Aplicaciones CRUD

Facultad de Ingeniería y Ciencias Naturales
Ingeniería en Sistemas Computacionales
Desarrollo del Software I Guía 7

I- Objetivos

- Conocer las aplicaciones CRUD
- Crear programas de mantenimiento a tablas maestras.

II- Contenido

Introducción

Aparte de mostrar datos, las aplicaciones también tienen que manipularlos: realizar inserciones, actualizaciones y borrados. Por tanto, a la hora de implementar la interfaz de usuario, una de las tareas más habituales con las que ha de enfrentarse un programador es la creación de formularios de manipulación de datos, también conocidos como formularios CRUD porque esas son las iníciales en inglés de las cuatro operaciones básicas que realizan (creación, lectura, actualización y borrado de datos). Dada su importancia práctica, este es el tema que abarca esta guía.

CRUD es el acrónimo de Crear, Leer, Actualizar y Borrar (del original en inglés: Create, Read, Update and Delete). Es usado para referirse a las funciones básicas en bases de datos o la capa de persistencia en un software.

Requerimientos:

- ✓ Clases de los paquetes de lógica de negocios
- ✓ Inicio de sesión, menú y cierre de sesión

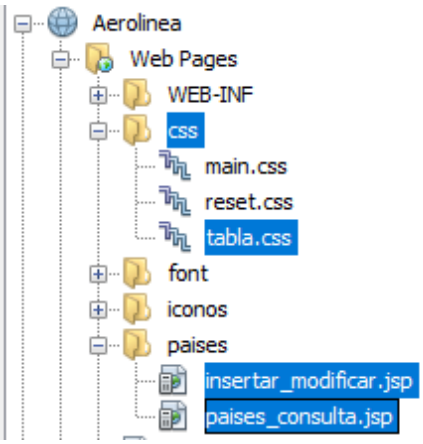
Paso 1. Utilizar directorios absolutos del proyecto y desde la BD.

La _top.jsp ya posee directorios absolutos como se mostró en la guía anterior con el uso de la siguiente expresión EL:

```
${pageContext.servletContext.contextPath}${menu.url}
```

Ejemplo 1. Aplicación CRUD para países

Crear los siguientes archivos a continuación señalados en el proyecto Aerolinea creado en la guía 6



Paso 2. Crear formulario para agregar/modificar países

Crear un folder llamado **paises** dentro del directorio del proyecto **Web Pages** y dentro crear la siguiente página

Página: insertar_modificar.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@include file="../_top.jsp"%>
<h1>Paises</h1>
<br/>
<form name="form_paises" onsubmit="return validar();"
action="${pageContext.servletContext.contextPath}/Paises?accion=insertar_modificar"
method="POST">
   <table border="0" id="table">
```

```
  <thead>
    <tr>
      <th colspan="">Complete la información<br><br></th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>ID País</td>
      <td><input type="text" name="txtIdpais" value="${pais.idpais}" readonly="readonly" /></td>
    </tr>
    <tr>
      <td>Nombre País</td>
      <td><input type="text" name="txtPais" id="txtPais" value="${pais.pais}" /></td>
    </tr>
  </tbody>
  </table>
  <br/>
<div class="buttons">
  <ul>
    <li><input type="submit" value="Guardar" name="guardar"/></li>
    <li><a href="#" onclick="javascript: return window.history.back()">Regresar</a></li>
  </ul>
</div>
</form>
 <script>
  function validar(){
         var pais = document.getElementById('txtPais');
         if (pais.value.length==0){
          pais.focus();
          alert("Digite nombre del país");
        return false;
         }

         return true;
  }
</script>
<%@include file="../_down.jsp"%>
```

Paso 3. Crear clase Tabla.java para manejar los Resultset de datos obtenidos de la BD

La siguiente clase será utilizada para modelar tablas HTML a partir de Resultset de datos mediante una conexión

Clase: Tabla.java; Paquete: Utilerias

Ir a aula virtual

Y descargar los archivos Tabla.java y tabla.css, luego agregar la clase Tabla.java al paquete utilerias y la hoja de estilo al folder llamado css.

Paso 4. Crear página para mostrar los países almacenados en tabla países

Crear la siguiente página dentro del folder **países**, esta se utilizará para mostrar una tabla HTML con los países y con las opciones de buscar, agregar, modificar y eliminar países

Página: paises_consulta.jsp

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@include file="../_top.jsp"%>
<c:if test="${resultado!=null}">
  <c:if test="${resultado==1}">
    <p style="color:darkgreen"><strong>Operación realizada correctamente.</strong></p>
  </c:if>
```

```
  <c:if test="${resultado==0}">
    <p style="color:darkred"><strong>La operación no se realizó.</strong></p>
  </c:if>
</c:if>
<h1>Listado Paises</h1><br>
<div class="busqueda" style="width: 50%; text-align: right">
  <form action="${pageContext.servletContext.contextPath}/Paises" method="get">
    <input type="text" name="txtBusqueda" id="txtBusqueda"
        value="${valor}" />
    <input type="submit" value="Buscar"/>
  </form>
</div>
<div class="buttons" >
  <ul>
    <li><a
href="${pageContext.servletContext.contextPath}/Paises?accion=insertar">Nuevo</a></li>
  </ul>
</div>
<br/>
${tabla}
<script>
  window.onload = function() {
    document.getElementById("txtBusqueda").focus();
  };
</script>

<%@include file="../_down.jsp"%>
```

Paso 5. Crear Servlet para realizar las operaciones de Consultar, Insertar, Actualizar y Eliminar.

Corregir un problema del menú

En com.aerolinea.control.Login.java añadir las siguientes líneas

```
....
List<Menu> permisos = getPermisos(u.getIdrol());
                    List<Menu> MenuPrincipal =
permisos.stream().filter(field ->
field.getIdpadre()==0).collect(Collectors.toList());

sesion.setAttribute("MenuPrincipal",MenuPrincipal);
  sesion.setAttribute("Permisos",permisos);
```

```
… .
```

En _top.jsp añadir la siguiente línea

```
…
<% HttpSession sesion = request.getSession();
List<Menu> MenuPrincipal =
(List<Menu>)sesion.getAttribute("MenuPrincipal");%>
…
```

Comentar en com.aerolinea.control.Principal las siguientes líneas

```
//          List<Menu> MenuPrincipal = per.stream().filter(field ->
field.getIdpadre()==0).collect(Collectors.toList());
//          request.setAttribute("MenuPrincipal", MenuPrincipal);
```

En la URL del menú países cambiar de /Principal a /Paises

Crear el siguiente **servlet** llamado **Paises** en el paquete control.

Página: Paises.java

```
package com.aerolinea.control;

import com.aerolinea.entidad.Pais;
import com.aerolinea.conexion.Conexion;
import com.aerolinea.conexion.ConexionPool;
import com.aerolinea.entidad.Menu;
import com.aerolinea.operaciones.Operaciones;
import com.aerolinea.utilerias.Tabla;
import java.io.IOException;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.stream.Collectors;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

```
@WebServlet(name = "Paises", urlPatterns = {"/Paises"})
public class Paises extends HttpServlet {

   @Override
   protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

//       HttpSession s = request.getSession();
//       List<Menu> per = (List<Menu>)s.getAttribute("Permisos");
//       List<Menu> MenuPrincipal = per.stream().filter(field ->
field.getIdpadre()==0).collect(Collectors.toList());
//       request.setAttribute("MenuPrincipal", MenuPrincipal);
//       String op = request.getParameter("op");
//       if (op!=null){
//          List<Menu> PermisosAsignados = per.stream().filter(field ->
field.getIdpadre()==Integer.parseInt(op)).collect(Collectors.toList());
//          request.setAttribute("PermisosAsignados", PermisosAsignados);
//       }
      String accion = request.getParameter("accion");
      if(accion==null) {
         if(request.getSession().getAttribute("resultado")!=null) {
            request.setAttribute("resultado", request.getSession().getAttribute("resultado"));
            request.getSession().removeAttribute("resultado");
         }
         try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();

            String sql = "";
            if(request.getParameter("txtBusqueda")!=null) {
               sql = "select * from pais where pais like ?";
            } else {
               sql = "select * from pais";
            }
            String[][] paises = null;
            if(request.getParameter("txtBusqueda")!=null) {
               List<Object> params = new ArrayList<>();
               params.add("%"+request.getParameter("txtBusqueda").toString()+"%");
               paises = Operaciones.consultar(sql, params);
```

```
} else {
   paises = Operaciones.consultar(sql, null);
}

//declaracion de cabeceras a usar en la tabla HTML
String[] cabeceras = new String[]{
  "ID País",
  "Nombre País"
};
//variable de tipo Tabla para generar la Tabla HTML
Tabla tab = new Tabla(paises, //array que contiene los datos
"50%", //ancho de la tabla px | %
Tabla.STYLE.TABLE01, //estilo de la tabla
Tabla.ALIGN.LEFT,  // alineacion de la tabla
cabeceras); //array con las cabeceras de la tabla

//boton eliminar
tab.setEliminable(true);
//boton actualizar
tab.setModificable(true);
//url del proyecto
tab.setPageContext(request.getContextPath());
//pagina encargada de eliminar
tab.setPaginaEliminable("/Paises?accion=eliminar");
//pagina encargada de actualizacion
tab.setPaginaModificable("/Paises?accion=modificar");
//pagina encargada de seleccion para operaciones
tab.setPaginaSeleccionable("/Paises?accion=modificar");
//icono para modificar y eliminar
tab.setIconoModificable("/iconos/edit.png");
tab.setIconoEliminable("/iconos/delete.png");
//columnas seleccionables
tab.setColumnasSeleccionables(new int[]{1});
//pie de tabla
tab.setPie("Resultado paises");

//imprime la tabla en pantalla
String tabla01 = tab.getTabla();
request.setAttribute("tabla", tabla01);
request.setAttribute("valor", request.getParameter("txtBusqueda"));
request.getRequestDispatcher("paises/paises_consulta.jsp").forward(request, response);
```

```
        } catch(Exception ex) {
            try {
                Operaciones.rollback();
            } catch (SQLException ex1) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
        } finally {
            try {
                Operaciones.cerrarConexion();
            } catch (SQLException ex) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        //request.getRequestDispatcher("paises/paises_consulta.jsp").forward(request, response);
    } else if(accion.equals("insertar")) {
        request.getRequestDispatcher("paises/insertar_modificar.jsp").forward(request, response);
    }  else if(accion.equals("modificar")) {
        try {
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();
            Pais p = Operaciones.get(Integer.parseInt(request.getParameter("id")), new Pais());
            request.setAttribute("pais", p);
            Operaciones.commit();
        } catch(Exception ex) {
            try {
                Operaciones.rollback();
            } catch (SQLException ex1) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
        } finally {
            try {
                Operaciones.cerrarConexion();
            } catch (SQLException ex) {
                Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        request.getRequestDispatcher("paises/insertar_modificar.jsp").forward(request, response);
    } else if(accion.equals("eliminar")) {
        try {
```

```
            Conexion conn = new ConexionPool();
            conn.conectar();
            Operaciones.abrirConexion(conn);
            Operaciones.iniciarTransaccion();
            Pais p = Operaciones.eliminar(Integer.parseInt(request.getParameter("id")), new Pais());
            if(p.getIdpais()!=0) {
               request.getSession().setAttribute("resultado", 1);
            } else {
               request.getSession().setAttribute("resultado", 0);
            }
            Operaciones.commit();
         } catch(Exception ex) {
            try {
               Operaciones.rollback();
            } catch (SQLException ex1) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
            request.getSession().setAttribute("resultado", 0);
         } finally {
            try {
               Operaciones.cerrarConexion();
            } catch (SQLException ex) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
         }
         response.sendRedirect(request.getContextPath()+"/Paises");
      }
   }

   @Override
   protected void doPost(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      String accion = request.getParameter("accion");
      switch(accion) {
         case "insertar_modificar": {
            String idPais = request.getParameter("txtIdpais");
            String pais = request.getParameter("txtPais");
            try {
               Conexion conn = new ConexionPool();
               conn.conectar();
               Operaciones.abrirConexion(conn);
```

```
            Operaciones.iniciarTransaccion();
            if(idPais!=null && !idPais.equals("")) {
               Pais p = new Pais(Integer.parseInt(idPais), pais);
               p = Operaciones.actualizar(p.getIdpais(), p);
               if(p.getIdpais()!=0) {
                  request.getSession().setAttribute("resultado", 1);
               } else {
                  request.getSession().setAttribute("resultado", 0);
               }
            } else {
               Pais p = new Pais();
               p.setPais(pais);
               p = Operaciones.insertar(p);
               if(p.getIdpais()!=0) {
                  request.getSession().setAttribute("resultado", 1);
               } else {
                  request.getSession().setAttribute("resultado", 0);
               }
            }
            Operaciones.commit();
         } catch(Exception ex) {
            try {
               Operaciones.rollback();
            } catch (SQLException ex1) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex1);
            }
            request.getSession().setAttribute("resultado", 2);
            ex.printStackTrace();
         } finally {
            try {
               Operaciones.cerrarConexion();
            } catch (SQLException ex) {
               Logger.getLogger(Paises.class.getName()).log(Level.SEVERE, null, ex);
            }
         }
         response.sendRedirect(request.getContextPath()+"/Paises");
         break;
      }
      case "eliminar": {
         break;
      }
```

```
    }
  }

}
```

III- Tarea

- ✓ Crear el CRUD para la tabla usuario.
- ✓ Crear el resto de mantenimientos para las tablas maestras: aeropuertos, aviones y roles