=Vous Avez Enregistré Avec Succès.
```

# faces-config.xml

```
<?xml version="1.0"?>
<faces-config ...>
  <application>
    <resource-bundle>
      <base-name>messages1</base-name>
      <var>msgs1</var>
    </resource-bundle>
    <resource-bundle>
      <base-name>messages2</base-name>
      <var>msgs2</var>
    </resource-bundle>
  </application>
</faces-config>
```

41

# internationalized-messages.xhtml (Snippet)

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
  <f:view locale="#{facesContext.externalContext.requestLocale}">
    <h:head><title>#{msgs2.registrationTitle}</title>
    ...
    <h3>
    <h:outputFormat value="#{msgs2.registrationText}">
      <f:param value="#{msgs2.firstName}"/>
      <f:param value="#{msgs2.lastName}"/>
      <f:param value="#{msgs2.emailAddress}"/>
    </h:outputFormat>
    ...
  </f:view></html>
```

All uses of `#{msgs2...}` exactly the same as in previous example. However, which properties files are loaded depends on the browser settings.

42

# Person3.java

```
@ManagedBean
public class Person3 extends Person {
    public String doRegistration() {
        return("success3");
    }
}
```

43

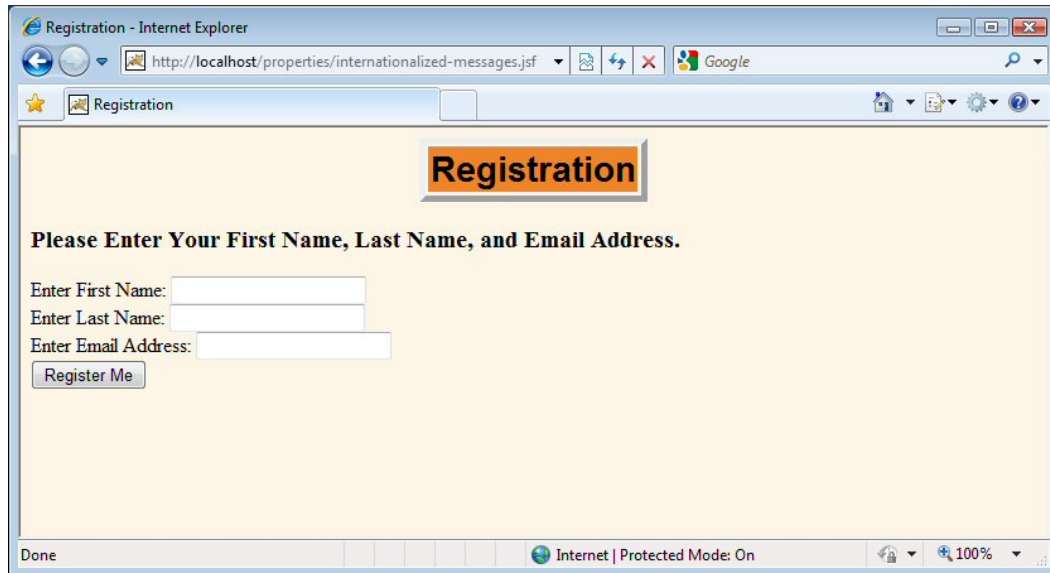
# success3.xhtml

```
<!DOCTYPE ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html">
  <f:view locale="#{facesContext.externalContext.requestLocale}">
  <h:head><title>#{msgs2.successTitle}</title>
  ...
  </h:head>
  <h:body>
  <table border="5" align="center">
    <tr><th class="title">#{msgs2.successTitle}</th></tr>
  </table>
  <h3>#{msgs2.successText}</h3>
  <ul>
    <li>#{msgs2.firstName}: #{person3.firstName}</li>
    <li>#{msgs2.lastName}: #{person3.lastName}</li>
    <li>#{msgs2.emailAddress}: #{person3.emailAddress}</li>
  </ul>
  </h:body></f:view>
```

44

All uses of #{msgs2...} exactly the same as in previous example. However, which properties files are loaded depends on the browser settings.

# Result (Browser Language English)



Registration - Internet Explorer

http://localhost/properties/internationalized-messages.jsf

Registration

## Registration

**Please Enter Your First Name, Last Name, and Email Address.**

Enter First Name:

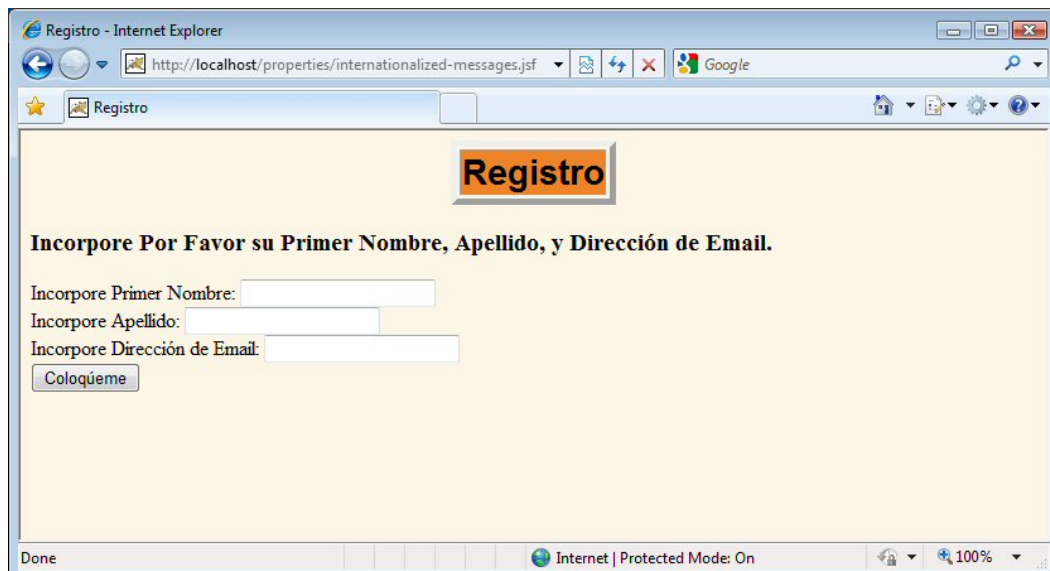
Enter Last Name:

Enter Email Address:

Done Internet | Protected Mode: On 100%

45

# Result (Browser Language Spanish)



Registro - Internet Explorer

http://localhost/properties/internationalized-messages.jsf

Registro

## Registro

**Incorpore Por Favor su Primer Nombre, Apellido, y Dirección de Email.**

Incorpore Primer Nombre:

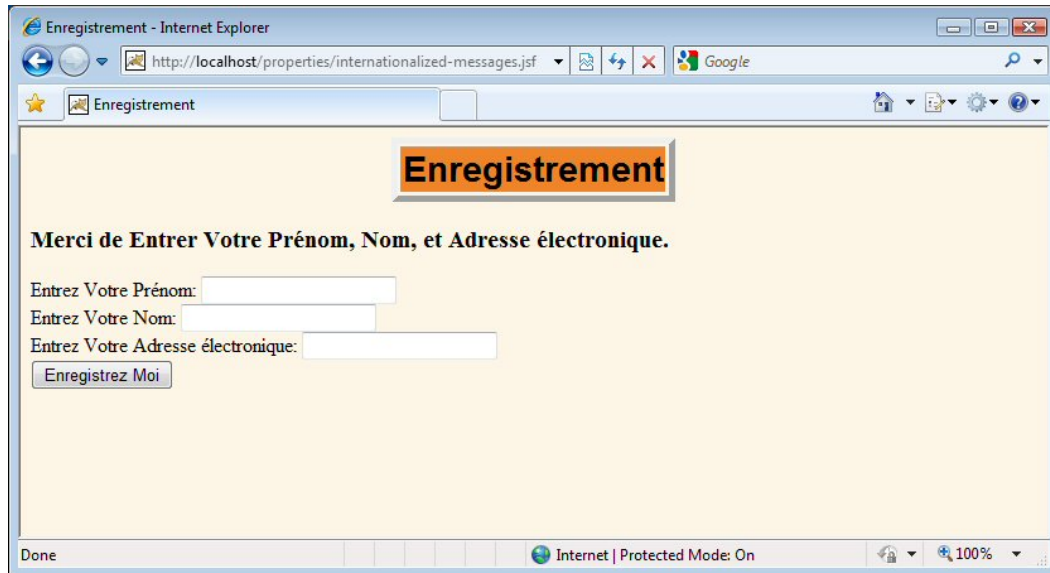
Incorpore Apellido:

Incorpore Dirección de Email:

Done Internet | Protected Mode: On 100%

46

# Result (Browser Language French)



47

© 2012 Marty Hall



## Wrap-Up

**Customized Java EE Training: <http://courses.coreservlets.com/>**

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.  
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

48

# Summary

- **Deploy one or more .properties files**
  - In Eclipse, you put .properties file in “src” folder, and it gets deployed to WEB-INF/classes automatically
- **Declare with resource-bundle in faces-config**

```
<application>
  <resource-bundle>
    <base-name>someFile</base-name>
    <var>someVar</var>
  </resource-bundle> ...
</application>
```
- **Output values using normal EL**
  - `#{someVar.someNameFromFile}` for simple values
  - `h:outputFormat` for parameterized values
- **Set view's locale if I18N needed**
  - Extract it from browser setting or user setting
    - We'll cover user settings in section on event handling
  - Automatically loads locale-specific resource bundle

49

© 2012 Marty Hall



## Questions?

**Customized Java EE Training: <http://courses.coreservlets.com/>**

Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

50