Events

Events are action controllers or event listeners:

1. **Action controllers** handle main form submission
   1. Fire *after* bean has been populated (click button)
   2. Fire *after* validation logic (JSF does validation but we have not enabled it yet)
   3. Return strings that directly affect page navigation
2. **Event listeners** handle UI events
   1. Often fire *before* bean has been populated
   2. Often bypass validation logic
   3. Never directly affect page navigation

Ajax

To update the entire page, use event listeners.
If you need to update only part of the page, use Ajax.

Types of Event Listeners

1. ActionListener

Fired by buttons, image maps, and hypertext links**:**

```
<h:commandButton value="..." .../>
<h:commandButton image="..." .../>
<h:commandLink .../>
```

Button in same h:form as input elements

<h:commandButton … **actionListener**="#{testBean.*method1(anything)*}" **immediate**="true"/>

public void *method1*(ActionEvent e) { …}

The **immediate** attribute prevents the setter methods from firing for the input elements, and blocks validation
The **method1** format must follow the signature shown above.

We use e <h:commandButton actionListener="..." .../>

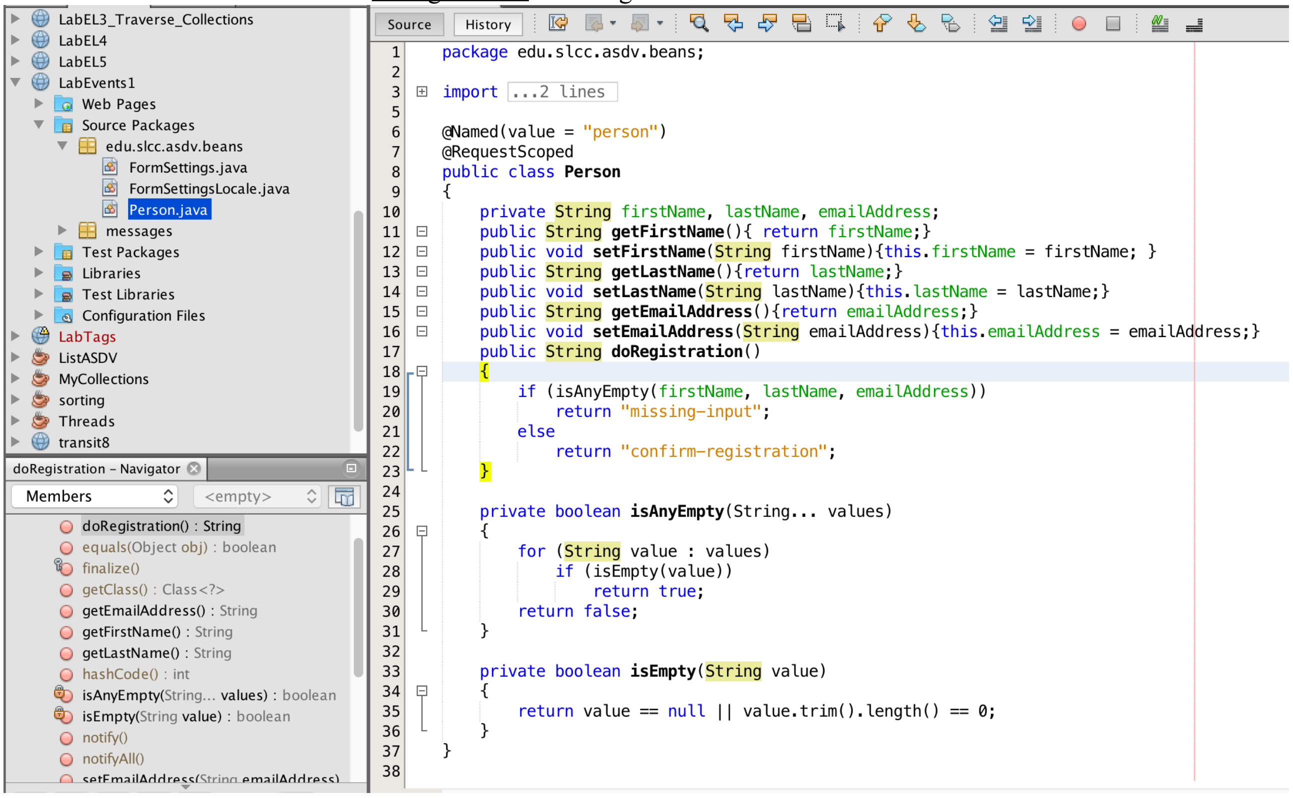
We usually want this method1 to occur before beans are populated and especially before validation occurs so use “immediate” to designate that listener fires before validation is performed or beans are populated

Button in separate h:form from **input elements**

<h:commandButton… **action**="#{testBean.**navigation**}"/>

The form has no input elements.
The navigation has normal action controller signature (zero args and returning String), and can just return null to tell JSF to redisplay original page after the method finishes.

1. Create Web App LabEvents1 and the request scope bean <u>Person</u>. Setter and getters for the 3 properties via Insert Code. The method <u>doRegistration</u> for navigation.



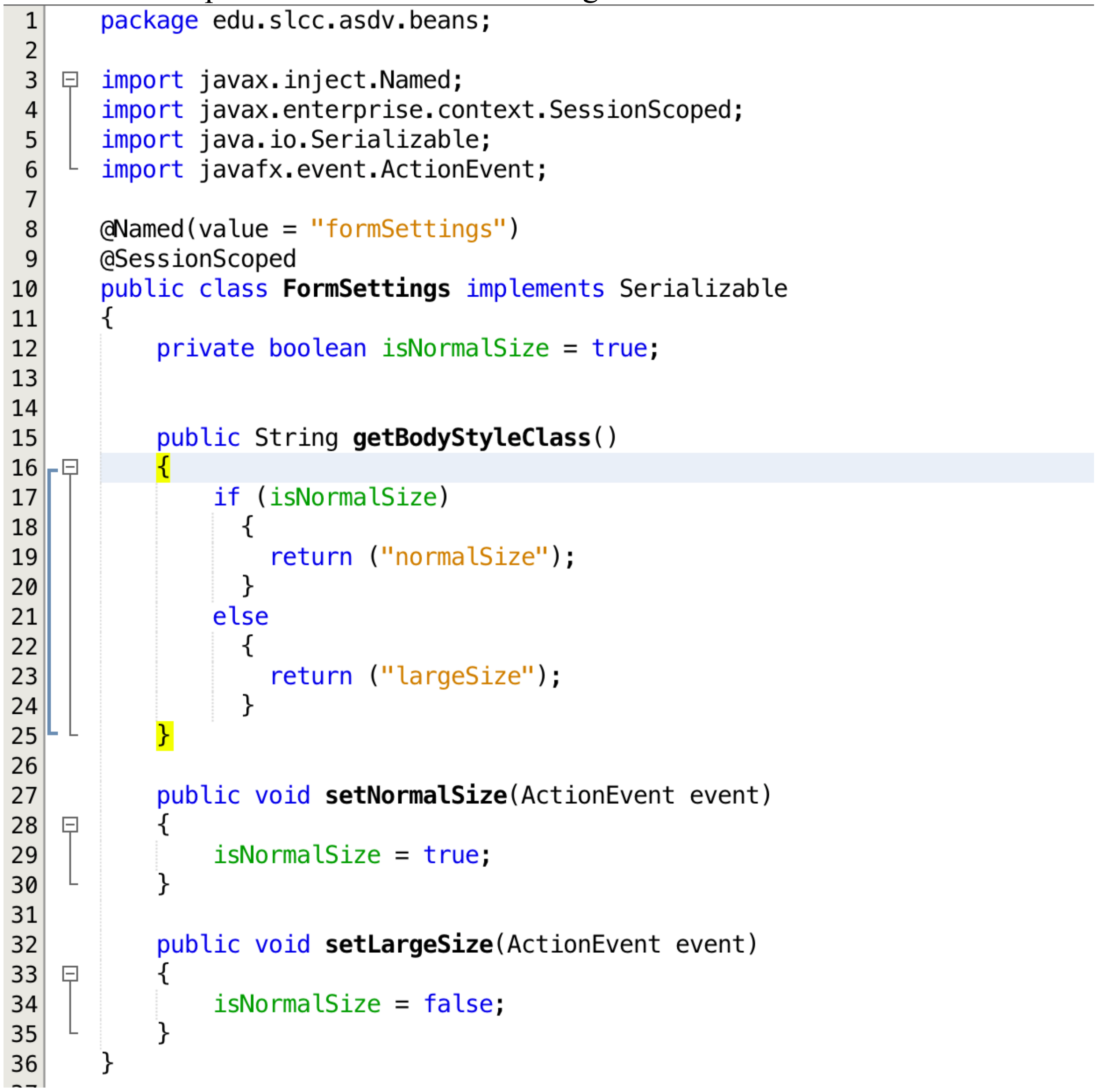
LabEL3_Traverse_Collections
LabEL4
LabEL5
LabEvents1
Web Pages
Source Packages
edu.slcc.asdv.beans
FormSettings.java
FormSettingsLocale.java
Person.java
messages
Test Packages
Libraries
Test Libraries
Configuration Files
LabTags
ListASDV
MyCollections
sorting
Threads
transit8

doRegistration - Navigator

doRegistration() : String
equals(Object obj) : boolean
finalize()
getClass() : Class<?>
getEmailAddress() : String
getFirstName() : String
getLastName() : String
hashCode() : int
isAnyEmpty(String... values) : boolean
isEmpty(String value) : boolean
notify()
notifyAll()
setEmailAddress(String emailAddress)

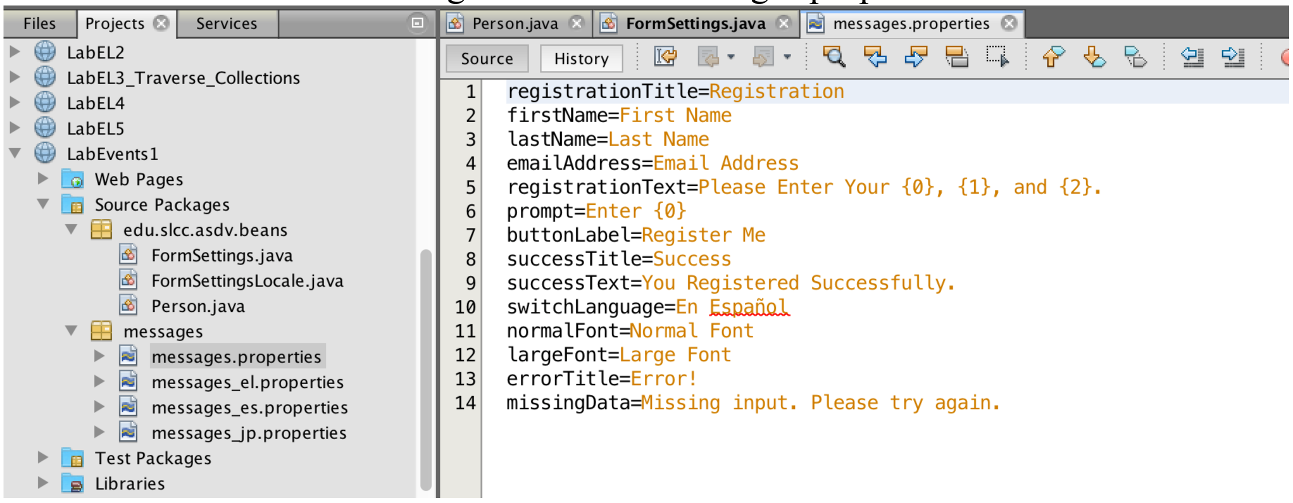
```java
package edu.slcc.asdv.beans;

import ...2 lines

@Named(value = "person")
@RequestScoped
public class Person
{
    private String firstName, lastName, emailAddress;
    public String getFirstName(){ return firstName;}
    public void setFirstName(String firstName){this.firstName = firstName; }
    public String getLastName(){return lastName;}
    public void setLastName(String lastName){this.lastName = lastName;}
    public String getEmailAddress(){return emailAddress;}
    public void setEmailAddress(String emailAddress){this.emailAddress = emailAddress;}
    public String doRegistration()
    {
        if (isAnyEmpty(firstName, lastName, emailAddress))
            return "missing-input";
        else
            return "confirm-registration";
    }

    private boolean isAnyEmpty(String... values)
    {
        for (String value : values)
            if (isEmpty(value))
                return true;
        return false;
    }

    private boolean isEmpty(String value)
    {
        return value == null || value.trim().length() == 0;
    }
}
```

2. Create scoped session bean FormSettings



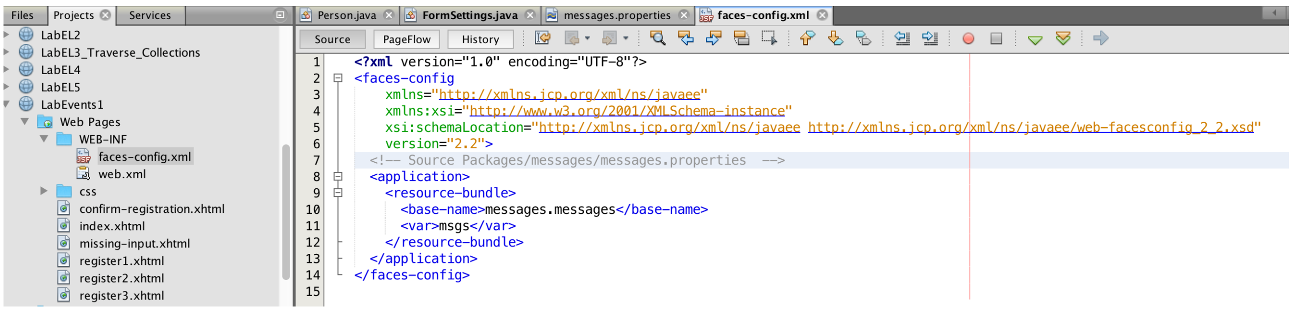
```java
package edu.slcc.asdv.beans;

import javax.inject.Named;
import javax.enterprise.context.SessionScoped;
import java.io.Serializable;
import javafx.event.ActionEvent;

@Named(value = "formSettings")
@SessionScoped
public class FormSettings implements Serializable
{
    private boolean isNormalSize = true;


    public String getBodyStyleClass()
    {
        if (isNormalSize)
          {
            return ("normalSize");
          }
        else
          {
            return ("largeSize");
          }
    }

    public void setNormalSize(ActionEvent event)
    {
        isNormalSize = true;
    }

    public void setLargeSize(ActionEvent event)
    {
        isNormalSize = false;
    }
}
```

3. Under the folder messages create the messages.properties file



```
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.
switchLanguage=En Español
normalFont=Normal Font
largeFont=Large Font
errorTitle=Error!
missingData=Missing input. Please try again.
```

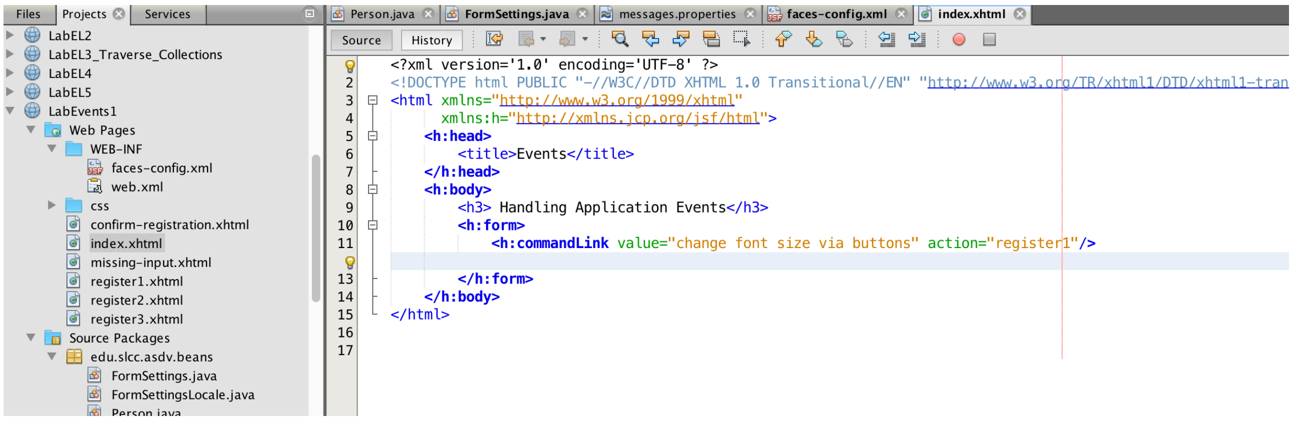
4. Create the config and put in the properties file



```xml
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-facesconfig_2_2.xsd"
    version="2.2">
  <!-- Source Packages/messages/messages.properties  -->
  <application>
    <resource-bundle>
      <base-name>messages.messages</base-name>
      <var>msgs</var>
    </resource-bundle>
  </application>
</faces-config>
```

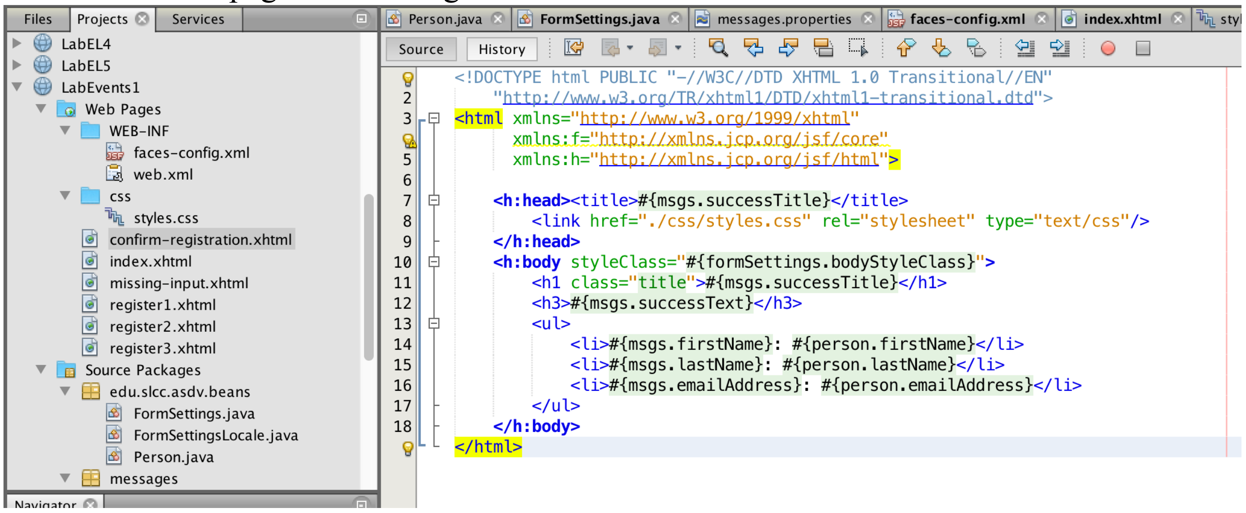
5. The css file is posted.

6. The index



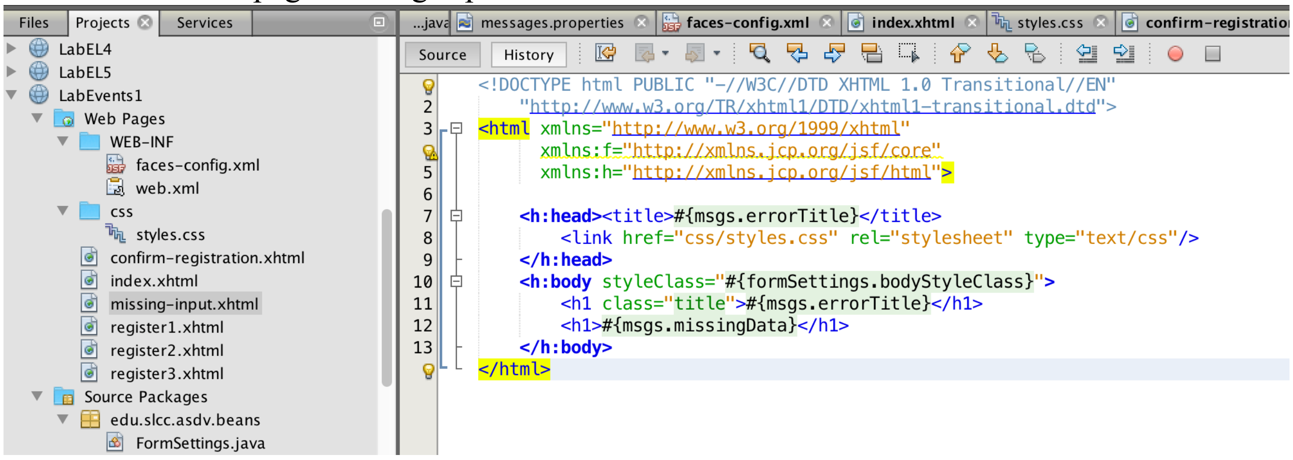
```xml
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-tran
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
    <h:head>
        <title>Events</title>
    </h:head>
    <h:body>
        <h3> Handling Application Events</h3>
        <h:form>
            <h:commandLink value="change font size via buttons" action="register1"/>

        </h:form>
    </h:body>
</html>
```

7. Create JSF page <u>confirm-registration</u>



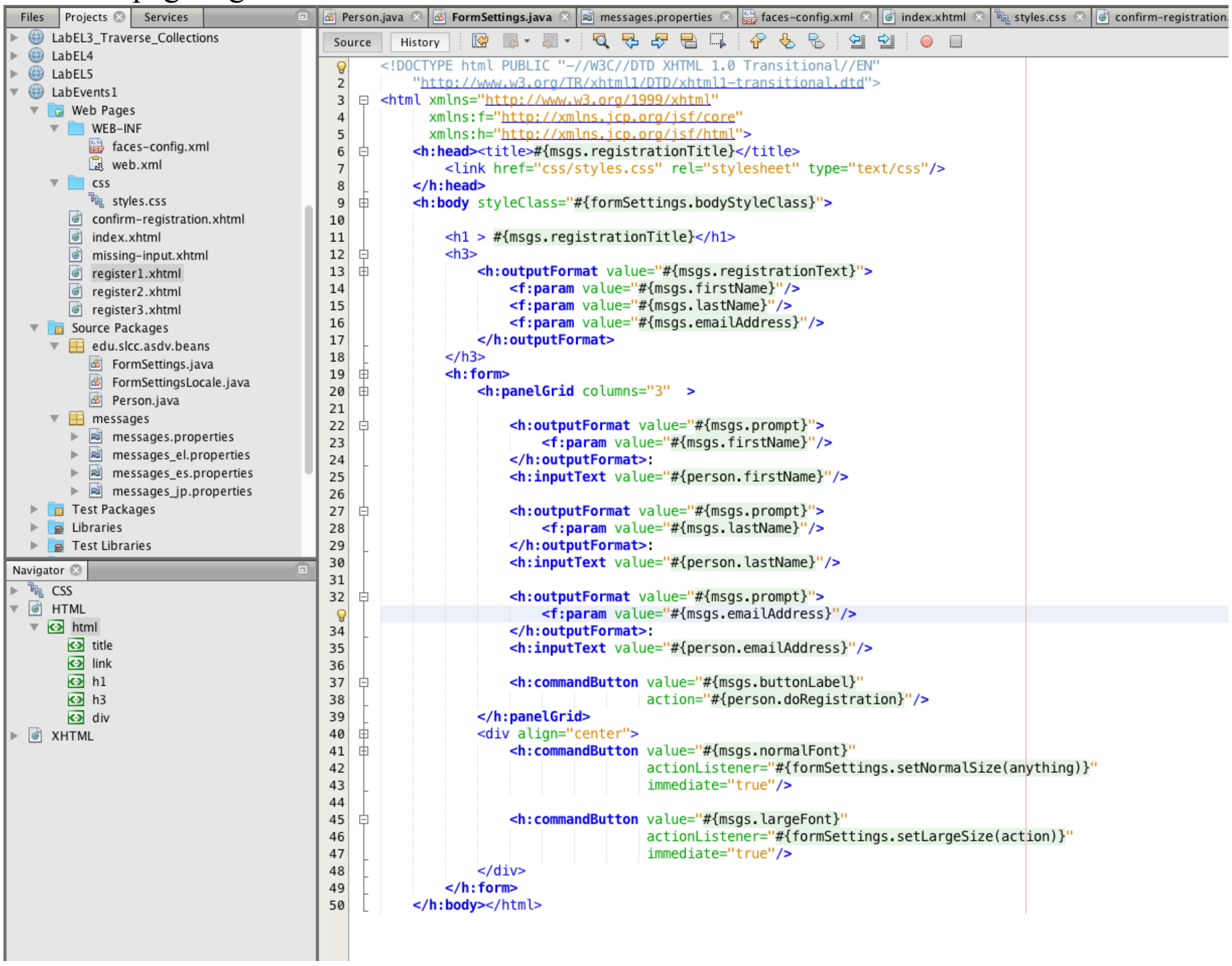
```xml
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:h="http://xmlns.jcp.org/jsf/html">

    <h:head><title>#{msgs.successTitle}</title>
        <link href="./css/styles.css" rel="stylesheet" type="text/css"/>
    </h:head>
    <h:body styleClass="#{formSettings.bodyStyleClass}">
        <h1 class="title">#{msgs.successTitle}</h1>
        <h3>#{msgs.successText}</h3>
        <ul>
            <li>#{msgs.firstName}: #{person.firstName}</li>
            <li>#{msgs.lastName}: #{person.lastName}</li>
            <li>#{msgs.emailAddress}: #{person.emailAddress}</li>
        </ul>
    </h:body>
</html>
```

8. Create JSF page missing-input



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:h="http://xmlns.jcp.org/jsf/html">

    <h:head><title>#{msgs.errorTitle}</title>
        <link href="css/styles.css" rel="stylesheet" type="text/css"/>
    </h:head>
    <h:body styleClass="#{formSettings.bodyStyleClass}">
        <h1 class="title">#{msgs.errorTitle}</h1>
        <h1>#{msgs.missingData}</h1>
    </h:body>
</html>
```

9. JSF page register1



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
    <h:head><title>#{msgs.registrationTitle}</title>
        <link href="css/styles.css" rel="stylesheet" type="text/css"/>
    </h:head>
    <h:body styleClass="#{formSettings.bodyStyleClass}">

        <h1 > #{msgs.registrationTitle}</h1>
        <h3>
            <h:outputFormat value="#{msgs.registrationText}">
                <f:param value="#{msgs.firstName}"/>
                <f:param value="#{msgs.lastName}"/>
                <f:param value="#{msgs.emailAddress}"/>
            </h:outputFormat>
        </h3>
        <h:form>
            <h:panelGrid columns="3"  >

                <h:outputFormat value="#{msgs.prompt}">
                    <f:param value="#{msgs.firstName}"/>
                </h:outputFormat>:
                <h:inputText value="#{person.firstName}"/>

                <h:outputFormat value="#{msgs.prompt}">
                    <f:param value="#{msgs.lastName}"/>
                </h:outputFormat>:
                <h:inputText value="#{person.lastName}"/>

                <h:outputFormat value="#{msgs.prompt}">
                    <f:param value="#{msgs.emailAddress}"/>
                </h:outputFormat>:
                <h:inputText value="#{person.emailAddress}"/>

                <h:commandButton value="#{msgs.buttonLabel}"
                                 action="#{person.doRegistration}"/>
            </h:panelGrid>
            <div align="center">
                <h:commandButton value="#{msgs.normalFont}"
                                 actionListener="#{formSettings.setNormalSize(anything)}"
                                 immediate="true"/>

                <h:commandButton value="#{msgs.largeFont}"
                                 actionListener="#{formSettings.setLargeSize(action)}"
                                 immediate="true"/>
            </div>
        </h:form>
    </h:body></html>
```

10. Clean and build, run.