

## Lab 04

### Sử dụng bộ thẻ JSTL, Phân trang

#### Mục tiêu

- Thực hành với bộ thẻ JSTL của JSP
- Phân trang (pagination)

#### Phần I Bài tập step by step

##### STEP 1: Tạo project web tên JSPPagination

File -> New Project -> Java Web -> Web Application

Chọn server: GlashFish 4.1

Chọn Java EE Version 7

Finish

##### STEP 2: Tạo connection tới CSDL ProductDB (đã tạo trên MySQL hoặc SQLServer từ Lab 03)

Kiểm tra xem đã tồn tại JNDI “jdbc/ProductDB” trên JNDI DataSource của GlassFish server chưa. Nếu chưa có thì tạo JNDI này.

Resources (10)				
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="New..."/>	<input type="button" value="Delete"/>	<input type="button" value="Enable"/> <input type="button" value="Disable"/>
Select	JNDI Name	Logical JNDI Name	Enabled	
<input type="checkbox"/>	derby/PlayerDB		✓	
<input type="checkbox"/>	jdbc/BookStore		✓	
<input type="checkbox"/>	jdbc/ProductDB		✓	
<input type="checkbox"/>	jdbc/SQLServerProducDB		✓	
<input type="checkbox"/>	jdbc/__TimerPool		✓	
<input type="checkbox"/>	jdbc/__default	java:comp/DefaultDataSource	✓	
<input type="checkbox"/>	jdbc/sample		✓	
<input type="checkbox"/>	mysql/BankTransfer		✓	
<input type="checkbox"/>	mysql/BookStore		✓	
<input type="checkbox"/>	mysql/eMarket		✓	

Config JNDI trong web.xml để tạo connection tới CSDL ProductDB.

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

<web-app 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"
  version="3.1">
  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
  <resource-ref>
    <description>DB Connection</description>
    <res-ref-name>jdbc/ProductDB</res-ref-name>
    <res-type>javax.sql.ConnectionPoolDataSource</res-type>
    <res-auth>Container</res-auth>
  </resource-ref>
</web-app>
```

### STEP 3: Tạo entity Product biểu diễn table Product của CSDL.

Tạo package model -> Tạo class Product.java

```
package model;

public class Product {
    private int productID;
    private String name;
    private int quantity;
    private int price;
    private int categoryID;
}
```

Tạo các **getter** và **setter** method cho các thuộc tính của **Product**. Sử dụng tính năng sinh code của Netbeans/Eclipse để thực hiện (chuột phải vào giữa không gian soạn thảo của class, chọn Generate getter and setter...)

Sau khi thực hiện một phần đoạn code như sau:

```
public class Product {
    private int productID;
    private String name;
    private int quantity;
    private int price;
    private int categoryID;

    public Product () {}

    /**
     * @return the productID
     */
    public int getProductID() {
        return productID;
    }

    /**
     * @param productID the productID to set
     */
    public void setProductID(int productID) {
        this.productID = productID;
    }
}
```

Tương tự cho các hàm **getter** và **setter** còn lại của các thuộc tính: **name**, **quantity**, **price**, **categoryID**

#### STEP 4: Tạo class ProductDAO truy xuất tới CSDL.

Trong bài này, pagination được thực hiện bằng phương pháp giới hạn số lượng bản ghi trả về (theo yêu cầu của người dùng).

Câu lệnh SQL: "select SQL\_CALC\_FOUND\_ROWS \* from ProductDB.Product limit" + offset + ", " + numberOfRecords.

**Offset:** thứ tự của bản ghi

**NumberOfRecords:** số lượng bản ghi lấy về

Chúng ta sử dụng tùy chọn "limit" của câu lệnh SQL để giới hạn số lượng bản ghi lấy về.

Tùy chọn được **SQL\_CALC\_FOUND\_ROWS** thêm vào để biết số lượng bản ghi thực sự có thể được trả về khi không bị giới hạn bởi **limit**

Để biết số lượng bản ghi thực sự có thể trả về, gọi câu lệnh: "**SELECT FOUND\_ROWS()**"

Chúng ta sẽ thiết kế class **ProductDAO**, làm nhiệm vụ xử lý truy xuất CSDL, nhận về số

bản ghi theo tham số truyền vào thông qua phương thức là:

- **viewAllProducts(int offset, int noOfRecords)**

Phương thức này trả về một danh sách (**List**) các **Product** lấy được từ CSDL từ phần tử thứ **offset** với số lượng phần tử quy định trong **noOfRecords**.

Class **ProductDAO** sẽ có 1 thuộc tính là **noOfRecords**

**package** dao;

```
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;
import model.Product;

public class ProductDAO {
    private int noOfRecords;
    public ProductDAO () {}

    public List<Product> viewAllProducts(int offset, int noOfRecords)
    {
        /*...*/
    }

    public int getNoOfRecords() {
        return noOfRecords;
    }
}
```

```

public List<Product> viewAllProducts(int offset, int noOfRecords)
{
    String query = "select SQL_CALC_FOUND_ROWS * from ProductDB.Product limit " + offset + ", " + noOfRecords;
    List<Product> list = new ArrayList<Product>();
    Product product = null;
    try {
        Context initContext = new InitialContext();
        Context envContext = (Context) initContext.lookup("java:comp/env");
        DataSource ds = (DataSource) envContext.lookup("jdbc/ProductDB");
        Connection connection = ds.getConnection();
        Statement statement = connection.createStatement();
        ResultSet rs = statement.executeQuery(query);
        while (rs.next()) {
            product = new Product();
            product.setProductID(rs.getInt("product_id"));
            product.setName(rs.getString("name"));
            product.setQuantity(rs.getInt("quantity"));
            product.setPrice(rs.getInt("price"));
            product.setCategoryID(rs.getInt("category_id"));
            list.add(product);
        }
        rs.close();

        rs = statement.executeQuery("SELECT FOUND_ROWS()");
        if(rs.next())
            this.noOfRecords = rs.getInt(1);

    } catch (SQLException | NamingException ex) {
        System.err.println(ex);
    }
    return list;
}

```

## STEP 5: Tạo servlet để thực hiện pagination.

Tạo package control -> Tạo Servlet ProductServlet, urlpattern = “/product.do”

```
@WebServlet(name = "ProductServlet", urlPatterns = {"/product.do"})
public class ProductServlet extends HttpServlet {

    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        int page = 1;
        int recordsPerPage = 5;
        if(request.getParameter("page") != null)
            page = Integer.parseInt(request.getParameter("page"));
        ProductDAO dao = new ProductDAO();
        List<Product> list = dao.viewAllProducts((page-1)*recordsPerPage,
            recordsPerPage);

        int noOfRecords = dao.getNoOfRecords();
        int noOfPages = (int) Math.ceil(noOfRecords * 1.0 / recordsPerPage);
        request.setAttribute("productList", list);
        request.setAttribute("noOfPages", noOfPages);
        request.setAttribute("currentPage", page);
        RequestDispatcher view = request.getRequestDispatcher("viewProducts.jsp");
        view.forward(request, response);
    }
}
```

## STEP 6: Tạo trang jsp để view kết quả

Trang index.html:

```
<html>

    <head>

        <title>Home Page</title>

        <meta charset="UTF-8">

        <meta name="viewport" content="width=device-width, initial-
scale=1.0">

    </head>

    <body>

        <div><a href="product.do">Product View</a></div>

    </body>

</html>
```

Trang viewProducts.jsp sẽ gồm 3 phần:

- Phần 1 chứa link tới các trang đã xem trước đó
- Phần 2 hiển thị bảng
- Phần 3 hiển thị link đến các trang tiếp theo

Chúng ta sẽ sử dụng bộ thẻ JSTL để thực hiện.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>View Products</title>
</head>
<body>
    <table border="1" cellpadding="5" cellspacing="5">
        <tr>
            <th>Product ID</th>
            <th>Product Name</th>
            <th>Quantity</th>
            <th>Price</th>
        </tr>

        <c:forEach var="product" items="${productList}">
            <tr>
                <td>${product.productID}</td>
                <td>${product.name}</td>
                <td>${product.quantity}</td>
                <td>${product.price}</td>
            </tr>
        </c:forEach>
    </table>
```

```

<!--For displaying Previous link except for the 1st page -->
<c:if test="${currentPage != 1}">
    <td><a href="product.do?page=${currentPage - 1}">Previous</a>
    </td>
</c:if>

<!--For displaying Page numbers.
The when condition does not display a link for the current page-->
<table border="1" cellpadding="5" cellspacing="5">
    <tr>
        <c:forEach begin="1" end="${noOfPages}" var="i">
            <c:choose>
                <c:when test="${currentPage eq i}">
                    <td>${i}</td>
                </c:when>
                <c:otherwise>
                    <td><a href="product.do?page=${i}">${i}</a></td>
                </c:otherwise>
            </c:choose>
        </c:forEach>
    </tr>
</table>

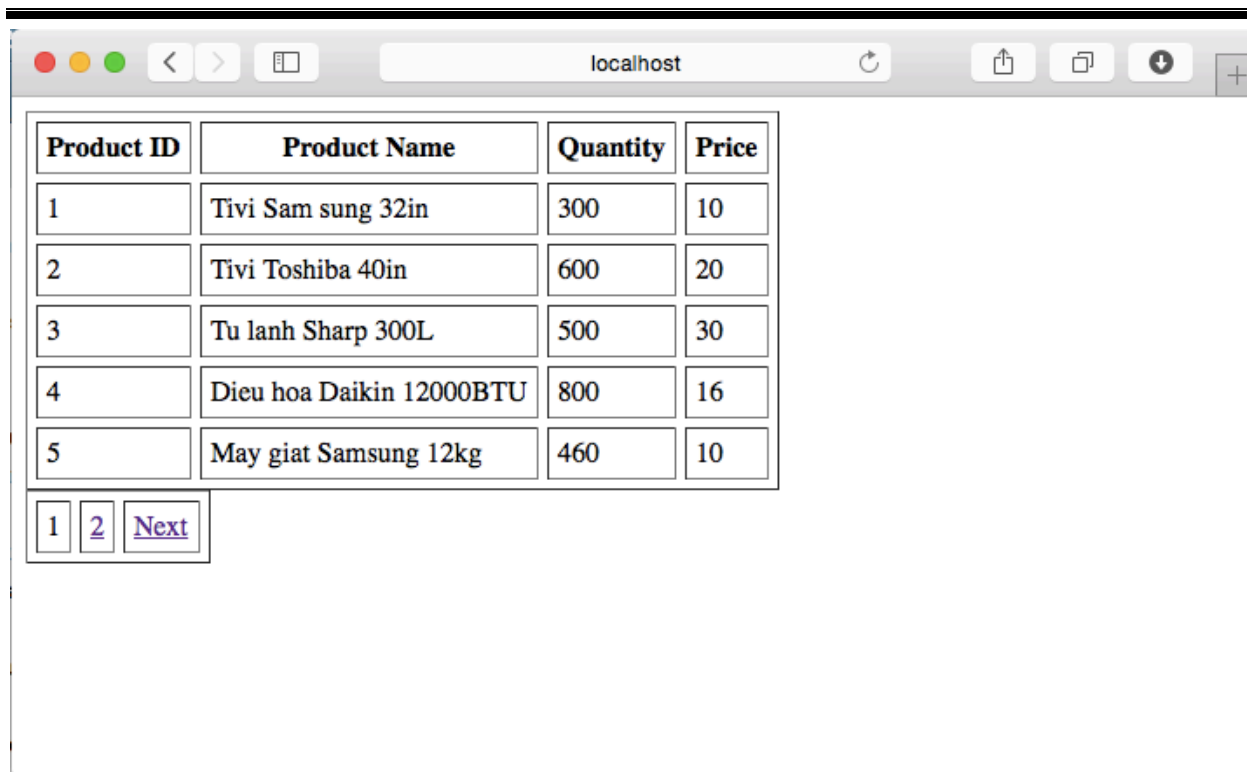
<!--For displaying Next link -->
<c:if test="${currentPage lt noOfPages}">
    <td><a href="product.do?page=${currentPage + 1}">Next</a></td>
</c:if>

</body>
</html>

```

Chạy chương trình, kết quả:





The screenshot shows a web browser window with the address bar set to 'localhost'. Inside the browser, there is a table with 4 columns: Product ID, Product Name, Quantity, and Price. The table contains 5 rows of data. Below the table, there is a pagination control with three buttons: '1', '2', and 'Next'. The '2' button is highlighted, indicating the current page.

Product ID	Product Name	Quantity	Price
1	Tivi Sam sung 32in	300	10
2	Tivi Toshiba 40in	600	20
3	Tu lanh Sharp 300L	500	30
4	Dieu hoa Daikin 12000BTU	800	16
5	May giat Samsung 12kg	460	10

1 2 Next

Note: cách khác để tạo pagination với JSP và Servlet là sử dụng thẻ xây dựng sẵn: pagination-tag. Đọc thêm: <http://www.javaworld.com/article/2072877/swing-gui-programming/paging-long-lists.html#resources>

## PHẦN 2: BÀI TẬP THỰC HÀNH

Tiếp tục bài thực hành ở Lab 03, thực hiện phân trang đối với danh sách sinh viên có trong CSDL.

**HẾT**