

JENI 3 – 10 Modul Praktikum JSF Lanjutan

1. Tujuan

- JSf Validator
- External Validation
- Converter
- Merubah default message dengan ResourceBundle properties
- Internalization

2. Latar Belakang

JSF Standard Validator menyediakan 3 macam validator yaitu DoubleRangeValidator, LengthValidator, LongRangeValidator. Penggunaan JSF Validator pada aplikasi cukup menambahkan tag ke dalam input form.

External Validation merupakan cara memvalidasi input dari user dengan menggunakan kode/ aturan yang kita definisikan sendiri, tidak menggunakan fungsi/ komponen yang sudah ada. Latihan External Validation akan mengambil kasus validasi email.

Converter

JSF menyediakan sejumlah converter yang bisa dipakai untuk mengubah input text ke tipe object. Dalam prosesnya converter juga mengecek apakah data yang akan dikonvert valid atau tidak (sesuai dengan tipe yang akan dikonvert).

Internationalization

JSF mendukung konfigurasi bahasa sesuai dengan setting language pada browser. Biasanya dipakai untuk membuat site yang multilanguage.

3. Percobaan

Percobaan 1 : JSF Validator



Tujuan section ini adalah menggunakan validator dalam pemrosesan form.

JSF Standard Validator

JSF Standard Validator menyediakan 3 macam validator yaitu DoubleRangeValidator, LengthValidator, LongRangeValidator. Penggunaan JSF Validator pada aplikasi cukup menambahkan tag ke dalam input form. Pada aplikasi GuestBook tadi tambahkan atribut required="true" jika field input harus

JENI 3 – 10 Modul Praktikum JSF Lanjutan

diisi, tag `<f:validateLength minimum="" maximum="">` untuk membatasi jumlah minimum dan maksimum karakter yang harus dituliskan. Selengkapnya kode berikut:

```
<body>
<h1>GuestBook Form</h1>
<f:view>
<h:form id="guestBookForm">
  <h:panelGrid columns="3" bgcolor="#e6edfd">
    <h:outputLabel for="name">
      <h:outputText value="Name:" />
    </h:outputLabel>
    <h:inputText id="name" value="#{guestBook.name}" required="true">
      <f:validateLength minimum="4"/>
    </h:inputText>
    <h:message for="name"/>

    <h:outputLabel for="sex">
      <h:outputText value="Sex:" />
    </h:outputLabel>
    <h:selectOneListbox id="sex" value="" size="3" title="select one
option">
      <f:selectItem id="s1" itemLabel="Male" itemValue="male" />
      <f:selectItem id="s2" itemLabel="Female"
itemValue="female" />
    </h:selectOneListbox>
    <h:message for="sex"/>

    <h:outputLabel for="email">
      <h:outputText value="Email:" />
    </h:outputLabel>
    <h:inputText id="email" value="#{guestBook.email}"
required="true"/> <h:message for="email" />

    <h:outputLabel for="birthdate">
      <h:outputText value="Birthdate:" />
    </h:outputLabel>
    <h:inputText id="birthdate" value="#{guestBook.birthdate}" />
    <h:message for="birthdate"/>

    <h:outputLabel for="message">
      <h:outputText value="Message:" />
    </h:outputLabel>
    <h:inputTextarea id="message" value="#{guestBook.message}" />
    <h:message for="message"/>

    <h:commandButton type="reset" value="Reset" />
    <h:commandButton type="submit" action="#{guestBook.addGuest}"
value="Add" />
    <h:outputText value="" />
  </h:panelGrid>
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

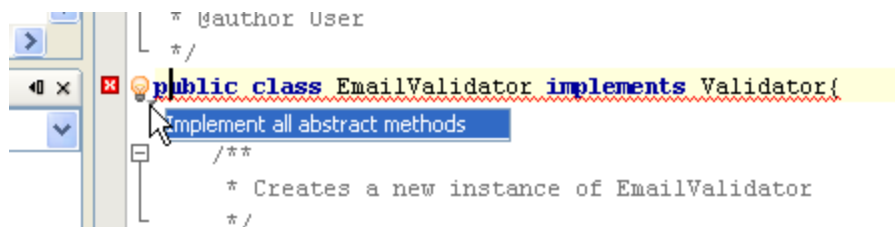
```
</h:form>
</f:view>
</body>
```

Menggunakan External Validation

External Validation merupakan cara memvalidasi input dari user dengan menggunakan kode/ aturan yang kita definisikan sendiri, tidak menggunakan fungsi/ komponen yang sudah ada. Latihan External Validation akan mengambil kasus validasi email.

Langkah-langkahnya:

1. Buat package **jeni3.jsf.validator**
2. Buat kelas baru **EmailValidator** dalam paket **jeni3.jsf.validator**. Kelas ini mengimplementasikan method validate dari interface Validator. Gambar berikut satu cara mengimplementasikan method validate secara otomatis oleh Netbeans.



Berikut kode selengkapnya:

```
package jeni3.jsf.validator;

import java.util.regex.Matcher;
import java.util.regex.Pattern;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.validator.Validator;
import javax.faces.validator.ValidatorException;

/**
 * @author mee_andto@yahoo.com
 * @version 0.5
 */
public class EmailValidator implements Validator{
    public void validate(FacesContext context, UIComponent component,
        Object value) throws ValidatorException {
        //get the component's contents and cast it to a String
        String enteredEmail = (String)value;
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
//set the email pattern
Pattern p = Pattern.compile(".*@.*\\.([a-z]+)");
//match the given string with the pattern
Matcher m = p.matcher(enteredEmail);
boolean matchFound = m.matches();
if (!matchFound){
    FacesMessage message = new FacesMessage();
    message.setDetail("Email not valid - The email must be in
the format yourname@yourdomain.com");
    message.setSummary("Email not valid - The email must be in
the format yourname@yourdomain.com");
    message.setSeverity(FacesMessage.SEVERITY_ERROR);
    throw new ValidatorException(message);
}
}
```

3. Definisikan class validator anda di file **faces-config.xml**, seperti kode berikut:

```
<validator>
    <description>A validator for checking the email</description>
    <validator-id>EmailValidator</validator-id>
    <validator-class>jeni3.jsf.validator.EmailValidator</validator-
class>
</validator>
```

4. Buat class Backen Bean dengan nama LoginFormBean.
(**LoginFormBean.java**), letakkan dalam paket **jeni3.jsf.login**.

```
package jeni3.jsf.login;

import java.util.Map;
import javax.faces.context.FacesContext;

/**
 * @author mee\_andto@yahoo.com
 * @version 0.5
 */
public class LoginForm {

    /** Creates a new instance of LoginForm */
    public LoginForm() {
    }

    private String username = "";
    private String email = "";

    public String authenticate(){
        if (username.equals("jeni") &&
email.equals("arek@malang.info")){
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
        FacesContext context = FacesContext.getCurrentInstance();

        Map sessionMap =
context.getExternalContext().getSessionMap();
        sessionMap.put("email",email);
        return "success";
    }else{
        return "failure";
    }
}
//add more here getter and setter for each private variable
```

Jangan lupa menambahkan getter dan setter. **Refactor – Encapsulate Fields.**

5. Definisikan bean dan navigation rule dalam file **faces-config.xml**.

```
<managed-bean>
  <description>
  </description>
  <managed-bean-name>loginForm</managed-bean-name>
  <managed-bean-class>jeni3.jsf.login.LoginForm</managed-bean-class>
  <managed-bean-scope>request</managed-bean-scope>
</managed-bean>
<navigation-rule>
  <from-view-id>/login/loginForm.jsp</from-view-id>
  <navigation-case>
    <from-action>#{loginForm.authenticate}</from-action>
    <from-outcome>success</from-outcome>
    <to-view-id>/login/loginSuccess.jsp</to-view-id>
  </navigation-case>
  <navigation-case>
    <from-action>#{loginForm.authenticate}</from-action>
    <from-outcome>failure</from-outcome>
    <to-view-id>/login/loginError.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
```

6. Buat form login berupa file JSP. Nama file **loginForm.jsp**, letakkan dalam direktori **login**. Berikut kodenya:

```
<%@ page contentType="text/html"%>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
<style>
//css style
.error{
    size:15px;
}
</style>
<f:view>
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
<html>
  <head><title>Login Page</title></head>
  <body>
    <h1>Login Form</h1>
    <h:form>
      <h:panelGrid columns="3" bgcolor="#e6edfd">
        <h:outputText value="Enter your name: " />
        <h:inputText id="username" value="#{loginForm.username}"
required="true"/>
          <h:message for="username" styleClass="error" />

        <h:outputText value="Enter your email: " />
        <h:inputText id="email" value="#{loginForm.email}"
required="true">
          <f:validator validatorId="EmailValidator" />
        </h:inputText>
        <h:message for="email" styleClass="error" />

        <h:outputText value="" />
        <h:commandButton type="submit" value="Login"
action="#{loginForm.authenticate}" />
        <h:outputText value="" />
      </h:panelGrid>
    </h:form>
  </body>
</html>
</f:view>
```

7. Buat file JSP jika login berhasil. File **loginSuccess.jsp** dalam folder **login**.

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-
8">
    <title>Login Successfull</title>
  </head>
  <body>
    <f:view>
      <h:panelGrid columns="2" bgcolor="#e6edfd">
        <f:facet>
          <h:outputText style="font-size:14px" value="Login
successfull" />
        </f:facet>
        <h:outputText style="font-size:14px" value="Name ="/>
          <h:outputText value="#{loginForm.username}"/><br/>
        <h:outputText style="font-size:14px" value="Email ="/>
          <h:outputText value="#{loginForm.email}"/>
        </h:panelGrid>
      </f:view>
    </body>
  </html>
```

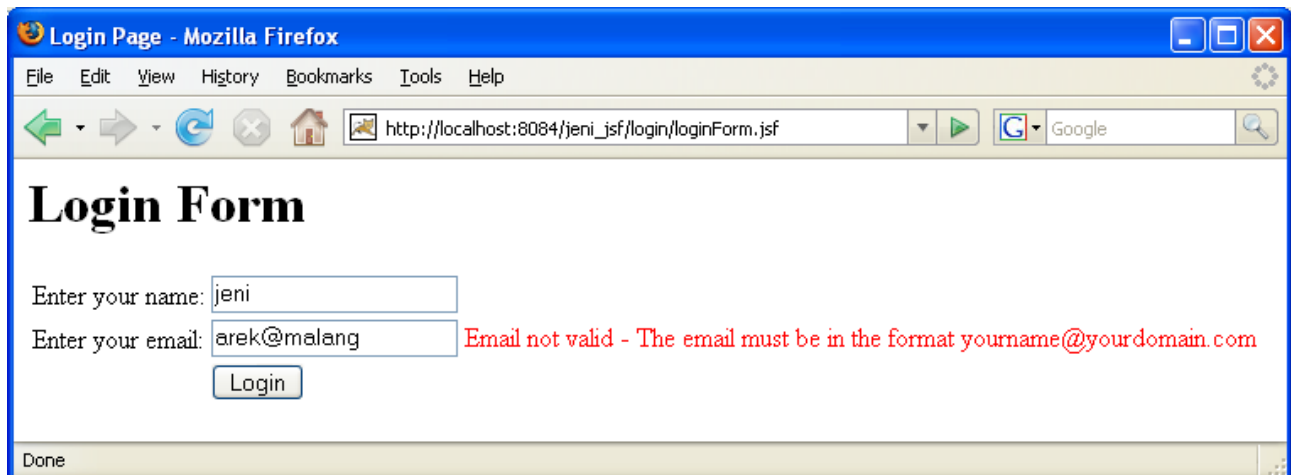
JENI 3 – 10 Modul Praktikum JSF Lanjutan

8. Buat file JSP jika login berhasil. File **loginError.jsp** dalam folder **login**.

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Login Error</title>
  </head>
  <body>

    <h1>Login Error</h1>
  </body>
</html>
```

9. Run – Run Main Project



Sampai disini, anda bisa menggunakan EmailValidator untuk memvalidasi email di sembarang form. Nah coba anda validasi Form Guestbook pada latihan sebelumnya dengan EmailValidator ini. Anda juga bisa membuat Validator lainnya sekarang.

JENI 3 – 10 Modul Praktikum JSF Lanjutan

Percobaan 2 : JSF Converter



Info

Tujuan section ini adalah membuat converter

JSF menyediakan sejumlah converter yang bisa dipakai untuk mengubah input text ke tipe object. Dalam prosesnya converter juga mengecek apakah data yang akan dikonvert valid atau tidak (sesuai dengan tipe yang akan dikonvert).

Menggunakan Converter Standard.

Langkah-langkahnya:

- Berikut latihan untuk mencoba beberapa converter yang telah disediakan oleh JSF framework.
- Pada Web Pages buat halaman JSP dengan nama file **converterForm.jsp** dalam folder **converter**. Berikut kodenya:

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>JSF Converters</title>
  </head>
  <style>
    //css style
    .error{
      size:15px;
    }
  </style>
  <body>
    <f:view>
      <table border="0">
        <tbody>
          <tr>
            <td>
              <h4>DateTime converter</h4>
              <p>Display only the date and the dateStyle is
<i>short</i></p>
              <h:outputText value="#{myForm.dateVar}">
                <f:convertDateTime type="date" dateStyle="short" />
              </h:outputText>
              <p>Display only the time and the timeStyle is
<i>full</i></p>
              <h:outputText value="#{myForm.dateVar}">
                <f:convertDateTime type="time" timeStyle="full" />
              </h:outputText>
              <p>Display both date and time, the dateStyle is <i>full</i>

```


JENI 3 – 10 Modul Praktikum JSF Lanjutan

```

and the
    locale is <i>ru</i></p>
    <h:outputText value="#{myForm.dateVar}">
        <f:convertDateTime type="both" dateStyle="full"
locale="ru" />
    </h:outputText>
    <p>Display the both, date and time and the dateStyle is
    <i>short</i></p>
    <h:outputText value="#{myForm.dateVar}">
        <f:convertDateTime type="both" dateStyle="short" />
    </h:outputText>
    <p>Display a date with the pattern <i>dd.MM.yyyy
HH:mm</i></p>
    <h:outputText value="#{myForm.dateVar}">
        <f:convertDateTime pattern="dd.MM.yyyy HH:mm" />
    </h:outputText>
    <h:form id="datetime1">
        <p>Input a date and the dateStyle is <i>short</i></p>
        <h:inputText value="#{myForm.dateVar}">
            <f:convertDateTime type="date" dateStyle="short" />
        </h:inputText>
        <h:commandButton value="Send" />
    </h:form>
    <h:form id="datetime2">
        <p>Input a date matching this pattern
<i>dd.MM.yyyy</i></p>
        <h:inputText value="#{myForm.dateVar}">
            <f:convertDateTime pattern="dd.MM.yyyy" />
        </h:inputText>
        <h:commandButton value="Send" />
    </h:form>
</td>

        <td>
            <h4>Number converter</h4>
            <p>Display maximal <i>3 integer digits</i></p>
            <h:outputText value="#{myForm.integerVar}">
                <f:convertNumber maxIntegerDigits="3" />
            </h:outputText>
            <p>Display type is <i>currency</i> and the currencySymbol is
            <i>$</i></p>
            <h:outputText value="#{myForm.integerVar}">
                <f:convertNumber type="currency" currencySymbol="$" />
            </h:outputText>
            <p>Display type is <i>percent</i></p>
            <h:outputText value="#{myForm.integerVar}">
                <f:convertNumber type="percent" />
            </h:outputText>
            <p>Display maximal 4 fraction digits</p>
            <h:outputText value="#{myForm.floatVar}">
                <f:convertNumber maxFractionDigits="4" />
            </h:outputText>
            <p>Display number matching pattern <i>###0,00%</i></p>
            <h:outputText value="#{myForm.floatVar}">

```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```

        <f:convertNumber pattern="###0,00%"/>
    </h:outputText>
    <h:form id="number1">
        <p>Input a number but only the integer digits will be
        processed</p>
        <h:inputText value="#{myForm.integerVar}">
            <f:convertNumber integerOnly="true"/>
        </h:inputText>
        <h:commandButton value="Send" />
    </h:form>
    <h:form id="number2">
        <p>Input a number matching the pattern <i>##0,00</i></p>
        <h:inputText value="#{myForm.integerVar}">
            <f:convertNumber pattern="##0,00"/>
        </h:inputText>
        <h:commandButton value="Send" />
    </h:form>
</td>
        </tr>
    </tbody>
</table>

</f:view>
</body>
</html>

```

- **Buat Backen Bean:**

```

package jeni3.jsf.converter;

import java.util.Date;

/**
 * @author mee_andto@yahoo.com
 * @version 0.5
 */
public class MyFormBean {
    private Date dateVar = new Date();
    private Integer integerVar = new Integer(2023);
    private Float floatVar = new Float(9.099783);

    //Add here more getter and setter (use menu Refactor->Encapsulate
    Fields)
}

```

- **Definisikan bean dalam faces-config.xml:**

```

<managed-bean>
    <managed-bean-name>myForm</managed-bean-name>
    <managed-bean-class>jeni3.jsf.converter.MyFormBean</managed-bean-
class>
    <managed-bean-scope>request</managed-bean-scope>
</managed-bean>

```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
<navigation-rule>
  <from-view-id>/converter/converterForm.jsp</from-view-id>
</navigation-rule>
```

- Run – Run Main Project

Membuat Converter sendiri

Untuk membuat Converter sendiri dapat dilakukan dengan membuat class baru yang mengimplementasikan abstract class `javax.faces.convert.Converter`.

Langkah-langkahnya:

- Buat Java class `MyDateConverter` dalam pacakage `jeni3.jsf.converter` dengan mengimplementasikan interface `Converter`.

```
package jeni3.jsf.converter;

public class MyDateConverter implements Converter{

}
```

Gunakan hint pada Netbeans untuk mengimplementasikan method yang dimiliki oleh `Converter`, maka anda akan mendapatkan dua fungsi berikut:

```
public Object getAsObject(FacesContext context, UIComponent component,
String value) {
}
public String getAsString(FacesContext context, UIComponent component,
Object value) {
}
```

- Kemudian tambahkan kode untuk convert format tanggal, seperti berikut:

```
public Object getAsObject(FacesContext context, UIComponent component,
String value) throws ConverterException{
    String pattern = "dd/MM/yyyy";
    SimpleDateFormat sdf = new SimpleDateFormat(pattern);
    Date nDate;
    try {
        nDate = sdf.parse(value);
    } catch (ParseException e) {
        FacesMessage message = new FacesMessage();
        message.setDetail("Date is missing or not valid");
        message.setSummary("Date is missing or not valid");
        message.setSeverity(FacesMessage.SEVERITY_ERROR);
        throw new ConverterException(message);
    }
}
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
        if (nDate.getTime() > new Date().getTime()){
            FacesMessage message = new FacesMessage();
            message.setDetail("Date is bigger than current date");
            message.setSummary("Date is bigger than current date");
            message.setSeverity(FacesMessage.SEVERITY_ERROR);
            throw new ConverterException(message);
        }
        return nDate;
    }
```

Info: Tekan Alt+Shift+F atau gunakan Netbeans hint untuk menambahkan statements import class (pilih java.text.ParseException, import java.text.SimpleDateFormat dan java.util.Date).
Method ini menggunakan ConverterException class untuk menangani Exception yang terjadi.

- Nah return value dari method `getAsObject` masih berupa Object sehingga perlu dirubah ke String, maka dipakailah fungsi `getString`. Tambahkan kode berikut:

```
public String getString(FacesContext context, UIComponent component,
Object value) {
    return value.toString();
}
```

- Kemudian definisikan class `MyDateConverter` in dalam file konfigurasi `faces-config.xml`.

```
<converter>
    <converter-id>MyDateConverter</converter-id>
    <converter-class>jeni3.jsf.converter.MyDateConverter</converter-
class>
</converter>
```

Selanjutnya `MyDateConverter` kita gunakan dalam tag `<f:converter />`.

- Buka file **login/loginForm.jsp** dan tambahkan kode berikut:

```
<h:outputText value="Enter your birthdate (dd/mm/yyyy): " />
<h:inputText id="birthdate" value="#{loginForm.birthdate}"
required="true">
    <f:converter converterId="MyDateConverter" />
</h:inputText>
<h:message for="birthdate" styleClass="error" />
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

- Tambahkan juga pada file **login/loginSuccess.jsp** kode untuk menampilkan *birthdate*:

```
<h:outputText value="BirthDate = " />
<h:outputText value="#{loginForm.birthdate}" />
```

- Buka file **LoginFormBean.java** dan tambahkan field *birthdate*, seperti kode berikut:

```
package jeni3.jsf.login;

import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIInput;
import javax.faces.context.FacesContext;
import javax.faces.validator.ValidatorException;

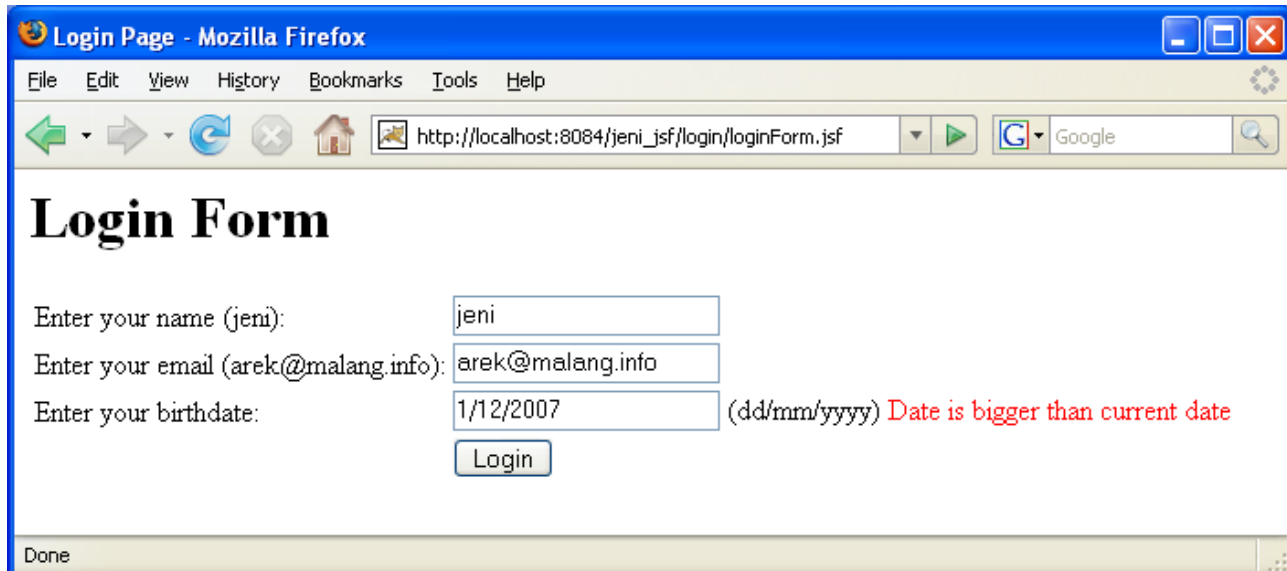
/**
 * @author mee_andto@yahoo.com
 * @version 0.5
 */
public class LoginFormBean {

    private String username;
    private String email;
    private Date birthdate;

    //Add here more getter and setter (Netbeans = menu Refactor->Encapsulate Fields)
}
```

- Run-Run Main Project

JENI 3 – 10 Modul Praktikum JSF Lanjutan



Percobaan 3 : Resource Bundle dan Internasionalisasi



Tujuan section ini adalah membuat pesan-pesan dan menyediakan akses berbagai bahasa.

Merubah default message dengan ResourceBundle properties

Pesan-pesan error atau informasi yang tampil dikendalikan oleh file *Message.properties* yang terletak dalam paket *javax.faces* di *jsf-impl.jar*.

Untuk mengganti pesan-pesan error yang muncul bisa dilakukan dengan cara sebagai berikut:

- Buat file properties bernama **myMessages.properties** letakkan dalam package **jeni3.jsf.resource**

Salin key yang ada dalam file *Message.properties* ke *myMessages.properties* atau buat key sendiri.

```
javax.faces.component.UIInput.REQUIRED=Please enter a value for this field.
javax.faces.converter.DateTimeConverter.DATE=Please enter a valid date.
javax.faces.converter.DateTimeConverter.DATE_detail=Please enter a valid date. Example: {1}
```

```
title = Login Form
username = Enter your name :
email= Enter your email :
birthdate = Enter your birthdate :
login= Login
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

- Buka file *faces-config.xml* dan deklarasikan dimana letak *myMessages.properties*.

```
<application>
    <message-bundle>jeni3.jsf.resource.MyMessages</message-bundle>
</application>
```

- Untuk menggunakannya dalam file JSP dapat dilakukan dengan tag berikut:
Load **<f:loadBundle basename="jeni3.jsf.resource.myMessages" var="msg"/>**

<h:outputText value="#{msg.title}">

Berikut modifikasi untuk loginForm.jsp:

```
<%@ page contentType="text/html"%>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
<style>
//css style
.error{
    size:15px;
}
</style>
<f:view>
<f:loadBundle basename="jeni3.jsf.resource.myMessages" var="msg"/>
<html>
    <head><title>Login Page</title></head>
    <body>
        <h:outputText value="#{msg.title}"/>
        <h:form>
            <h:panelGrid columns="3" bgcolor="#e6edfd">
                <h:outputText value="#{msg.username}"/>
                <h:inputText id="username" value="#{loginForm.username}"
required="true"/>
                    <h:message for="username" styleClass="error" />

                <h:outputText value="#{msg.email}"/>
                <h:inputText id="email" value="#{loginForm.email}"
required="true">
                    <f:validator validatorId="EmailValidator" />
                </h:inputText>
                <h:message for="email" styleClass="error" />

                <h:outputText value="#{msg.birthdate}"/>
                <h:inputText id="birthdate" value="#{loginForm.birthdate}"
required="true">
                    <f:converter converterId="MyDateConverter" />
                </h:inputText>
                <h:message for="birthdate" styleClass="error" />
            </h:panelGrid>
        </h:form>
    </body>
</html>
```

JENI 3 – 10 Modul Praktikum JSF Lanjutan

```
<h:outputText value="" />
<h:commandButton value="#{msg.login}"
action="#{loginForm.authenticate}" />
<h:outputText value="" />
</h:panelGrid>
</h:form>
</body>
</html>
</f:view>
```

Internationalization

JSF mendukung konfigurasi bahasa sesuai dengan setting language pada browser. Biasanya dipakai untuk membuat site yang multilanguage. Untuk kasus login di atas akan dibuat tampilan yang mendukung bahasa Inggris (en), bahasa Jerman (de) dan bahasa Indonesia (id).

Berikut langkah-langkahnya:

- Buat file properties untuk ke-3 bahasa tersebut:
File **myMessages.properties** sebagai default sama dengan sebelumnya (tidak ada perubahan)
File **myMessages_de.properties** untuk bahasa jerman.

```
javax.faces.component.UIInput.REQUIRED=Bitte geben Sie einen Wert für  
dieses Feld.  
javax.faces.converter.DateTimeConverter.DATE=Bitte geben Sie ein  
gültiges Datum an.  
javax.faces.converter.DateTimeConverter.DATE_detail=Bitte geben Sie ein  
gültiges Datum an. Beispiel: {1}  
  
title = Anmeldung  
username = Geben Sie Ihren Name:  
email= Geben Sie Ihre E-mail-Adresse :  
birthdate = Geben Sie Ihr Geburtsdatum :  
login= Einloggen
```

File **myMessages_id.properties** untuk bahasa Indonesia.

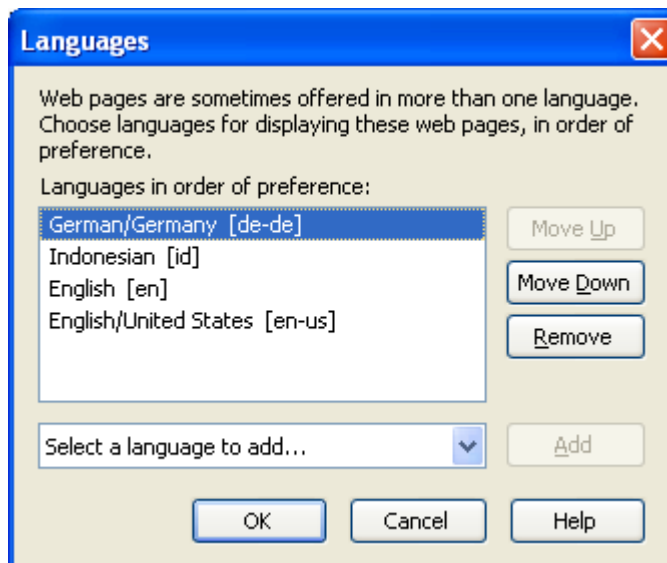
```
javax.faces.component.UIInput.REQUIRED=Masukkan sebuah nilai untuk field  
ini.  
javax.faces.converter.DateTimeConverter.DATE=Masukkan tanggal yang  
benar.  
javax.faces.converter.DateTimeConverter.DATE_detail=Masukkan tanggal  
yang benar. Contoh: {1}  
  
title = Form Login  
username = Masukkan nama anda:  
email= Masukkan alamat email anda:  
birthdate = Masukkan tanggal lahir anda:  
login= Login
```


JENI 3 – 10 Modul Praktikum JSF Lanjutan

- Pada file loginForm.jsp modifikasi kode f:view dengan menambahkan atribut locale yaitu definisi bahasa yang sedang dipakai.

```
<f:view locale="facesContext.externalContext.request.locale">
```

- Run – Run main Project
- Tambahkan bahasa yang dalam konfigurasi browser anda. Konfigurasi disini hanyalah urutan prioritas bahasa yang akan digunakan untuk menampilkan pesan.



- Untuk testing gantilah Regional Language melalui Control Panel.