AUT04_03 LoginMVC

1. Objetivos

Los objetivos de esta actividad son:

- Comprender el concepto de patrón MVC
- Aprender a usar el objeto RequestDispatcher para incluir o delegar en otro servlet (o JSP) la respuesta al cliente web.
- Aprender cómo se pasan objetos de un web component (JSP o Servlet) a otro a través de la request.

2. Puntos clave

- Paso de objetos a otro servlet, incluyéndolos en el ámbito (scope) de request.

```
    // Incluímos el objeto login en la request, con clave "loginBean"
    request.setAttribute("loginBean", login);
```

- Obtener un dispatcher para incluir (include) o continuar/delegar (forward) en otro web component (servlet/JSP) la generación de la respuesta.

```
// Obtenemos el ServletContext
// Interface que define un conjunto de métodos que el servlet utiliza
// para comunicarse con su contenedor de servlets
//http://docs.oracle.com/javaee/6/api/javax/servlet/ServletContext.html
ServletContext servletCtx = getServletContext();


// Obtenemos el RequestDispatcher para comunicar con la JSP
// loginSuccess.jsp y le pasamos el control (continúa tú)
servletCtx.getRequestDispatcher("/loginSuccess.jsp").
      forward(request, response);
```

- Uso de un bean en la JSP de destino
  - Definimos el bean que recuperamos del scope request.

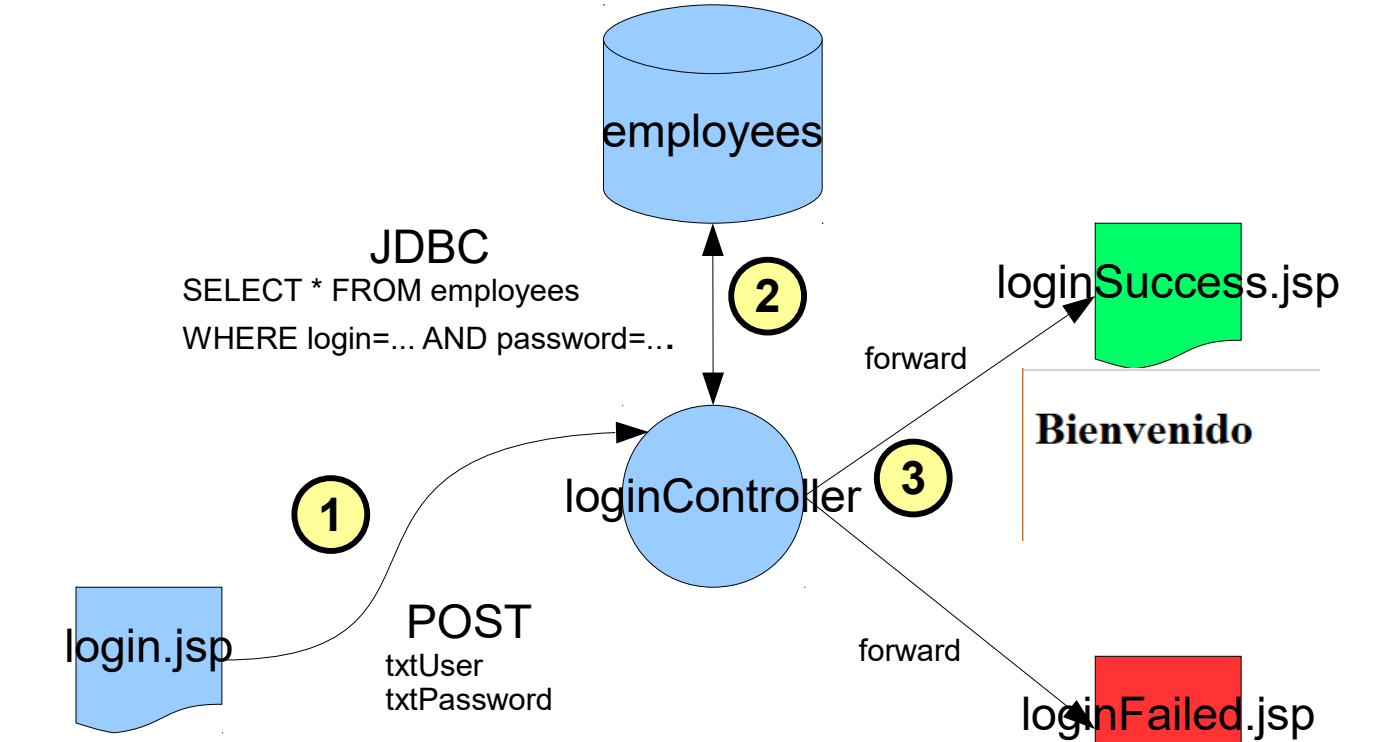
```
<jsp:useBean id="loginBean"

class="es.cifpcm.loginmvc.web.model.Login" scope="request"/>
```

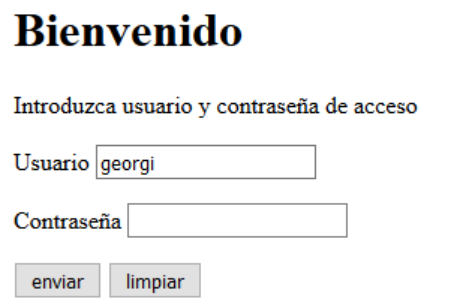
- Lo utilizamos para recuperar la propiedad login.

```
Bienvenido <jsp:getProperty name="loginBean" property="login"/>
```

3. Descripción de la actividad

En esta actividad vamos a crear un sistema de login básico que permitirá al usuario introducir su login y contraseña para validarse. La validación la realizará un servlet y en función de si la validación fue correcta devolverá la vista loginSuccess.jsp o si, por el contrario, fue incorrecta la vista loginFailed.jsp.





3.1 Comunicación entre web componentes (servlets, JSPs) a través de la request

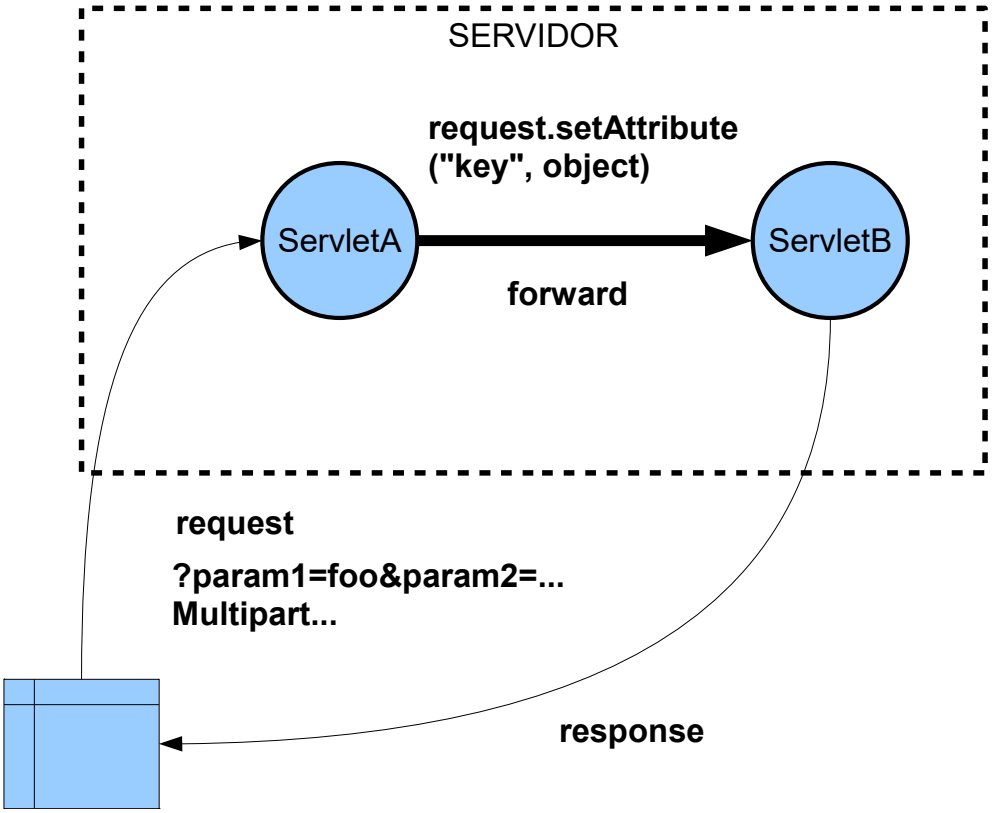
Así como la petición que nos viene del navegador está limitada (exceptuando AJAX) a contener elementos "simples" de información: parámetros de tipo texto (request.getParameter), de tipo multipart (request.getPart), ya a nivel de servidor, los servlet y JSP pueden pasarse información ya sea en el ámbito de request, sesión o aplicación.

Para definir un objeto en el ámbito de request, utilizaremos el método setAttribute indicando una clave para poder recuperar luego el objeto.

request.setAttribute(<*key*>, <*object*>);

Ejemplo:

request.setAttribute("loginBean", *loginObject*);



4. Crear la tabla login en la base de datos employees

Pegar y ejecutar el siguiente código.

```
USE employees;


CREATE TABLE employees.login

(

      login varchar(20) NOT NULL

    , password varchar(20) NOT NULL

    , emp_no int(11) NOT NULL

);


ALTER TABLE employees.login ADD PRIMARY KEY PK_login_login (login);


ALTER TABLE employees.login ADD CONSTRAINT FK_login_employees_empno FOREIGN KEY
(emp_no) REFERENCES

employees.employees (emp_no);

INSERT INTO employees.login (login, password, emp_no) VALUES ('georgi', 'georgi',
10001);

INSERT INTO employees.login (login, password, emp_no) VALUES ('parto', 'parto',
10003);
```

5. Crear la página jsp de login

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

    <head>

        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

        <title>Login</title>

    </head>
```

```
    <body>


        <h1>Bienvenido</h1>
        <p>Introduzca usuario y contraseña de acceso</p>
        <form action="loginController" method="post">
            <p>
            <label for="txtUser">Usuario</label>
            <input type="text" id="txtUser" name="txtUser"/>
            </p>
                        <p>
            <label for="txtPassword">Contraseña</label>
            <input type="password" id="txtPassword" name="txtPassword"/>
            </p>
            <p>
                <input type="submit" value="enviar"/>
                <input type="reset" value="limpiar"/>
            </p>
        </form>
    </body>
</html>
```

6. Crear el servlet controlador

```
package es.cifpcm.loginmvc.web.controller;


import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


/**
 *
 * @author Inma
 */
public class LoginControllerServlet extends HttpServlet {


    @Override
    public void init() throws ServletException {
        try {
            super.init();
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(LoginControllerServlet.class.getName()).log(Level.SEVERE
, null, ex);
        }
    }


    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
```

```
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
      protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {


                                                try      (Connection      conn      =
DriverManager.getConnection("jdbc:mysql://localhost:3306/employees",
                "2dawa", "2dawA2!06")) {


              String query = "SELECT login, emp_no FROM employees.login WHERE login=?
AND password=?";


            try (PreparedStatement pstmt = conn.prepareStatement(query)) {


                String user = request.getParameter("txtUser");
                String password = request.getParameter("txtPassword");


                pstmt.setString(1, user);
                pstmt.setString(2, password);


                try (ResultSet rs = pstmt.executeQuery()) {


                    if (rs.next()) {    // Hay registro luego login ok


                        getServletContext().getRequestDispatcher("/loginSuccess.jsp")
.forward(request, response);
```

```
                    } else {
                        getServletContext().getRequestDispatcher("/loginFailed.jsp").
forward(request, response);

                    }
                }
            }

        } catch (SQLException ex) {
            Logger.getLogger(LoginControllerServlet.class.getName()).log(Level.SEVERE
, null, ex);
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the
+ sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

7. web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app           version="3.1"           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-app_3_1.xsd">
    <servlet>
        <servlet-name>LoginControllerServlet</servlet-name>
            <servlet-class>es.cifpcm.loginmvc.web.controller.LoginControllerServlet</
servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>LoginControllerServlet</servlet-name>
        <url-pattern>/loginController</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>login.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

8. Crear las páginas de éxito y fracaso

8.1 LoginSuccess.jsp

```
<%--
    Document   : loginSuccess
    Created on : 07-nov-2016, 22:40:39
    Author     : Inma
--%>
```

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<jsp:useBean id="loginBean" class="es.cifpcm.loginmvc.web.model.Login" scope="request"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Bienvenido <jsp:getProperty name="loginBean" property="login"/></h1>
    </body>
</html>
```

8.2 loginFailed.jsp

```
<%--
    Document   : loginFailed
    Created on : 07-nov-2016, 22:40:48
    Author     : Fer
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
```

```
        <h1>Fuera ladr&oacute;n</h1>

    </body>

</html>
```