Servicios Telemáticos Avanzados

8.- Capa de Datos en Java EE

OpenCourseWare 2014

Maider Huarte y Gorka Prieto
Escuela Técnica Superior de Ingeniería de Bilbao
Departamento de Ingeniería de Comunicaciones
Universidad del País Vasco (UPV/EHU)

Servicios Telemáticos Avanzados:
8.- JPA.odp

Capa de Datos en Java EE

ÍNDICE

1.- Entidades
- 1.1.- Introducción
- 1.2.- Anotaciones

2.- Gestión
- 2.1.- Introducción
- 2.2.- Gestor

3.- Consultas
- 3.1.- Introducción
- 3.2.- JPQL

4.- Configuración
- 4.1.- Ficheros

5.- Ejemplos

1.- Entidades

1.1.- Introducción

- Conceptos
  - JPA
  - Entidades
  - Relación con Bases de Datos
  - Estado de persistencia
- Requisitos
  - Anotación @Entity
  - Constructor por defecto
  - No final
  - Implementar Serializable
  - Usar getters/setters

1.- Entidades

1.2.- Anotaciones

- Obligatorias
  - @Entity
  - @Id
- Propiedades
  - @NotNull
  - @Pattern
- Consultas
  - @NamedQuery
  - @NamedQueries
- Relaciones
  - @OneToOne, @OneToMany, @ManyToOne, @ManyToMany

2.- Gestión

2.1.- Introducción

- Contexto de Persistencia (persistence context)
  - Entidades persistidas
  - Sincronización transparente
  - Gestor (EntityManager)
- EntityManager
  - Gestionado por la aplicación
  - Gestionado por el contenedor
    - Java Transaction API (JTA)
    - Inyectado con @PersistenceContext

```
@PresistenceContext
EntityManager em;
```

2.- Gestión

2.2.- Gestor

- Operaciones
  - find: SELECT FROM WHERE PK

```
Customer cust = em.find(Customer.class, custID);
```

  - createNamedQuery
  - persist: INSERT/UPDATE

```
LineItem li = new LineItem(order, product, quantity);
em.persist(li);
```

  - remove: DELETE

```
Order order = em.find(Order.class, orderId);
em.remove(order);
```

  - detach
  - flush
  - clear

3.- Consultas

3.1. Introducción

- Opciones
  - Java Persistence Query Language (JPQL)
    - Similar a SQL
    - Necesario casting
    - Fácilmente leíble
  - Criteria API
    - Más eficiente
    - No requiere aprender JPQL
    - No necesario casting
    - Código más extenso -> objetos y operaciones intermedias

3.- Consultas

3.2. JPQL

- Consultas dinámicas

```
public List findWithName(String name) {
   return em
       .createQuery("SELECT c FROM Customer c WHERE c.name LIKE :custName")
       .setParameter("custName", name)
       .setMaxResults(10)
       .getResultList();
}
```

- Consultas estáticas

```
@NamedQuery(
   name="findAllCustomersWithName",
   query="SELECT c FROM Customer c WHERE c.name LIKE :custName"
)
@Entity
public class Entidad {
   ...
}


public class Bean {

   @PersistenceContext
   public EntityManager em;

   ...
   customers = em
       .createNamedQuery("findAllCustomersWithName")
       .setParameter("custName", "Smith")
       .getResultList();
}
```

4.- Configuración

4.1. Ficheros

- Ficheros
  - JavaEE: persistence.xml
    - Nombre de conexión a BD (JNDI)
    - Clases de entidades de persistencia (opcional)
    - Tipo de BD y gestión de tablas al arranque
  - JBoss: standalone.xml
    - Datos de la conexión
    - Drivers para conexiones

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

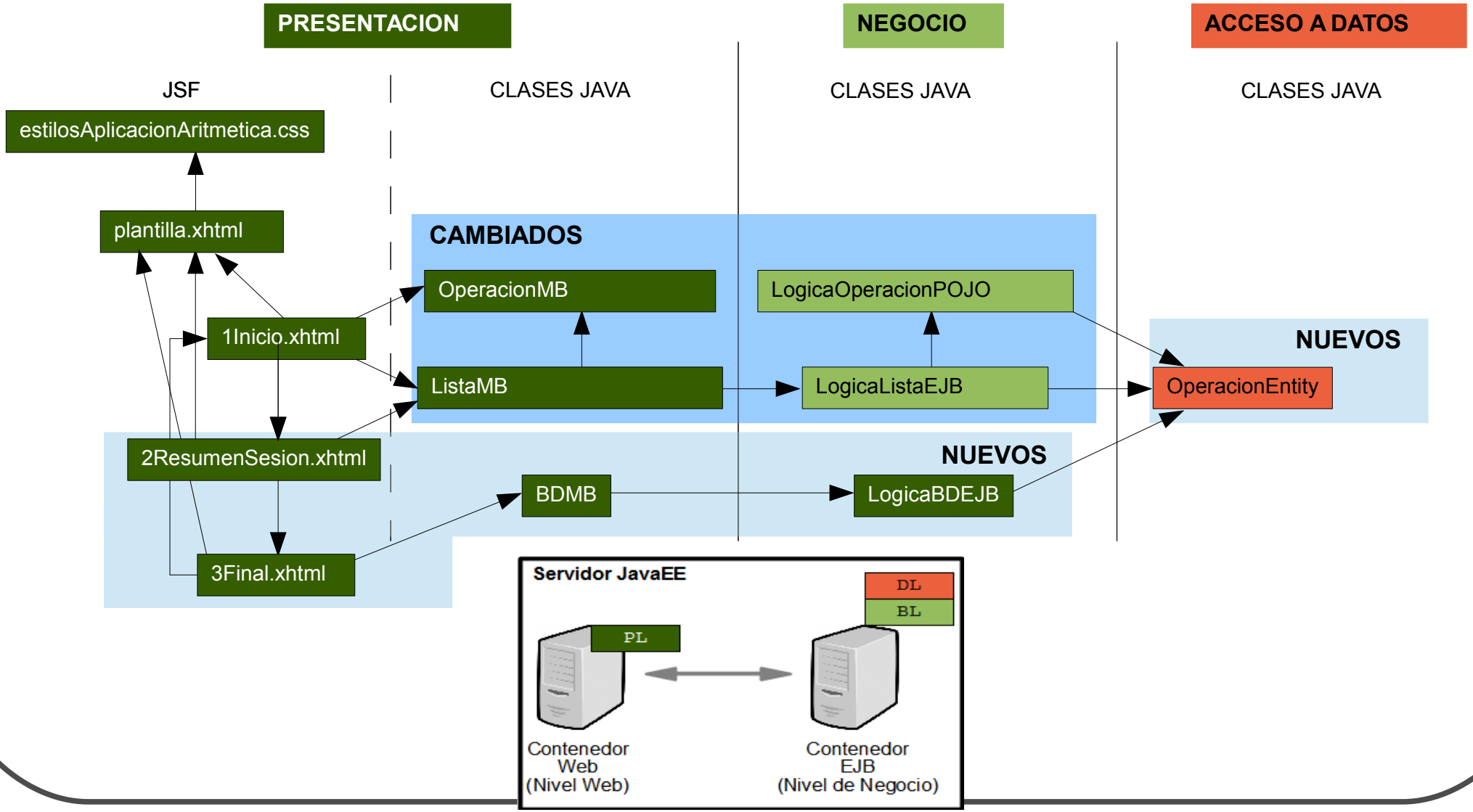
persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0" xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
  <persistence-unit name="T6-6AD_JSF-EJB-JPA_1" transaction-type="JTA">
     <jta-data-source>java:jboss/datasources/ZTA_T6_AD6</jta-data-source>
     <properties>
          <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect" />
          <property name="hibernate.hbm2ddl.auto" value="update" />
     </properties>
  </persistence-unit>
</persistence>
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities



5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

Entity: dl.OperacionEntity.java

```
package dl;

import java.io.Serializable;
import javax.persistence.*;

@Entity
@Table(name="Operacion")
@NamedQueries
({
  @NamedQuery(name="OperacionEntity.findAll", query="SELECT o FROM OperacionEntity o"),
  @NamedQuery(name="OperacionEntity.findAllSesion",query="SELECT e FROM OperacionEntity e WHERE e.sesion = :sesion
ORDER BY e.orden")
})
public class OperacionEntity implements Serializable {
  private static final long serialVersionUID = 1L;
  @Id
  @GeneratedValue(strategy=GenerationType.IDENTITY)
  private int idOperacion;
  @Column(name="Op")
  private char op;
  @Column(name="Op1")
  private float op1;
  //RESTO DE ATRIBUTOS CORRESPONDIENTES A LAS DEMÁS COLUMNAS DE LA TABLA

  public OperacionEntity() {
  }

  public int getIdOperacion() {
      return this.idOperacion;
  }

  public void setIdOperacion(int idOperacion) {
      this.idOperacion = idOperacion;
  }

  //MÉTODOS getters y setters DEL RESTO DE ATRIBUTOS
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

POJO: bl.LogicaOperacionPOJO.java

```
package bl;

import dl.OperacionEntity;

public class LogicaOperacionPOJO {

  public static void calcularOperacion(OperacionEntity o) {
     //RESTO DE CÓDIGO IGUAL QUE EN LA VERSIÓN ANTERIOR
  }
}
```

POJO: bl.LogicaListaEJB.java

```
package bl;

//imports NECESARIOS

@Stateful
@LocalBean
public class LogicaListaEJB {

  private List<OperacionEntity> lista;

  public LogicaListaEJB() {
     lista=new ArrayList<OperacionEntity>();
  }

  public List<OperacionEntity> getlista() {
     return lista;
  }

  public void addOperacion(OperacionEntity e) {
     //MISMO CÓDIGO QUE EN L AVERSIÓN ANTERIOR
  }
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

EJB: bl.LogicaBDEJB.java

```
package bl;

//imports NECESARIOS

import dl.OperacionEntity;

@Stateless
@LocalBean
public class LogicaBDEJB {

  @PersistenceContext
  private EntityManager em;

  public void addListaDB(List<OperacionEntity> lista) {
     for (int i = 0; i < lista.size(); i++) {
          OperacionEntity eDB = lista.get(i);
          eDB.setOrden(i + 1);
          em.persist(eDB);
     }
  }

  public List<OperacionEntity> getListaDB( String nombreSesion ) {
     @SuppressWarnings("unchecked")
     List<OperacionEntity> listaDB = (List<OperacionEntity>) em
          .createNamedQuery("OperacionEntity.findAllSesion")
          .setParameter("sesion", nombreSesion).getResultList();

     return listaDB;
  }
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

ManagedBean: pl.OperacionMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@RequestScoped
public class OperacionMB extends OperacionEntity {
  private static final long serialVersionUID = 1L;

  //RESTO DE CÓDIGO COMO EN LA VERSIÓN ANTERIOR
}
```

ManagedBean: pl.ListaMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@SessionScoped
public class ListaMB {

  @EJB
  private LogicaListaEJB lista;

  public ListaMB() {
     lista = new LogicaListaEJB();
  }

  public List<OperacionEntity> getLista() {
     return lista.getLista();
  }

  // Continuación en la siguiente transparencia
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

ManagedBean: pl.ListaMB.java

```
// Continuación de la anterior transparencia

public void addOperacion(OperacionMB e) {
    OperacionEntity eDB = new OperacionEntity();
    eDB.setSesion(getNombreSesion());
    eDB.setOp1(e.getOp1());
    eDB.setOp(e.getOp());
    eDB.setOp2(e.getOp2());
    eDB.setRes(e.getRes());
    lista.addOperacion(eDB);
    e.setTerminada();
}

public void endSesion() {
    if (!lista.getLista().isEmpty())
          lista.getLista().clear();

    HttpSession sesioa = (HttpSession) FacesContext.getCurrentInstance()
                .getExternalContext().getSession(true);
    sesioa.invalidate();
}

public String getNombreSesion() {
    String user = ((HttpServletRequest) FacesContext.getCurrentInstance()
                .getExternalContext().getRequest()).getRemoteAddr();
    HttpSession sesioa = (HttpSession) FacesContext.getCurrentInstance()
                .getExternalContext().getSession(true);
    String nombreSesion = "IP-CLIENTE:" + user + " TIME:"
                + new Date(sesioa.getCreationTime());

    return nombreSesion;
}
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

Managed Bean: pl.BDMB.java

```
package pl;

// imports

@ManagedBean
@RequestScoped
public class BDMB {

   @EJB
   private LogicaBDEJB db;

   public BDMB() {
       db = new LogicaBDEJB();
   }

   public List<OperacionEntity> getListaDB(String nombreSesion) {
       return db.getListaDB(nombreSesion);
   }

   public void addListaDB(List<OperacionEntity> lista) {
       db.addListaDB(lista);
       return;
   }
}
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

JSF: 1Inicio.xhtml

```
<!DOCTYPE html ...>
<!-- CÓDIGO IGUAL QUE EN LA VERSIÓN ANTERIOR -->
      <h:commandButton action="2ResumenSesion" id="submit" value="VER OPERACIONES" />
  </h:form>
      </p>
</ui:define>
</ui:composition>
</html>
```

JSF: 2ResumenSesion.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">2ResumenSesion</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: RESUMEN DE SESION</ui:define>
<ui:define name="Contenido">
  <h1>RESUMEN DE SESION</h1>
  NÂ° de operaciones: #{listaMB.lista.size()}
  <h:dataTable value="#{listaMB.lista}" var="operacion" border="2">
     <h:column>
          <f:facet name="header">OPERACIONES</f:facet>
          #{operacion.op1}#{operacion.op}#{operacion.op2}
     </h:column>
     <h:column>
          <f:facet name="header">RESULTADOS</f:facet>
          #{operacion.res}
     </h:column>
  </h:dataTable>
  <p>
  <h:form>
     <h:commandButton action="3Final" actionListener="#{bDMB.addListaDB(listaMB.lista)}" id="submit" value="GRABAR
SESION EN BD" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```

5.- Ejemplos

5.1.- Ejemplo1

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

JSF: 3Final.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">3Amaiera: Aplikazio aritmetikoaren amaierako orrialdea</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: FINAL</ui:define>
<ui:define name="Contenido">
  <h1>SESION GRABADA EN LA BD</h1>
  Sesion: #{listaMB.nombreSesion}
  <h:dataTable value="#{bDMB.getListaDB(listaMB.nombreSesion)}" var="operacion" border="2">
     <h:column>
          <f:facet name="header">OPERACIONES</f:facet>
          #{operacion.op1}#{operacion.op}#{operacion.op2}
     </h:column>
     <h:column>
          <f:facet name="header">RESULTADOS</f:facet>
          #{operacion.res}
     </h:column>
  </h:dataTable>
  <p>
  <h:form>
     <h:commandButton action="1Inicio" actionListener="#{listaMB.endSesion()}" id="submit" value="EMPEZAR NUEVA
SESION" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```

5.- Ejemplos

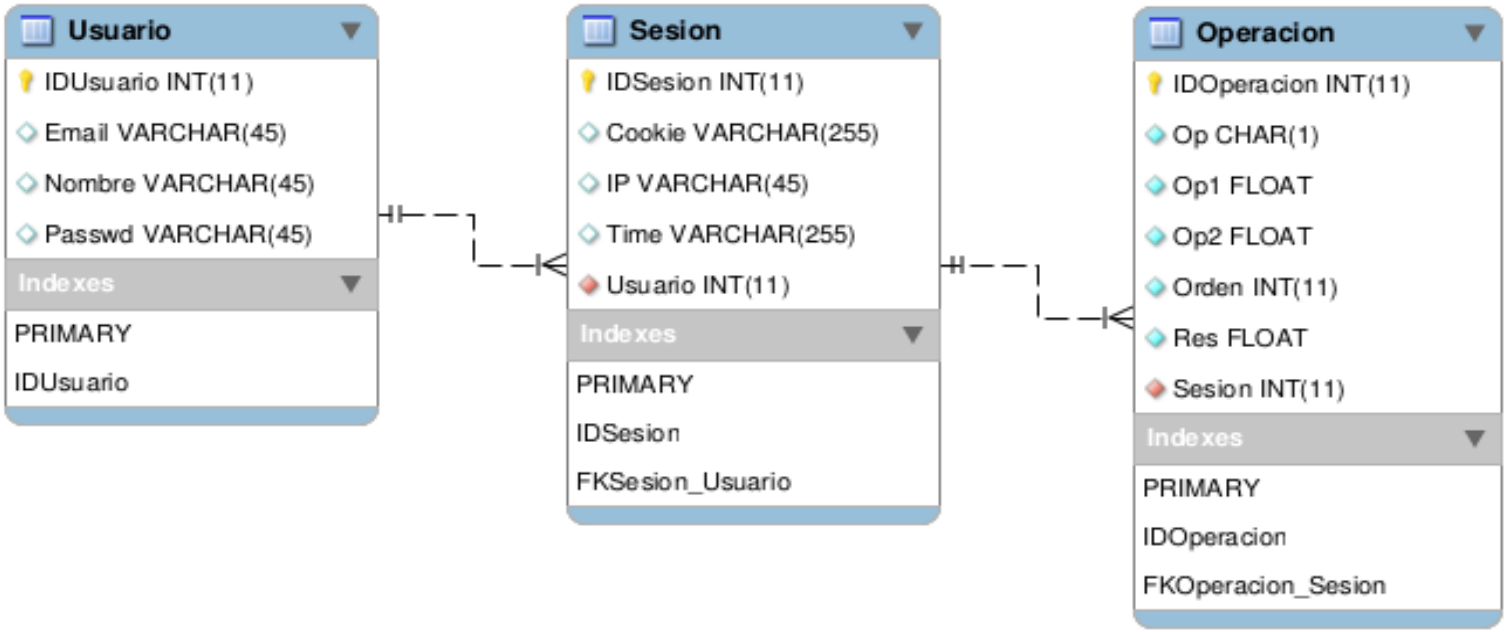
5.2.- Ejemplo2

- Aplicaciones JSFs+EJBs+JPA
  - Bloques a programar
    - Base de Datos
    - Aplicación JavaEE con JSFs
      - Clases Java de cada capa
        - DL
        - BL
        - PL
      - Ficheros .xhtml
      - Ficheros de configuración
        - persistence.xml
        - web.xml
        - ...

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS
  - Planteamiento posible
    - Base de datos: Modelo de datos y relaciones



5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS
  - Planteamiento posible
    - Base de datos: Generación de tablas

```
CREATE  TABLE `STA`.`Usuario` (
  `IDUsuario` INT(11) NOT NULL AUTO_INCREMENT ,
  `Email` VARCHAR(45) NULL DEFAULT NULL ,
  `Nombre` VARCHAR(45) NULL DEFAULT NULL ,
  `Passwd` VARCHAR(45) NULL DEFAULT NULL ,
  PRIMARY KEY (`IDUsuario`) ,
  UNIQUE INDEX `IDUsuario` (`IDUsuario` ASC) )
ENGINE = InnoDB
AUTO_INCREMENT = 0;
```

Sentencias SQL para tabla *Usuario*

```
CREATE  TABLE IF NOT EXISTS `STA`.`Sesion` (
  `IDSesion` INT(11) NOT NULL AUTO_INCREMENT ,
  `Cookie` VARCHAR(255) NULL DEFAULT NULL ,
  `IP` VARCHAR(45) NULL DEFAULT NULL ,
  `Time` VARCHAR(255) NULL DEFAULT NULL ,
  `Usuario` INT(11) NOT NULL ,
  PRIMARY KEY (`IDSesion`) ,
  UNIQUE INDEX `IDSesion` (`IDSesion` ASC) ,
  INDEX `FKSesion_Usuario` (`Usuario` ASC) ,
  CONSTRAINT `FKSesion_Usuario`
    FOREIGN KEY (`Usuario` )
    REFERENCES `STA`.`Usuario` (`IDUsuario` ))
ENGINE = InnoDB
AUTO_INCREMENT = 0;
```

Sentencias SQL para tabla *Sesion*

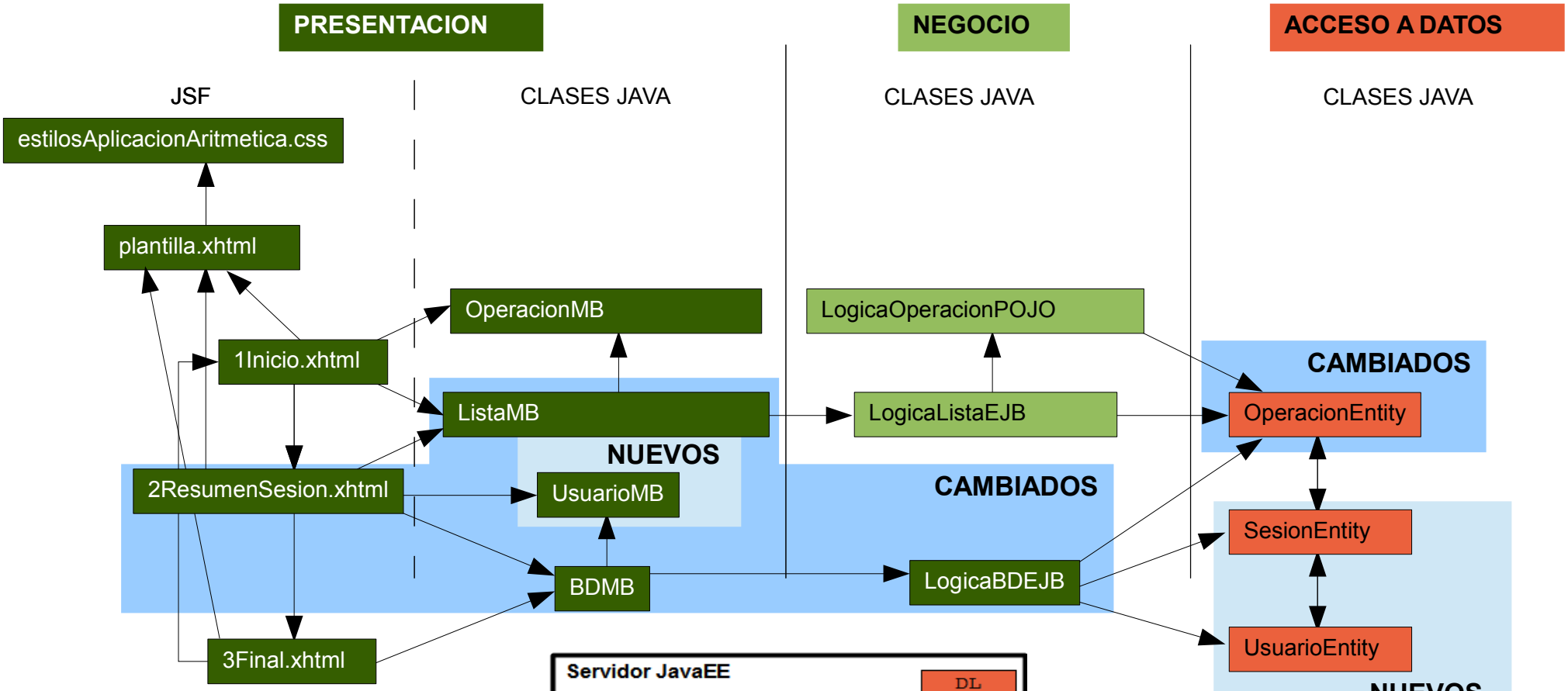
```
CREATE  TABLE IF NOT EXISTS `STA`.`Operacion` (
  `IDOperacion` INT(11) NOT NULL AUTO_INCREMENT ,
  `Op` CHAR(1) NOT NULL ,
  `Op1` FLOAT NOT NULL ,
  `Op2` FLOAT NOT NULL ,
  `Orden` INT(11) NOT NULL ,
  `Res` FLOAT NOT NULL ,
  `Sesion` INT(11) NOT NULL ,
  PRIMARY KEY (`IDOperacion`) ,
  UNIQUE INDEX `IDOperacion` (`IDOperacion` ASC) ,
  INDEX `FKOperacion_Sesion` (`Sesion` ASC) ,
  CONSTRAINT `FKOperacion_Sesion`
    FOREIGN KEY (`Sesion` )
    REFERENCES `STA`.`Sesion` (`IDSesion` ))
ENGINE = InnoDB
AUTO_INCREMENT = 0;
```

Sentencias SQL para tabla *Operacion*

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS



5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

Entity: dl.OperacionEntity.java

```
package dl;

//MISMO CÓDIGO QUE EN LA VERSIÓN ANTERIOR

  //bi-directional many-to-one association to SesionEntity
  @ManyToOne
  @JoinColumn(name="Sesion")
  private SesionEntity sesionBean;

  public OperacionEntity() {
  }

  public OperacionEntity(char op, float op1, float op2, int orden, float res, SesionEntity sesionBean) {
      this.op=op;
      this.op1=op1;
      this.op2=op2;
      this.orden=orden;
      this.res=res;
      this.sesionBean=sesionBean;
  }

  //MISMOS getters y setters QUE EN LA VERSIÓN ANTERIOR

  public SesionEntity getSesionBean() {
      return this.sesionBean;
  }

  public void setSesionBean(SesionEntity sesionBean) {
      this.sesionBean = sesionBean;
  }

}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

Entity: dl.SesionEntity.java

```
package dl;

//imports NECESARIOS

@Entity
@Table(name="Sesion")
@NamedQuery(name="SesionEntity.findAll", query="SELECT s FROM SesionEntity s")
public class SesionEntity implements Serializable {
  private static final long serialVersionUID = 1L;

  @Id
  @GeneratedValue(strategy=GenerationType.IDENTITY)
  private int idSesion;

  @Column(name="Cookie")
  private String cookie;

  @Column(name="IP")
  private String ip;

  @Column(name="Time")
  private String time;

  //bi-directional many-to-one association to OperacionEntity
  @OneToMany(mappedBy="sesionBean",cascade=CascadeType.ALL)
  private List<OperacionEntity> operacions;

  //bi-directional many-to-one association to UsuarioEntity
  @ManyToOne
  @JoinColumn(name="Usuario")
  private UsuarioEntity usuarioBean;

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

Entity: dl.SesionEntity.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  public SesionEntity() {
  }

  public SesionEntity(String cookie, String ip, String time, UsuarioEntity usuarioBean) {
      this.cookie=cookie;
      this.ip=ip;
      this.time=time;
      this.usuarioBean=usuarioBean;
  }

  public int getIdSesion() {
      return this.idSesion;
  }

  public void setIdSesion(int idSesion) {
      this.idSesion = idSesion;
  }

  //RESTO DE MÉTODOS getters y setters DE ATRIBUTOS NO FK

  public List<OperacionEntity> getOperacions() {
      return this.operacions;
  }

  public void setOperacions(List<OperacionEntity> operacions) {
      this.operacions = operacions;
  }

  public UsuarioEntity getUsuarioBean() {
      return this.usuarioBean;
  }

  public void setUsuarioBean(UsuarioEntity usuarioBean) {
      this.usuarioBean = usuarioBean;
  }
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

Entity: dl.UsuarioEntity.java

```
package dl;

//imports NECESARIOS

@Entity
@Table(name="Usuario")
@NamedQueries
({
  @NamedQuery(name="UsuarioEntity.findAll", query="SELECT u FROM UsuarioEntity u"),
  @NamedQuery(name="UsuarioEntity.findAllNombre",query="SELECT u FROM UsuarioEntity u WHERE u.nombre = :nombre")
})
public class UsuarioEntity implements Serializable {
  private static final long serialVersionUID = 1L;

  @Id
  @GeneratedValue(strategy=GenerationType.IDENTITY)
  private int idUsuario;

  @Column(name="Email")
  private String email;

  @Column(name="Nombre")
  private String nombre;

  @Column(name="Password")
  private String password;

  //bi-directional many-to-one association to SesionEntity
  @OneToMany(mappedBy="usuarioBean")
  private List<SesionEntity> sesionEntities;

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

Entity: dl.UsuarioEntity.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  public UsuarioEntity() {
  }

  public UsuarioEntity(String email, String nombre, String password) {
     this.email=email;
     this.nombre=nombre;
     this.password=password;
  }

  public int getIdUsuario() {
     return this.idUsuario;
  }

  public void setIdUsuario(int idUsuario) {
     this.idUsuario = idUsuario;
  }

  //RESTO DE MÉTODOS getters y setters DE ATRIBUTOS NO FK

  public List<SesionEntity> getSesions() {
     return this.sesionEntities;
  }

  public void setSesions(List<SesionEntity> sesionEntities) {
     this.sesionEntities = sesionEntities;
  }
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

EJB: bl.LogicaBDEJB.java

```
package bl;

//imports NECESARIOS

@Stateful
@LocalBean
public class LogicaBDEJB {

  @PersistenceContext
  private EntityManager em;
  private UsuarioEntity usuarioDB;
  private SesionEntity sesionDB;

  public void addListaDB(List<OperacionEntity> lista, UsuarioEntity usuario, SesionEntity sesion ) {
     addUsuarioDB(usuario);

     if (usuarioDB != null) {
          addSesionDB(sesion);

          OperacionEntity e;
          for (int i = 0; i < lista.size(); i++) {
               e = lista.get(i);
               e.setOrden(i+1);
               e.setSesionBean(sesionDB);
               em.persist(e);
          }
     }
  }

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE

}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

EJB: bl.LogicaBDEJB.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  private void addUsuarioDB(UsuarioEntity e) {
      try {
           usuarioDB = (UsuarioEntity) em
                      .createNamedQuery("UsuarioEntity.findAllNombre")
                      .setParameter("nombre", e.getNombre()).getSingleResult();

           if (!((usuarioDB.getPassword()).equals(e.getPassword())))
                usuarioDB = null;
      } catch (Exception ex) {
           usuarioDB = e;
           em.persist(usuarioDB);
      }
  }

  private void addSesionDB( SesionEntity sesionEntity ) {
      sesionDB = sesionEntity;
      sesionDB.setUsuarioBean(usuarioDB);
      em.persist(sesionDB);
  }

  public UsuarioEntity getUsuarioDB() {
      return usuarioDB;
  }

  public SesionEntity getSesionDB() {
      return sesionDB;
  }

  public List<OperacionEntity> getListaDB() {
      @SuppressWarnings("unchecked")
      List<OperacionEntity> listaDB = (List<OperacionEntity>) em
                .createNamedQuery("OperacionEntity.findAllSesion")
                .setParameter("sesion", sesionDB).getResultList();
      return listaDB;
  }
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

ManagedBean: pl.BDMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@RequestScoped
public class BDMB {
  @EJB
  private LogicaBDEJB db;

  public BDMB() {
     db = new LogicaBDEJB();
  }

  public void addListaDB(List<OperacionEntity> lista, UsuarioMB usuario,
          SesionEntity sesion) {
     UsuarioEntity ue = new UsuarioEntity(usuario.getEmail(),
               usuario.getNombre(), usuario.getPassword());
     db.addListaDB(lista, ue, sesion);
  }

  public List<OperacionEntity> getListaDB() {
     return db.getListaDB();
  }

  public UsuarioEntity getUsuarioDB() {
     return db.getUsuarioDB();
  }

  public SesionEntity getSesionDB() {
     return db.getSesionDB();
  }
}
```

ManagedBean: pl.UsuarioMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@RequestScoped
public class UsuarioMB extends UsuarioEntity {
  private static final long serialVersionUID = 1L;
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 1 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities

ManagedBean: pl.ListaMB.java

```
@ManagedBean
@SessionScoped
public class ListaMB {

  @EJB
  private LogicaListaEJB lista;

  public ListaMB() {
     lista = new LogicaListaEJB();
  }

  public List<OperacionEntity> getlista() {
     return lista.getlista();
  }

  public void addOperacion(OperacionMB operacionMB) {
     OperacionEntity operacionEntity = new OperacionEntity(
               operacionMB.getOp(), operacionMB.getOp1(),
               operacionMB.getOp2(), 0, 0, null);
     lista.addOperacion(operacionEntity);
     operacionMB.setTerminada();
  }

  // endSesion() como antes

  public SesionEntity getSesion() {
     HttpSession sesion = (HttpSession) FacesContext.getCurrentInstance()
               .getExternalContext().getSession(true);
     String cookie = sesion.getId();
     String ip = ((HttpServletRequest) FacesContext.getCurrentInstance()
               .getExternalContext().getRequest()).getRemoteAddr();
     String time = "" + new Date(sesion.getCreationTime());
     SesionEntity se = new SesionEntity(cookie, ip, time, null);

     return se;
  }
}
```

5.- Ejemplos

5.2.- Ejemplo2

- EJEMPLO 2 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS

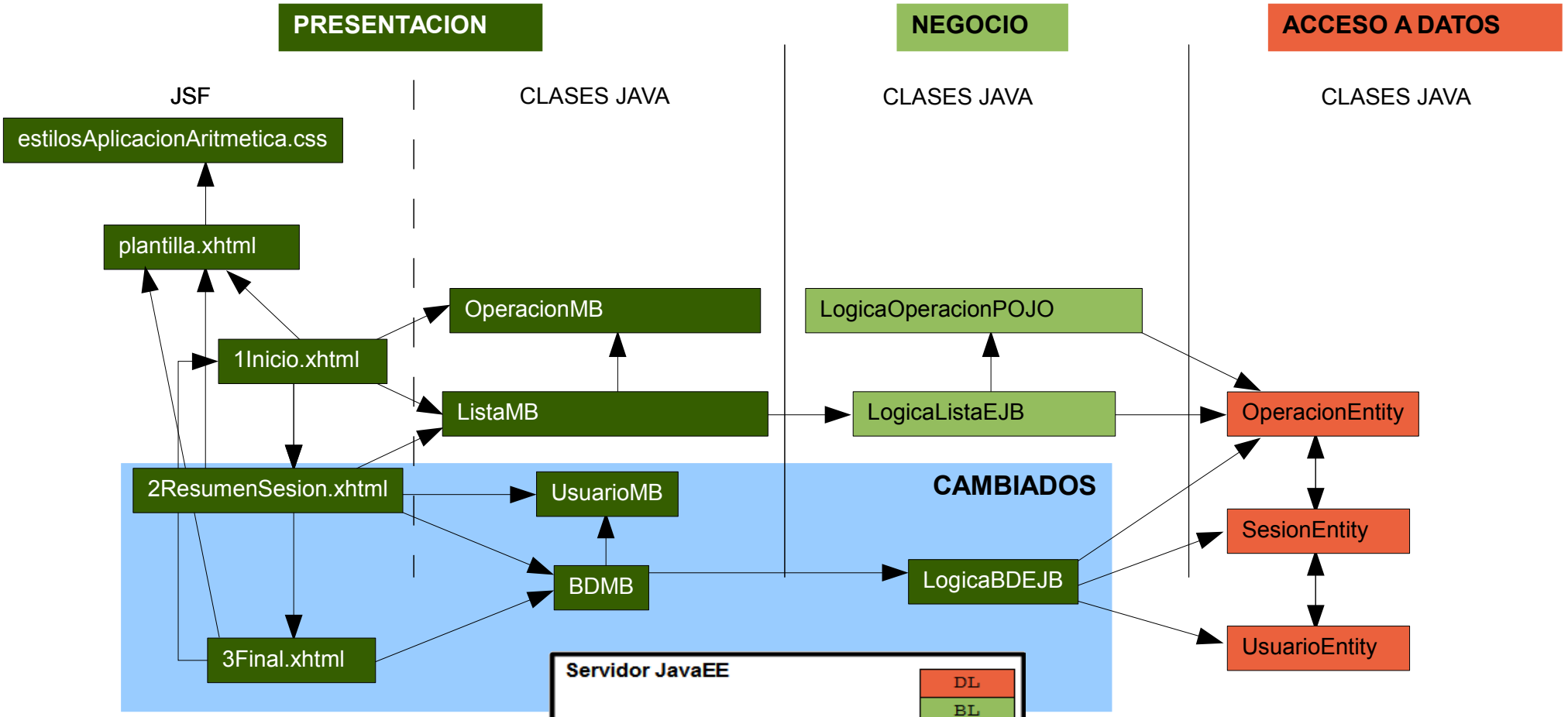
JSF: 2ResumenSesion.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">2ResumenSesio</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: RESUMEN DE SESION</ui:define>
<ui:define name="Contenido">
  <h1>RESUMEN DE SESION</h1>
  N° de operaciones: #{listaMB.lista.size()}
  <h:dataTable value="#{listaMB.lista}" var="operacion" border="2">
     <h:column>
          <f:facet name="header">OPERACIONES</f:facet>
          #{operacion.op1}#{operacion.op}#{operacion.op2}
     </h:column>
     <h:column>
          <f:facet name="header">RESULTADOS</f:facet>
          #{operacion.res}
     </h:column>
  </h:dataTable>
  <p>
  <h:form>
     <h1>INTRODUZCA LOS DATOS DE USUARIO</h1>
     <label>Nombre: </label><h:inputText id="nombre" value="#{usuarioMB.nombre}" required="true"
requiredMessage="ERROR: EL NOMBRE ES OBLIGATORIO" /><h:message for="nombre" style="color:blue" /><br />
     <label>Password: </label><h:inputText id="password" value="#{usuarioMB.password}" required="true"
requiredMessage="ERROR: EL PASSWORD ES OBLIGATORIO" /><h:message for="password" style="color:blue" /><br />
     <label>E-mail: </label><h:inputText id="email" value="#{usuarioMB.email}" /><br />
     <h:commandButton action="3Final" actionListener="#{bDMB.addListaDB(listaMB.lista,usuarioMB,listaMB.sesion)}"
id="submitAdd" value="GRABAR DATOS EN BD" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```

5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD



5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD

Entity: dl.UsuarioEntity.java

```
@Entity
@Table(name="Usuario")
@NamedQueries
({
  @NamedQuery(name="UsuarioEntity.findAll", query="SELECT u FROM UsuarioEntity u"),
  @NamedQuery(name="UsuarioEntity.findAllNombre",query="SELECT u FROM UsuarioEntity u WHERE u.nombre = :nombre")
})
public class UsuarioEntity implements Serializable {
  private static final long serialVersionUID = 1L;

  @Id
  @GeneratedValue(strategy=GenerationType.IDENTITY)
  private int idUsuario;

  @Column(name="Email")
  private String email;

  @Column(name="Nombre")
  private String nombre;

  @Column(name="Password")
  private String password;

  //bi-directional many-to-one association to SesionEntity
  @OneToMany(mappedBy="usuarioBean",cascade=CascadeType.ALL)
  private List<SesionEntity> sesionEntities;

// RESTO IGUAL QUE ANTES
```

5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD

EJB: bl.LogicaBDEJB.java

```
//MISMO CÓDIGO QUE EN LA VERSIÓN ANTERIOR+removeUsuarioDB

  public void removeUsuarioDB(UsuarioEntity e) {
     UsuarioEntity usuarioBean = null;
     try {
          usuarioBean = (UsuarioEntity) em
                     .createNamedQuery("UsuarioEntity.findAllNombre")
                     .setParameter("nombre", e.getNombre()).getSingleResult();
          if ((usuarioBean.getPassword()).equals(e.getPassword()))
               em.remove(usuarioBean);
     } catch (Exception ex) {
     }
  }
}
```

5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD

ManagedBean: pl.BDMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@RequestScoped
public class BDMB {
  @EJB
  private LogicaBDEJB db;
  private boolean addDB;
  private boolean removeDB;

  //Constructor IGUAL QUE EN LA VERSIÓN ANTERIOR

  public void addListaDB(List<OperacionEntity> lista, UsuarioMB usuario,
          SesionEntity sesion) {
     UsuarioEntity ue = new UsuarioEntity(usuario.getEmail(),
               usuario.getNombre(), usuario.getPassword());
     db.addListaDB(lista, ue, sesion);
     addDB = true;
  }

  public boolean getAddDB() {
     return addDB;
  }

  //getListaDB(), getUsuarioDB() y getSesionDB() IGUAL QUE EN LA VERSIÓN ANTERIOR

  public void removeUsuarioDB(UsuarioMB usuarioMB) {
     db.removeUsuarioDB(usuarioMB);
     removeDB = true;
  }

  public boolean getRemoveDB() {
     return removeDB;
  }
}
```

5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD

JSF:2ResumenSesion.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">2ResumenSesion</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: RESUMEN DE SESION</ui:define>
<ui:define name="Contenido">
  <h1>RESUMEN DE SESION</h1>
  N° de operaciones: #{listaMB.lista.size()}
  <h:dataTable value="#{listaMB.lista}" var="operacion" border="2">
     <h:column>
          <f:facet name="header">OPERACIONES</f:facet>
          #{operacion.op1}#{operacion.op}#{operacion.op2}
     </h:column>
     <h:column>
          <f:facet name="header">RESULTADOS</f:facet>
          #{operacion.res}
     </h:column>
  </h:dataTable>
  <p>
  <h:form>
     <h1>INTRODUZCA LOS DATOS DE USUARIO</h1>
     <label>Nombre: </label><h:inputText id="nombre" value="#{usuarioMB.nombre}" required="true"
requiredMessage="ERROR: EL NOMBRE ES OBLIGATORIO" /><h:message for="nombre" style="color:blue" /><br />
     <label>Password: </label><h:inputText id="password" value="#{usuarioMB.password}" required="true"
requiredMessage="ERROR: EL PASSWORD ES OBLIGATORIO" /><h:message for="password" style="color:blue" /><br />
     <label>E-mail: </label><h:inputText id="email" value="#{usuarioMB.email}" /><br />
     <h:commandButton action="3Final" actionListener="#{bDMB.addListaDB(listaMB.lista,usuarioMB,listaMB.sesion)}"
id="submitAdd" value="GRABAR DATOS EN BD" />
     <h:commandButton action="3Final" actionListener="#{bDMB.removeUsuarioDB(usuarioMB)}" id="submitRemove"
value="BORRAR DATOS DE BD" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```

5.- Ejemplos

5.3.- Ejemplo3

- EJEMPLO 3 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD

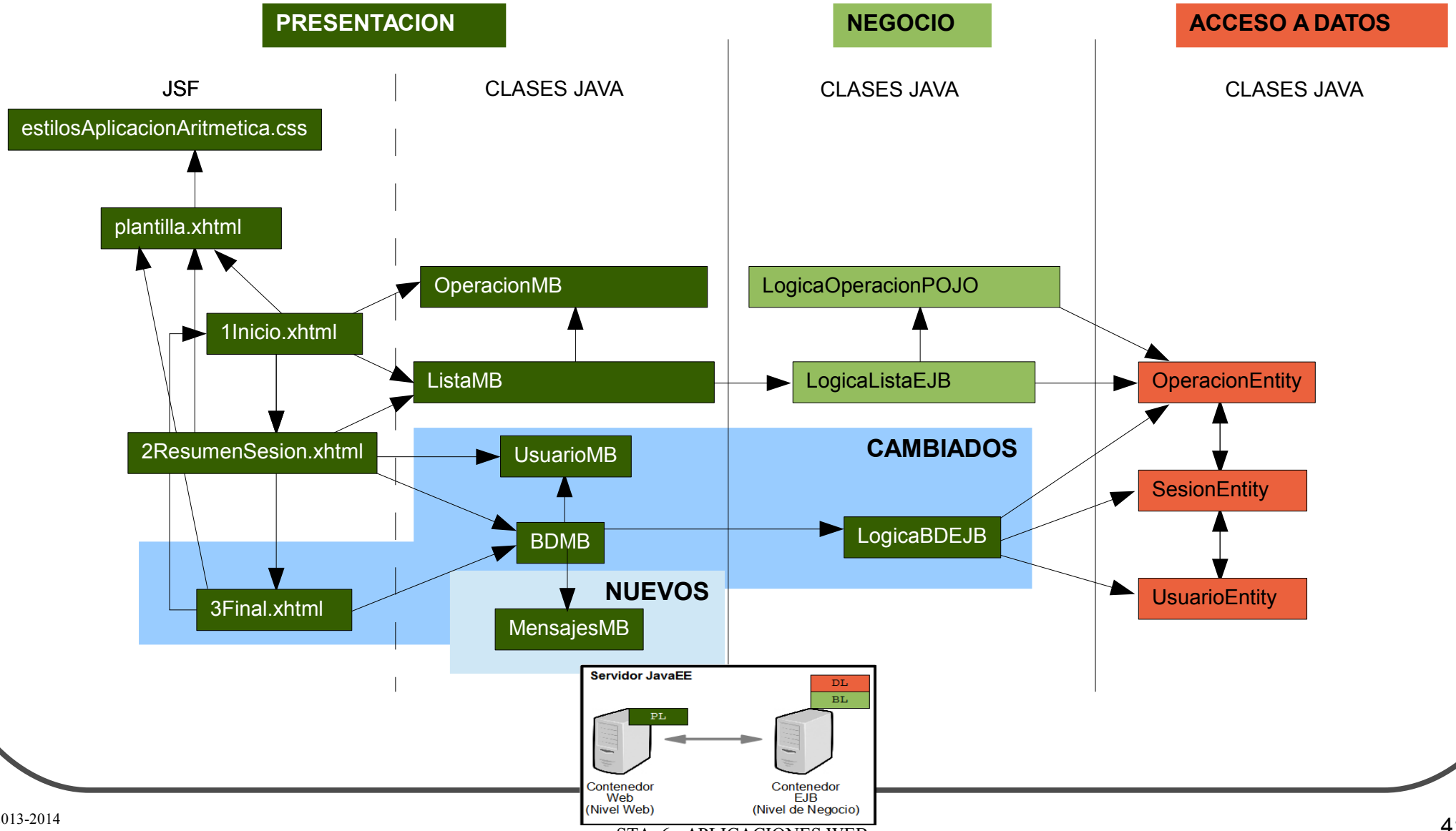
JSF:3Final.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">3Final</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: FINAL</ui:define>
<ui:define name="Contenido">
  <h:panelGroup rendered="#{bDMB.addDB}">
  <h1>SESION GRABADA EN LA BD</h1>
  Sesion: #{bDMB.sesionDB.cookie}
  <h:dataTable value="#{bDMB.listaDB}" var="operacion" border="2">
     <h:column>
          <f:facet name="header">OPERACIONES</f:facet>
          #{operacion.op1}#{operacion.op}#{operacion.op2}
     </h:column>
     <h:column>
          <f:facet name="header">RESULTADOS</f:facet>
          #{operacion.res}
     </h:column>
  </h:dataTable>
  </h:panelGroup>
  <h:panelGroup rendered="#{bDMB.removeDB}">
  <h1>USUARIO QUE SE HA INTENTADO BORRAR DE LA BD</h1>
     Usuario: #{usuarioMB.nombre}<br />
  </h:panelGroup>
  <p>
  <h:form>
     <h:commandButton action="1Inicio" actionListener="#{listaMB.endSesion()}" id="submit" value="EMPEZAR NUEVA
SESION" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores



5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

EJB: bl.LogicaBDEJBean.java

```
package bl;

//imports NECESARIOS

@Stateful
@LocalBean
public class LogicaBDEJB {

  @PersistenceContext
  private EntityManager em;
  private UsuarioEntity usuarioDB;
  private SesionEntity sesionDB;
  private int codigoDB;

  public void addListaDB(List<OperacionEntity> lista, UsuarioEntity usuario,
          SesionEntity sesion) {
     addUsuarioDB(usuario);

     if (usuarioDB != null) {
          addSesionDB(sesion);

          OperacionEntity e;
          for (int i = 0; i < lista.size(); i++) {
               e = lista.get(i);
               e.setOrden(i + 1);
               e.setSesionBean(sesionDB);
               em.persist(e);
          }
          codigoDB = 0;
     }
  }

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE
}
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

EJB: bl.LogicaBDEJBean.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  private void addUsuarioDB(UsuarioEntity e) {
      try {
            usuarioDB = (UsuarioEntity) em
                        .createNamedQuery("UsuarioEntity.findAllNombre")
                        .setParameter("nombre", e.getNombre()).getSingleResult();

            if (!((usuarioDB.getPassword()).equals(e.getPassword()))) {
                  usuarioDB = null;
                  codigoDB = 1;
            }
      } catch (Exception ex) {
            usuarioDB = e;
            em.persist(usuarioDB);
      }
  }

  private void addSesionDB(SesionEntity sesionEntity) {
      sesionDB = sesionEntity;
      sesionDB.setUsuarioBean(usuarioDB);
      em.persist(sesionDB);
  }

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE

}
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

EJB: bl.LogicaBDEJBean.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  public UsuarioEntity getUsuarioDB() {
      try {
            usuarioDB = em.find(UsuarioEntity.class, usuarioDB.getIdUsuario());
      } catch (Exception ex) {
            codigoDB = 2;
      }
      return usuarioDB;
  }

  public SesionEntity getSesionDB() {
      try {
            sesionDB = (SesionEntity) em.find(SesionEntity.class,
                        sesionDB.getIdSesion());
      } catch (Exception ex) {
            codigoDB = 2;
      }
      return sesionDB;
  }

  @SuppressWarnings("unchecked")
  public List<OperacionEntity> getListaDB() {
      List<OperacionEntity> listaDB = null;
      try {
            listaDB = (List<OperacionEntity>) em
                        .createNamedQuery("OperacionEntity.findAllSesion")
                        .setParameter("sesion", sesionDB).getResultList();
      } catch (Exception ex) {
            codigoDB = 2;
      }
      return listaDB;
  }

  //RESTO DE CÓDIGO EN LA PÁGINA SIGUIENTE
}
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

EJB: bl.LogicaBDEJBean.java

```
//CÓDIGO EN LA PÁGINA ANTERIOR

  public void removeUsuarioDB(UsuarioEntity e) {
     UsuarioEntity usuarioBean = null;
     codigoDB = 3;

     try {
          usuarioBean = (UsuarioEntity) em
                    .createNamedQuery("UsuarioEntity.findAllNombre")
                    .setParameter("nombre", e.getNombre()).getSingleResult();
          if ((usuarioBean.getPassword()).equals(e.getPassword()))
               em.remove(usuarioBean);
          else
               codigoDB = 4;
     } catch (Exception ex) {
          codigoDB = 5;
     }
  }

  public int getCodigoDB() {
     return codigoDB;
  }
}
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

POJO: pl.CastellanoPOJO.java

```
package pl;

public abstract class CastellanoPOJO {
  public static String mensajes[]={
      "TODO SE HA GUARDADO BIEN EN LA BD",//0
      "PASSWORD INCORRECTO: no se ha guardado nada",//1
      "ALGO NO SE HA GUARDADO BIEN EN LA BD",//2
      "TODOS LOS DATOS DEL USUARIO SE HA BORRADO BIEN DE LA BD",//3
      "PASSWORD INCORRECTO: no se ha borrado nada",//4
      "EL USUARIO INDICADO NO ESTÃˆ EN LA BD: no se ha borrado nada"//5
  };
}
```

5.- Ejemplos

5.4.- Ejemplo4

- EJEMPLO 4 JSFs+EJBs+JP:Aplicación Aritmética con JSFs, EJBs y JPEntities RELACIONADAS; borrar datos de la BD, controlar errores

ManagedBean: pl.BDMB.java

```
package pl;

//imports NECESARIOS

@ManagedBean
@RequestScoped
public class BDMB {
  @EJB
  private LogicaBDEJB db;
  private boolean addDB;
  private boolean removeDB;
  private int codigoDB;

  public BDMB() {
      db = new LogicaBDEJB();
  }

  //addListaDB(), getListaDB(), getUsuarioDB(), getSesionDB(): IGUAL QUE EN LA VERSIÓN ANTERIOR

  public void removeUsuarioDB(UsuarioMB usuarioMB) {
      db.removeUsuarioDB(usuarioMB);
      codigoDB = db.getCodigoDB();
      removeDB = true;
  }

  public boolean getRemoveDB() {
      return removeDB;
  }

  public int getCodigoDB() {
      return codigoDB;
  }

  public String getMensajeDB() {
      return CastellanoPOJO.mensajes[db.getCodigoDB()];
  }
}
```

5.- Ejemplos

5.4.- Ejemplo4

JSF: 3Final.xhtml

```
<!DOCTYPE html ...>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets">
<ui:composition template="/plantilla.xhtml">
<ui:define name="TituloDocumento">3Final</ui:define>
<ui:define name="TituloPagina">APLICACION ARITMETICA CON JSF, EJB y JPA: FINAL</ui:define>
<ui:define name="Contenido">
  <h1>#{bDMB.mensajeDB}</h1>
  <h:panelGroup rendered="#{bDMB.codigoDB==0}">
     Usuario: #{bDMB.usuarioDB.nombre}<br />
     Sesion: #{bDMB.sesionDB.cookie}<br />
     <h:dataTable value="#{bDMB.listaDB}" var="operacion" border="2">
          <h:column>
               <f:facet name="header">ERAGIKETAK</f:facet>
               #{operacion.op1}#{operacion.op}#{operacion.op2}
          </h:column>
          <h:column>
               <f:facet name="header">RESULTADOS</f:facet>
               #{operacion.res}
          </h:column>
     </h:dataTable>
  </h:panelGroup>
  <h:panelGroup rendered="${bDMB.codigoDB==1}">
     Usuario: #{usuarioMB.nombre}<br />
  </h:panelGroup>
  <h:panelGroup rendered="${bDMB.codigoDB==3 or bDMB.codigoDB==4 or bDMB.codigoDB==5}">
     Usuario: #{usuarioMB.nombre}<br />
  </h:panelGroup>
  <p>
  <h:form>
     <h:commandButton action="1Inicio" actionListener="#{listaMB.endSesion()}" id="submit" value="EMPEZAR NUEVA
SESION" />
  </h:form>
  </p>
</ui:define>
</ui:composition>
</html>
```