

CURSO DE JAVASERVER FACES

EJERCICIO

MANEJO DE ACTION LISTENER EN JSF



Experiencia y Conocimiento para tu vida

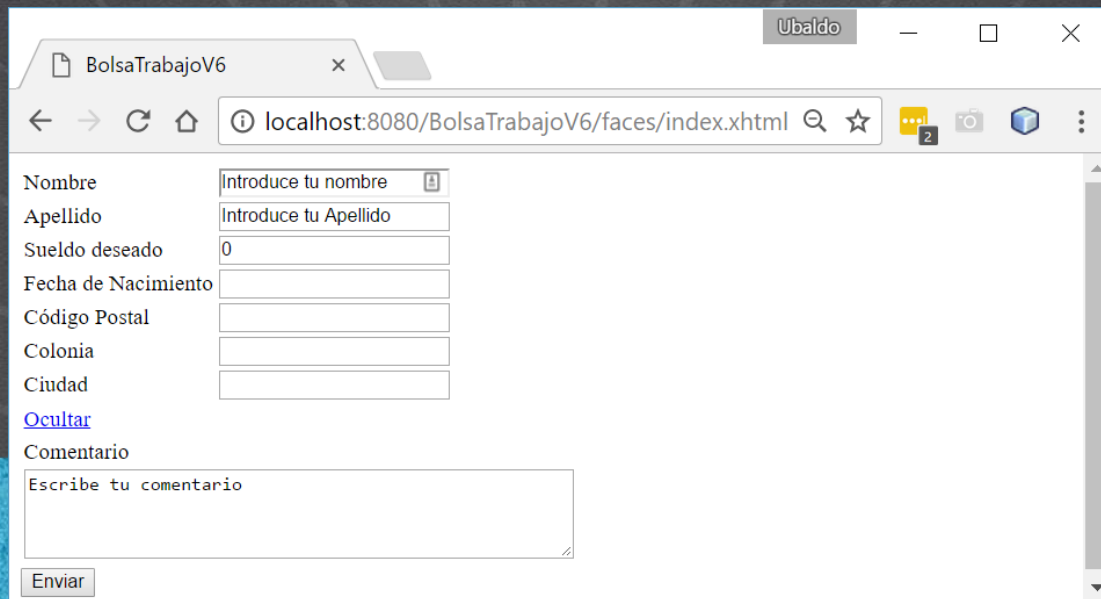
CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

El objetivo de este ejercicio es poner en práctica el concepto ActionListener.

Configuraremos un campo de comentarios, el cual se mostrará y ocultará dinámicamente asociando un evento de tipo ActionListener. El resultado se muestra a continuación:



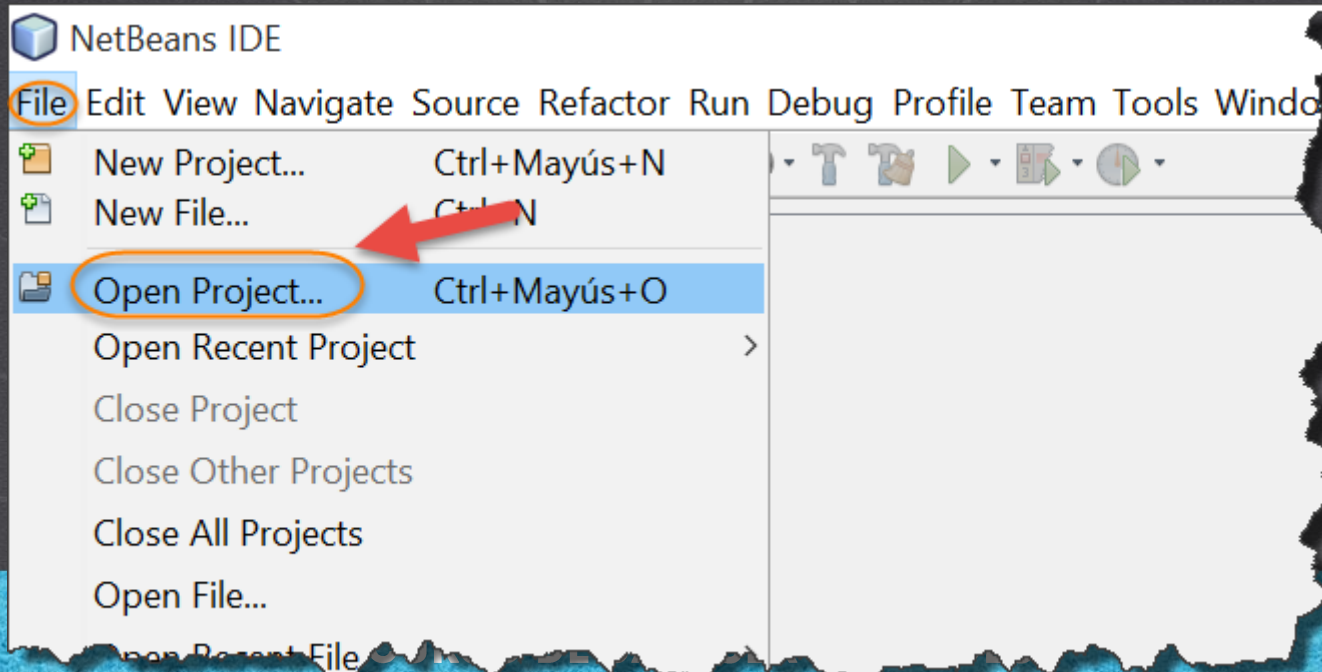
The screenshot shows a web browser window with the title 'Ubaldo'. The address bar displays 'localhost:8080/BolsaTrabajoV6/faces/index.xhtml'. The page content includes a form with the following fields and labels:

- Nombre: Introduce tu nombre
- Apellido: Introduce tu Apellido
- Sueldo deseado: 0
- Fecha de Nacimiento:
- Código Postal:
- Colonia:
- Ciudad:

Below these fields is a blue link labeled 'Ocultar'. Underneath the link is a text area labeled 'Comentario' with the placeholder text 'Escribe tu comentario'. At the bottom left of the form is a button labeled 'Enviar'.

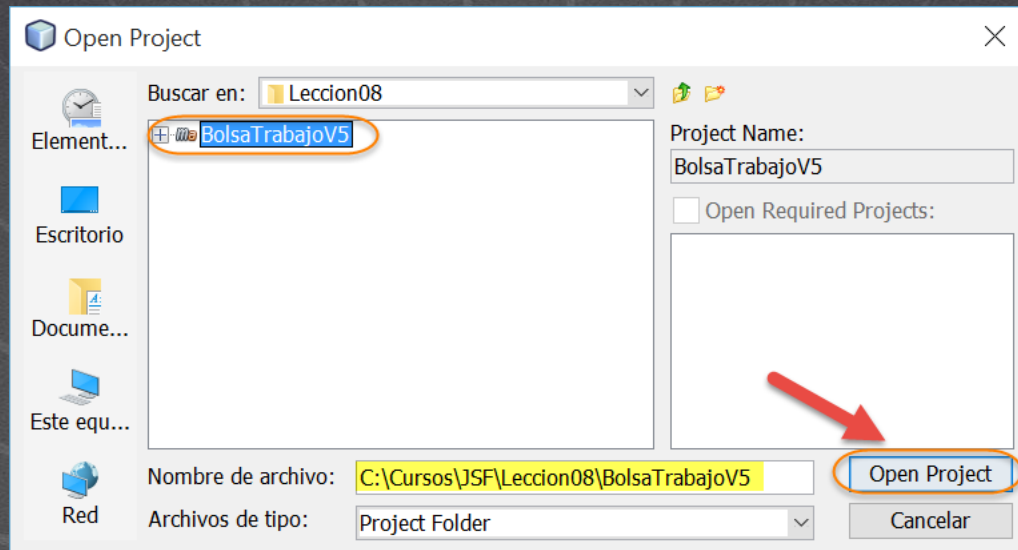
PASO 1. ABRIR EL PROYECTO

Abrimos el proyecto BolsaTrabajoV5, sólo en caso de que no lo tengamos ya abierto:



PASO 1. ABRIR EL PROYECTO

Abrimos el proyecto BolsaTrabajoV5, sólo en caso de que no lo tengamos ya abierto:

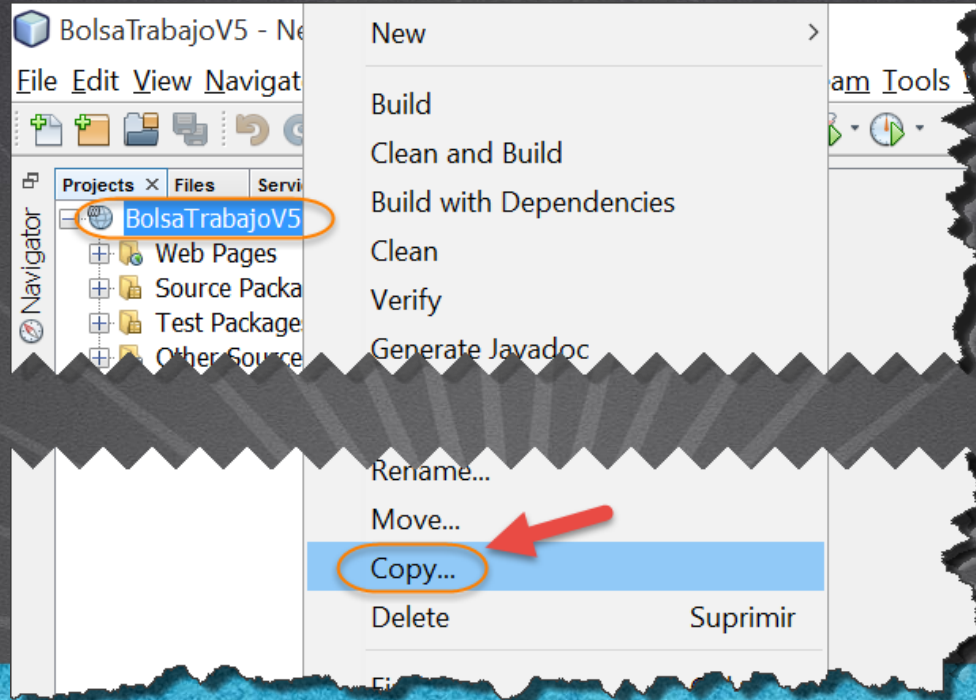


CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 2. COPIAR EL PROYECTO

Copiamos el proyecto BolsaTrabajoV5:

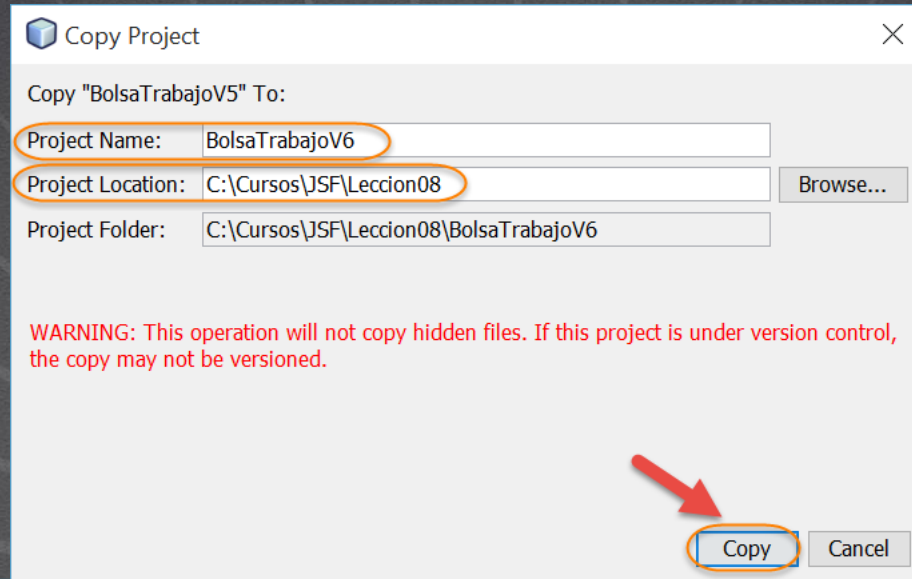


CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 2. COPIAR EL PROYECTO

Copiamos el proyecto BolsaTrabajoV5 y lo renombramos a BolsaTrabajoV6 y lo depositamos en la nueva ruta mostrada:

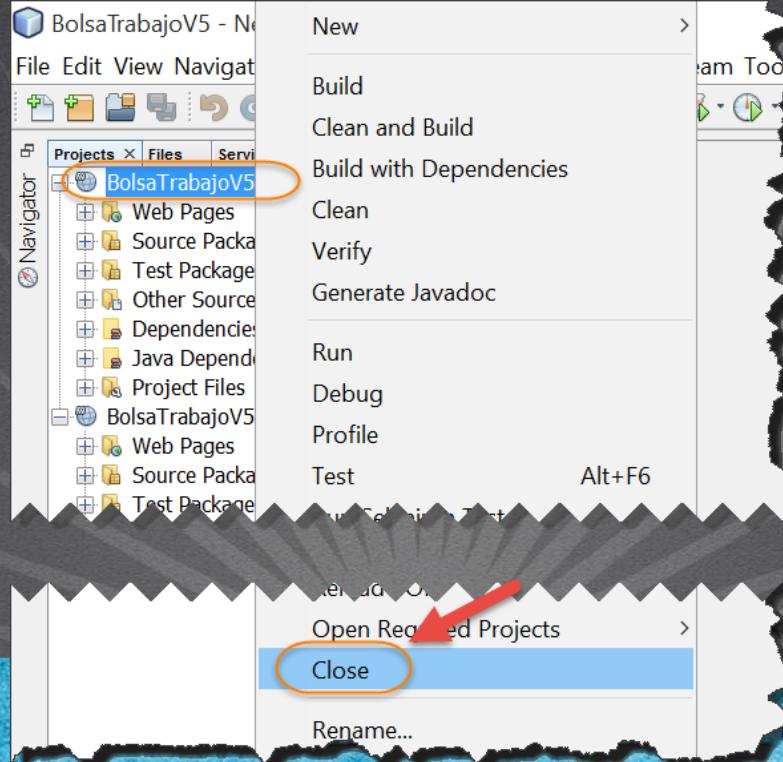


CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

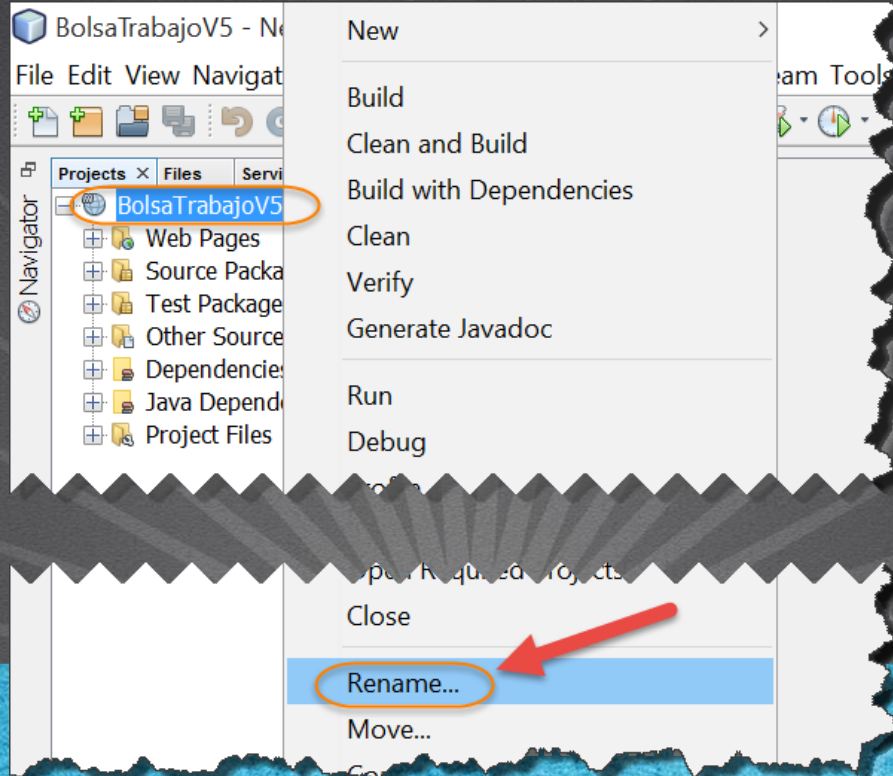
PASO 3. CERRAR PROYECTO

Cerramos el proyecto anterior, y dejamos solo el nuevo abierto:



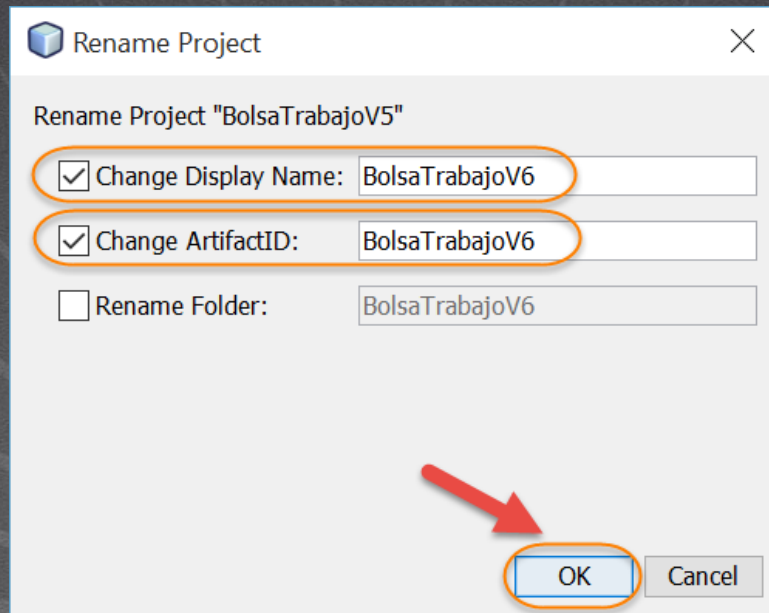
PASO 4. RENOMBRAR PROYECTO

Cambiamos el nombre del proyecto a BolsaTrabajoV6:



PASO 4. RENOMBRAR PROYECTO

Cambiamos el nombre del proyecto a BolsaTrabajoV6:



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 5. MODIFICAR UNA CLASE JAVA

Modificamos la clase Candidato.java, agregamos el campo de comentario:

```
private String comentario="Escribe tu comentario";
```

Generar los métodos getters y setters para este nuevo campo.



Experiencia y Conocimiento para tu vida

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Candidato.java:

Dar click para ir al código

```
package beans.model;

import java.util.Date;
import javax.inject.Named;
import javax.enterprise.context.RequestScoped;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

@RequestScoped
@Named
public class Candidato {

    Logger log = LogManager.getRootLogger();

    private String nombre = "Introduce tu nombre";
    private String apellido = "Introduce tu Apellido";
    private int sueldoDeseado;
    private Date fechaNacimiento;
    private String codigoPostal;
    private String colonia;
    private String ciudad;
    private String comentario = "Escribe tu comentario";

    public Candidato() {
        log.info("Creando el objeto Candidato");
    }
}
```

PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Candidato.java:

Dar click para ir al código

```
public String getNombre() {  
    return nombre;  
}  
  
public void setNombre(String nombre) {  
    this.nombre = nombre;  
    log.info("Modificando la propiedad nombre:" + this.nombre);  
}  
  
public String getApellido() {  
    return apellido;  
}  
  
public void setApellido(String apellido) {  
    this.apellido = apellido;  
    log.info("Modificando la propiedad apellido:" + this.apellido);  
}  
  
public int getSueldoDeseado() {  
    return sueldoDeseado;  
}  
  
public void setSueldoDeseado(int sueldoDeseado) {  
    this.sueldoDeseado = sueldoDeseado;  
    log.info("Modificando la propiedad sueldoDeseado:" + this.sueldoDeseado);  
}
```


PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Candidato.java:

Dar click para ir al código

```
public Date getFechaNacimiento() {  
    return fechaNacimiento;  
}  
  
public void setFechaNacimiento(Date fechaNacimiento) {  
    this.fechaNacimiento = fechaNacimiento;  
    log.info("Modificando la propiedad fechaNacimiento:" + this.fechaNacimiento);  
}  
  
public String getCodigoPostal() {  
    return codigoPostal;  
}  
  
public void setCodigoPostal(String codigoPostal) {  
    this.codigoPostal = codigoPostal;  
}  
  
public String getColonia() {  
    return colonia;  
}  
  
public void setColonia(String colonia) {  
    this.colonia = colonia;  
}  
  
public String getCiudad() {  
    return ciudad;  
}
```

PASO 5. MODIFICAMOS EL CÓDIGO

Archivo Candidato.java:

Dar click para ir al código

```
public void setCiudad(String ciudad) {  
    this.ciudad = ciudad;  
}  
  
public String getComentario() {  
    return comentario;  
}  
  
public void setComentario(String comentario) {  
    this.comentario = comentario;  
}  
}
```


PASO 6. MODIFICAR UN ARCHIVO XHTML

Modificamos el archivo index.xhtml para ocultar/mostrar el campo de comentario:

```
<table>
  <tr>
    <td>
      <h:commandLink actionListener="#{vacanteForm.ocultarComentario}"
                    immediate="true" rendered="#{!vacanteForm.comentarioEnviado}"
                    value="#{msgs['vacanteForm.mostrar']}" />
      <h:commandLink actionListener="#{vacanteForm.ocultarComentario}"
                    immediate="true" rendered="#{vacanteForm.comentarioEnviado}"
                    value="#{msgs['vacanteForm.ocultar']}" />
    </td>
  </tr>
</table>
```

PASO 6. MODIFICAR UN ARCHIVO XHTML

Modificamos el archivo index.xhtml para agregar el campo de comentario:

```
<h:panelGroup rendered="#{vacanteForm.comentarioEnviado}">
  <table>
    <tr>
      <td><h:outputLabel for="comentario" value="#{msgs['vacanteForm.comentario']}" /></td>
    </tr>
    <tr>
      <td><h:inputTextarea id="comentario" cols="50" rows="10"
value="#{candidato.comentario}" /></td>
    </tr>
  </table>
</h:panelGroup>
```

PASO 6. MODIFICAMOS EL CÓDIGO

Archivo index.xhtml:

Dar click para ir al código

```
<?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-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core">
  <h:head>
    <title>BolsaTrabajoV6</title>
  </h:head>
  <h:body>
    <h:form id="vacanteForm">
      <h:messages globalOnly="true" ></h:messages>
      <table>
        <tr>
          <td><h:outputLabel for="nombre" value="#{msgs['vacanteForm.nombre']}" /></td>
          <td><h:inputText id="nombre" required="true" value="#{candidato.nombre}" /> </td>
          <td><h:message for="nombre" /></td>
        </tr>
        <tr>
          <td><h:outputLabel for="apellido" value="#{msgs['vacanteForm.apellido']}" /></td>
          <td><h:inputText id="apellido" required="true" value="#{candidato.apellido}" /></td>
          <td><h:message for="apellido" /></td>
        </tr>
        <tr>
          <td><h:outputLabel for="sueldoDeseado" value="#{msgs['vacanteForm.sueldoDeseado']}" /></td>
          <td><h:inputText id="sueldoDeseado" required="true"
                    value="#{candidato.sueldoDeseado}"
                    <f:validateLongRange minimum="10000" maximum="50000" />
          </td>
          <td><h:message for="sueldoDeseado" /></td>
        </tr>
      </table>
    </h:form>
  </h:body>
</html>
```


PASO 6. MODIFICAMOS EL CÓDIGO

[Archivo index.xhtml:](#)

Dar click para ir al código

```
<tr>
  <td><h:outputLabel for="fechaNacimiento" value="#{msgs['vacanteForm.nacimiento']}" /></td>
  <td><h:inputText id="fechaNacimiento" required="true"
    value="#{candidato.fechaNacimiento}"
    <f:convertDateTime pattern="MM/dd/yyyy" />
  </h:inputText></td>
  <td><h:message for="fechaNacimiento" /></td>
</tr>
<tr>
  <td><h:outputLabel for="codigoPostal" value="#{msgs['vacanteForm.codigoPostal']}" /></td>
  <td><h:inputText id="codigoPostal" immediate="true"
    onchange="this.form.submit()" required="true"
    value="#{candidato.codigoPostal}"
    valueChangeListener="#{vacanteForm.codigoPostalListener}" /></td>
  <td><h:message for="codigoPostal" /></td>
</tr>
<tr>
  <td><h:outputLabel for="colonia" value="#{msgs['vacanteForm.colonia']}" /></td>
  <td><h:inputText id="colonia" required="true"
    value="#{candidato.colonia}" /></td>
  <td><h:message for="colonia" /></td>
</tr>
<tr>
  <td><h:outputLabel for="ciudad" value="#{msgs['vacanteForm.ciudad']}" /></td>
  <td><h:inputText id="ciudad" required="true"
    value="#{candidato.ciudad}" /></td>
  <td><h:message for="ciudad" /></td>
</tr>
</table>
```

PASO 6. MODIFICAMOS EL CÓDIGO

Archivo index.xhtml:

Dar click para ir al código

```
<table>
  <tr>
    <td>
      <h:commandLink actionListener="#{vacanteForm.ocultarComentario}"
                     immediate="true" rendered="#{!vacanteForm.comentarioEnviado}"
                     value="#{msgs['vacanteForm.mostrar']}" />
      <h:commandLink actionListener="#{vacanteForm.ocultarComentario}"
                     immediate="true" rendered="#{vacanteForm.comentarioEnviado}"
                     value="#{msgs['vacanteForm.ocultar']}" />
    </td>
  </tr>
</table>
<h:panelGroup rendered="#{vacanteForm.comentarioEnviado}">
  <table>
    <tr>
      <td><h:outputLabel for="comentario" value="#{msgs['vacanteForm.comentario']}" /></td>
    </tr>
    <tr>
      <td><h:inputTextarea id="comentario" cols="50" rows="10" value="#{candidato.comentario}" /></td>
    </tr>
  </table>
</h:panelGroup>
<h:commandButton action="#{vacanteForm.enviar}"
                  value="#{msgs['vacanteForm.enviar']}" />
</h:form>
</h:body>
</html>
```

PASO 7. MODIFICAR UNA CLASE JAVA

Modificamos la clase VacanteForm.java para procesar el método action para mostrar/ocultar el campo de comentario.

Agregue el campo siguiente para VacanteForm.java:

```
private boolean comentarioEnviado= false;
```

Agregamos los métodos getter/setter para este nuevo campo.

PASO 7. MODIFICAR UNA CLASE JAVA

Agregar el método siguiente en la clase VacanteForm.java:

```
public void ocultarComentario (ActionEvent actionEvent) {  
    this.comentarioEnviado = !this.comentarioEnviado;  
}
```

Este es el método ActionListener que se llama cuando el usuario hace clic en el link de mostrar / ocultar. El ActionListener simplemente oculta todos los componentes del h: panelGroup.

Y agregar las siguientes librería a la clase VacanteForm.java:

```
import javax.faces.event.ActionEvent;
```

PASO 7. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
package beans.backing;

import beans.model.Candidato;
import javax.inject.Named;
import javax.enterprise.context.RequestScoped;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIInput;
import javax.faces.component.UIViewRoot;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.event.ValueChangeEvent;
import javax.inject.Inject;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
```

```
@RequestScoped
@Named
public class VacanteForm {

    Logger log = LogManager.getRootLogger();

    private boolean comentarioEnviado = false;

    @Inject
    private Candidato candidato;
```

PASO 7. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
public VacanteForm() {
    log.info("Creando objeto VacanteForm");
}

public void setCandidato(Candidato candidato) {
    this.candidato = candidato;
}

//Metodo de flujo de control
public String enviar() {
    System.out.println("enviar() Nombre=" + this.candidato.getNombre());
    System.out.println("enviar() Apellido=" + this.candidato.getApellido());
    System.out.println("enviar() Sueldo deseado" + this.candidato.getSueldoDeseado());
    if (this.candidato.getNombre().equals("Juan")) {

        if (this.candidato.getApellido().equals("Perez")) {
            String msg = "Gracias, pero Juan Perez ya trabaja con nosotros.";
            FacesMessage facesMessage = new FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
            FacesContext facesContext = FacesContext.getCurrentInstance();
            String clientId = null; //Este es un mensaje global
            facesContext.addMessage(clientId, facesMessage);
            return "index";
        }
        return "exito"; //exito.xhtml
    } else {
        return "fallo"; //fallo.xhtml
    }
}
```


PASO 7. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
//Metodo de tipo Value Change Listener
public void codigoPostalListener(ValueChangeEvent valueChangeEvent) {
    FacesContext facesContext = FacesContext.getCurrentInstance();
    UIViewRoot uiViewRoot = facesContext.getViewRoot();
    String newCodigoPostal = (String) valueChangeEvent.getNewValue();
    if ("03810".equals(newCodigoPostal)) {
        log.info("Modificamos los valores de colonia y ciudad dinamicamente con ValueChangeListener");
        //Utilizamos el nombre del form de index.xhtml para encontrar el componente
        UIInput coloniaInputText = (UIInput) uiViewRoot.findComponent("vacanteForm:colonia");
        String colonia = "Nápoles";
        coloniaInputText.setValue(colonia);
        coloniaInputText.setSubmittedValue(colonia);
        UIInput ciudadInputText = (UIInput) uiViewRoot.findComponent("vacanteForm:ciudad");
        String ciudad = "Ciudad de Mexico";
        ciudadInputText.setValue(ciudad);
        ciudadInputText.setSubmittedValue(ciudad);
        facesContext.renderResponse();
    }
}

public void ocultarComentario(ActionEvent actionEvent) {
    this.comentarioEnviado = !this.comentarioEnviado;
}

public boolean isComentarioEnviado() {
    return comentarioEnviado;
}

public void setComentarioEnviado(boolean comentarioEnviado) {
    this.comentarioEnviado = comentarioEnviado;
}
}
```

PASO 8. MODIFICAR UN ARCHIVO PROPERTIES

Modificamos el archivo de mensajes.properties y mensajes_en.properties para agregar las etiquetas del nuevo campo:

```
vacanteForm.comentario = Comentario  
vacanteForm.mostrar = Mostrar  
vacanteForm.ocultar = Ocultar
```

PASO 8. MODIFICAMOS EL CÓDIGO

Archivo mensajes.properties:

Dar click para ir al código

```
vacanteForm.nombre=Nombre  
vacanteForm.apellido=Apellido  
vacanteForm.sueldoDeseado=Sueldo deseado  
vacanteForm.nacimiento=Fecha de Nacimiento  
vacanteForm.codigoPostal=Código Postal  
vacanteForm.colonia=Colonia  
vacanteForm.ciudad=Ciudad  
vacanteForm.enviar=Enviar  
vacanteForm.comentario=Comentario  
vacanteForm.mostrar=Mostrar  
vacanteForm.ocultar=Ocultar
```


PASO 9. MODIFICAMOS EL CÓDIGO

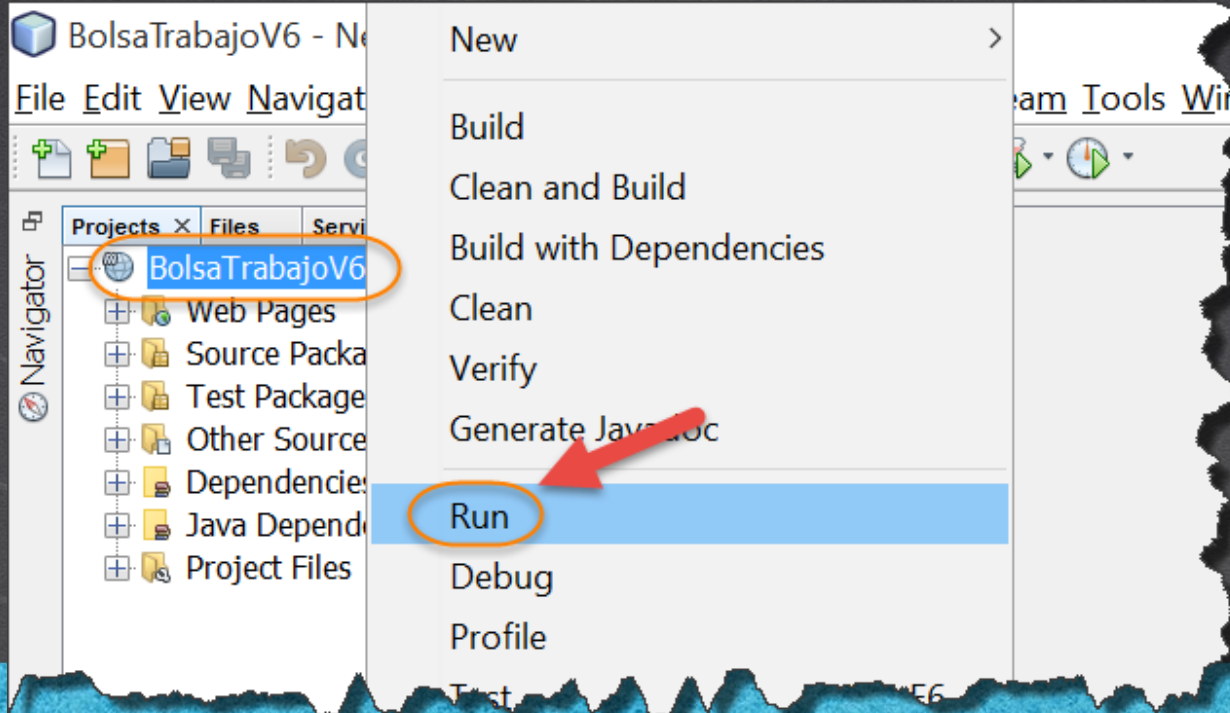
Archivo mensajes_en.properties:

Dar click para ir al código

```
vacanteForm.nombre=Name  
vacanteForm.apellido=Last Name  
vacanteForm.sueldoDeseado=Salary desired  
vacanteForm.nacimiento=Date of birth  
vacanteForm.codigoPostal=Zip Code  
vacanteForm.colonia=Suburb  
vacanteForm.ciudad=City  
vacanteForm.enviar=Submit  
vacanteForm.comentario=Annotation  
vacanteForm.mostrar=Show  
vacanteForm.ocultar=Hide
```

PASO 10. EJECUTAMOS EL PROYECTO

Ejecutamos el proyecto:



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 10. EJECUTAMOS EL PROYECTO

Ejecutamos el proyecto:

Ubaldo

BolsaTrabajoV6

localhost:8080/BolsaTrabajoV6/faces/index.xhtml

Nombre: Introduce tu nombre

Apellido: Introduce tu Apellido

Sueldo deseado: 0

Fecha de Nacimiento:

Código Postal: Valor Requerido

Colonia:

Ciudad:

[Ocultar](#)

Comentario: Escribe tu comentario

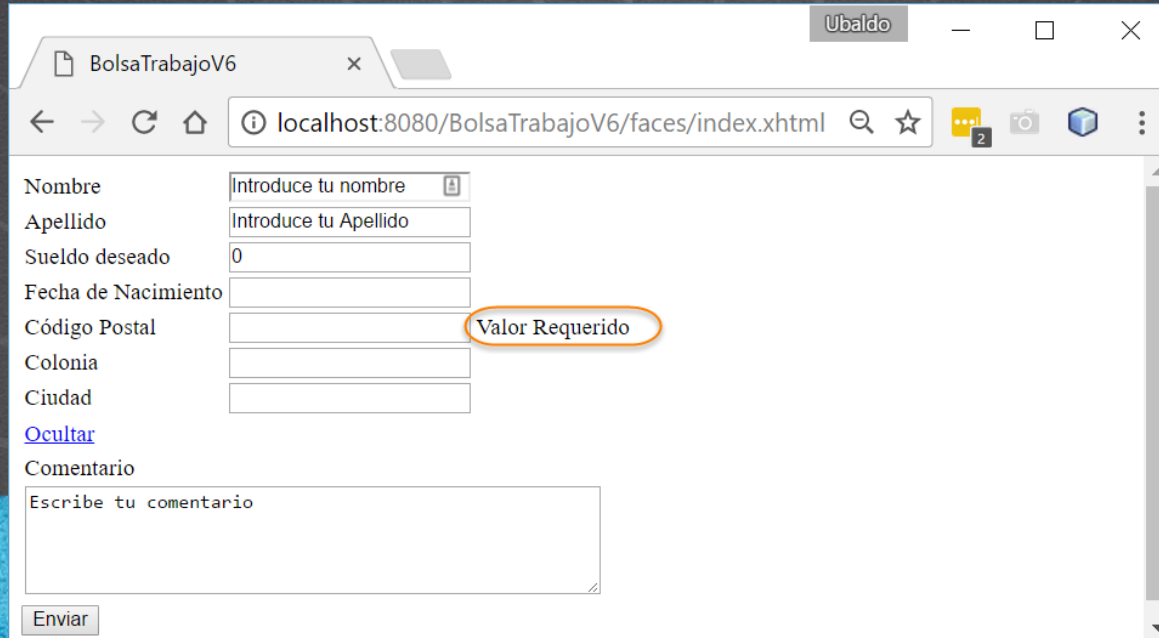
Enviar

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 10. EJECUTAMOS EL PROYECTO

Observar que al presionar el botón de mostrar hay un comportamiento indeseable en el campo del Código Postal. A continuación corregiremos este comportamiento:



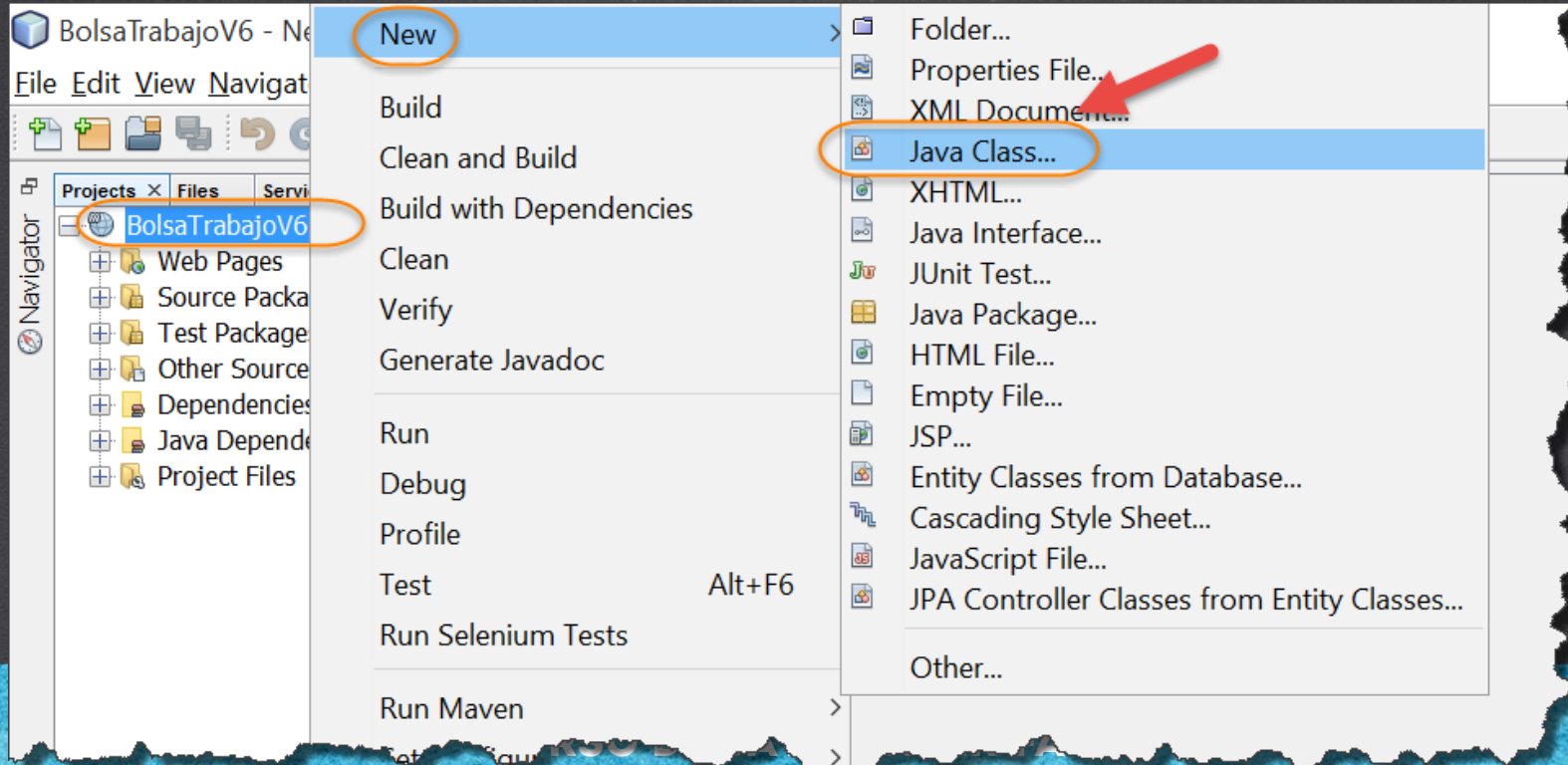
The screenshot shows a web browser window with the title 'BolsaTrabajoV6'. The address bar displays 'localhost:8080/BolsaTrabajoV6/faces/index.xhtml'. The form contains the following fields:

- Nombre:
- Apellido:
- Sueldo deseado:
- Fecha de Nacimiento:
- Código Postal: (highlighted with an orange circle and the text 'Valor Requerido')
- Colonia:
- Ciudad:
- [Ocultar](#)
- Comentario:

An 'Enviar' button is located at the bottom left of the form.

PASO 11. CREAR UNA CLASE JAVA

Creamos la clase FacesContextHelper.java



PASO 11. CREAR UNA CLASE JAVA

Creamos la clase FacesContextHelper.java

New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name: FacesContextHelper

Project: BolsaTrabajoV6

Location: Source Packages

Package: beans.helper

Created File: F:\Leccion08\BolsaTrabajoV6\src\main\java\beans\helper\FacesContextHelper.java

< Back Next > **Finish** Cancel Help

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 12. MODIFICAMOS EL CÓDIGO

Archivo FacesContextHelper.java:

Dar click para ir al código

```
package beans.helper;

import java.util.Iterator;
import java.util.List;
import javax.faces.application.FacesMessage;
import javax.faces.component.ActionSource;
import javax.faces.component.EditableValueHolder;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;

public class FacesContextHelper {

    public static void limpiarFacesMessages(FacesContext facesContext, String clientId) {
        Iterator<FacesMessage> facesMessages = facesContext.getMessages(clientId);
        while (facesMessages.hasNext()) {
            facesMessages.next();
            facesMessages.remove();
        }
    }

    public static void limpiarImmediateFacesMessages(FacesContext facesContext) {
        limpiarImmediateFacesMessages(facesContext, facesContext.getViewRoot());
    }
}
```

PASO 12. MODIFICAMOS EL CÓDIGO

Archivo FacesContextHelper.java:

Dar click para ir al código

```
public static void limpiarImmediateFacesMessages(FacesContext facesContext, UIComponent uiComponent) {  
    if (uiComponent instanceof ActionSource) {  
        ActionSource actionSource = (ActionSource) uiComponent;  
        if (actionSource.isImmediate()) {  
            limpiarFacesMessages(facesContext, uiComponent.getClientId(facesContext));  
        }  
    } else if (uiComponent instanceof EditableValueHolder) {  
        EditableValueHolder editableValueHolder = (EditableValueHolder) uiComponent;  
        if (editableValueHolder.isImmediate()) {  
            limpiarFacesMessages(facesContext, uiComponent.getClientId(facesContext));  
        }  
    }  
    List<UIComponent> childComponents = uiComponent.getChildren();  
    for (UIComponent childComponent : childComponents) {  
        limpiarImmediateFacesMessages(facesContext, childComponent);  
    }  
}
```

PASO 13. MODIFICAR UNA CLASE JAVA

Modificamos la clase VacanteForm.java. VacanteForm.java sustituiremos el código mostrado:

```
public void ocultarComentario(ActionEvent actionEvent) {  
    this.ocultarComentario = !this.ocultarComentario;  
}
```

Por el siguiente código:

```
public void ocultarComentario(ActionEvent actionEvent) {  
    this.comentarioEnviado = !this.comentarioEnviado;  
    FacesContext facesContext = FacesContext.getCurrentInstance();  
    FacesContextHelper.limpiarImmediateFacesMessages(facesContext);  
}
```

Importar:

```
beans.helper.FacesContextHelper
```


PASO 14. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
package beans.backing;

import beans.helper.FacesContextHelper;
import beans.model.Candidato;
import javax.inject.Named;
import javax.enterprise.context.RequestScoped;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIInput;
import javax.faces.component.UIViewRoot;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.event.ValueChangeEvent;
import javax.inject.Inject;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
```

```
@RequestScoped
@Named
public class VacanteForm {

    Logger log = LogManager.getRootLogger();

    private boolean comentarioEnviado = false;

    @Inject
    private Candidato candidato;
```

PASO 14. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
public VacanteForm() {
    log.info("Creando objeto VacanteForm");
}

public void setCandidato(Candidato candidato) {
    this.candidato = candidato;
}

//Metodo de flujo de control
public String enviar() {

    System.out.println("enviar() Nombre=" + this.candidato.getNombre());
    System.out.println("enviar() Apellido=" + this.candidato.getApellido());
    System.out.println("enviar() Sueldo deseado" + this.candidato.getSueldoDeseado());
    if (this.candidato.getNombre().equals("Juan")) {

        if (this.candidato.getApellido().equals("Perez")) {
            String msg = "Gracias, pero Juan Perez ya trabaja con nosotros.";
            FacesMessage facesMessage = new FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
            FacesContext facesContext = FacesContext.getCurrentInstance();
            String clientId = null; //Este es un mensaje global
            facesContext.addMessage(clientId, facesMessage);
            return "index";
        }
        return "exito";//exito.xhtml
    } else {
        return "fallo"; //fallo.xhtml
    }
}
```

PASO 14. MODIFICAMOS EL CÓDIGO

Archivo VacanteForm.java:

Dar click para ir al código

```
//Metodo de tipo Value Change Listener
public void codigoPostalListener(ValueChangeEvent valueChangeEvent) {
    FacesContext facesContext = FacesContext.getCurrentInstance();
    UIViewRoot uiViewRoot = facesContext.getViewRoot();
    String newCodigoPostal = (String) valueChangeEvent.getNewValue();
    if ("03810".equals(newCodigoPostal)) {
        log.info("Modificamos los valores de colonia y ciudad dinamicamente con ValueChangeListener");
        //Utilizamos el nombre del form de index.xhtml para encontrar el componente
        UIInput coloniaInputText = (UIInput) uiViewRoot.findComponent("vacanteForm:colonia");
        String colonia = "Nápoles";
        coloniaInputText.setValue(colonia);
        coloniaInputText.setSubmittedValue(colonia);

        UIInput ciudadInputText = (UIInput) uiViewRoot.findComponent("vacanteForm:ciudad");
        String ciudad = "Ciudad de Mexico";
        ciudadInputText.setValue(ciudad);
        ciudadInputText.setSubmittedValue(ciudad);

        facesContext.renderResponse();
    }
}

public void ocultarComentario(ActionEvent actionEvent) {
    this.comentarioEnviado = !this.comentarioEnviado;
    log.info("Mostrando/ocultando el comentario");
    FacesContext facesContext = FacesContext.getCurrentInstance();
    FacesContextHelper.limpiarImmediateFacesMessages(facesContext);
}
```


PASO 14. MODIFICAMOS EL CÓDIGO

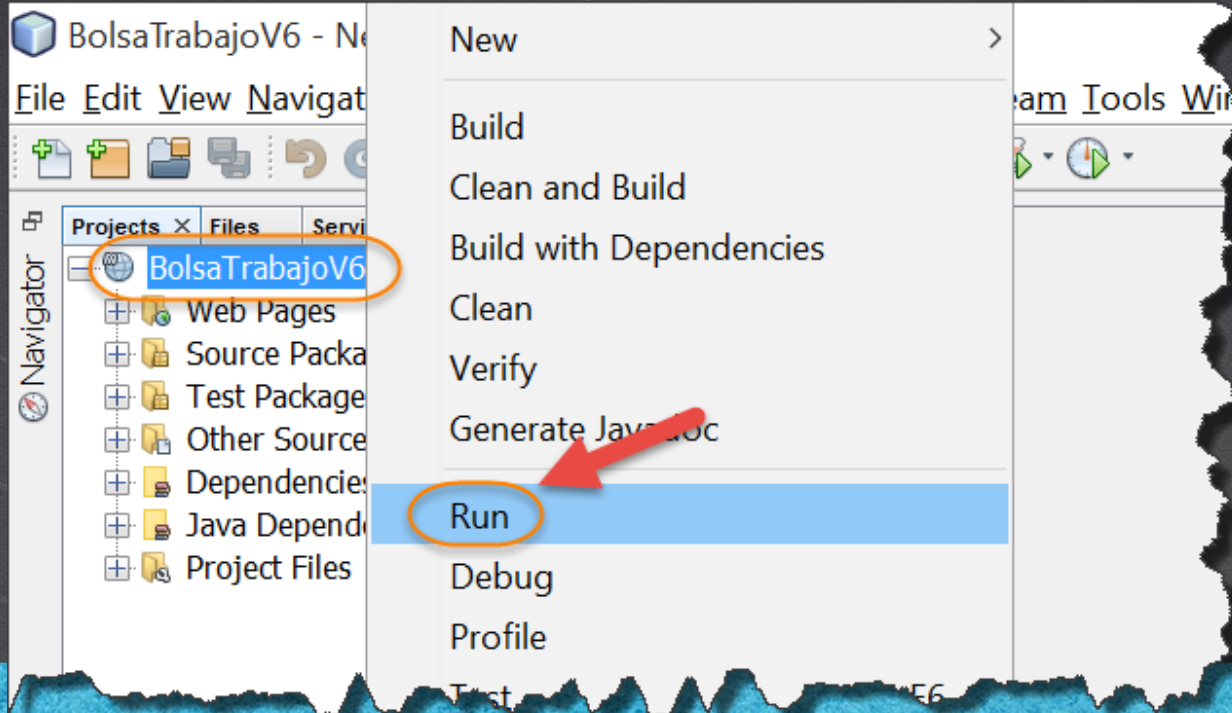
Archivo VacanteForm.java:

Dar click para ir al código

```
public boolean isComentarioEnviado() {  
    return comentarioEnviado;  
}  
  
public void setComentarioEnviado(boolean comentarioEnviado) {  
    this.comentarioEnviado = comentarioEnviado;  
}  
}
```

PASO 15. EJECUTAMOS EL PROYECTO

Ejecutamos el proyecto:

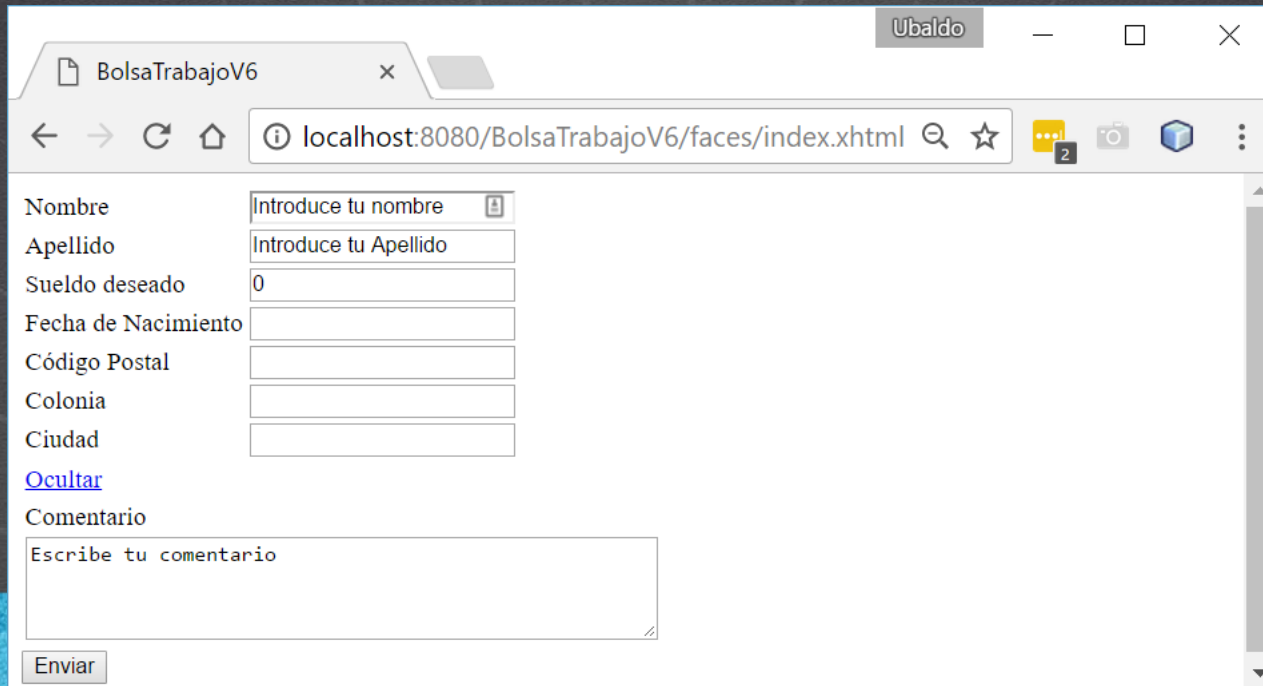


CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

PASO 15. EJECUTAMOS EL PROYECTO

Observar que al presionar el botón de mostrar ya no se muestra nada en el campo del Código Postal:



Ubaldo

BolsaTrabajoV6

localhost:8080/BolsaTrabajoV6/faces/index.xhtml

Nombre

Apellido

Sueldo deseado

Fecha de Nacimiento

Código Postal

Colonia

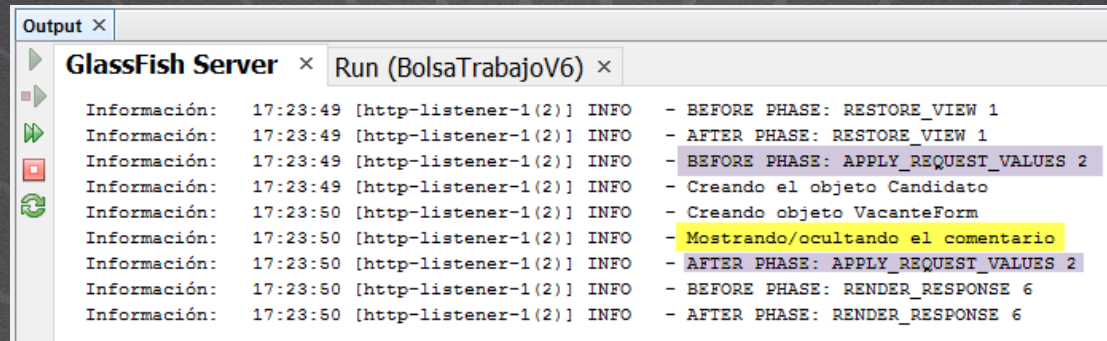
Ciudad

[Ocultar](#)

Comentario

TAREAS EXTRAS

- Realizar la prueba del ejercicio y observar la salida del archivo log para analizar lo correspondiente a las fases del ciclo de vida JSF.
- El resultado debe ser similar a lo siguiente:



The screenshot shows the Eclipse IDE's Output window with two tabs: 'GlassFish Server' and 'Run (BolsaTrabajoV6)'. The 'Run' tab is active, displaying a log of JSF lifecycle events. The log entries are as follows:

Timestamp	Source	Level	Message
17:23:49	[http-listener-1(2)]	INFO	- BEFORE PHASE: RESTORE_VIEW 1
17:23:49	[http-listener-1(2)]	INFO	- AFTER PHASE: RESTORE_VIEW 1
17:23:49	[http-listener-1(2)]	INFO	- BEFORE PHASE: APPLY_REQUEST_VALUES 2
17:23:49	[http-listener-1(2)]	INFO	- Creando el objeto Candidato
17:23:50	[http-listener-1(2)]	INFO	- Creando objeto VacanteForm
17:23:50	[http-listener-1(2)]	INFO	- Mostrando/ocultando el comentario
17:23:50	[http-listener-1(2)]	INFO	- AFTER PHASE: APPLY_REQUEST_VALUES 2
17:23:50	[http-listener-1(2)]	INFO	- BEFORE PHASE: RENDER_RESPONSE 6
17:23:50	[http-listener-1(2)]	INFO	- AFTER PHASE: RENDER_RESPONSE 6

CONCLUSIÓN DEL EJERCICIO

- Con este ejercicio hemos agregado el concepto de ActionListener con JSF.
- Agregamos un nuevo campo llamado comentario, el cual se muestra u oculta dependiendo de la acción solicitada, y esto lo hemos procesado por medio de un método de tipo ActionListener.
- Con esto ya tenemos las bases del manejo de acciones con JSF.



Experiencia y Conocimiento para tu vida

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

CURSO ONLINE

JAVASERVER FACES (JSF)

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx