Bài 5: JSP custom tag

Nôi dung bài học

- Custom tags là gì?
- Tại sao cần Custom tags?
- TAG LIBRARY
- Simple Custom Tag
- Attibute Custom Tag

Custom tags là gì?

- Là các phần tử JSP do User tự định nghĩa (ngược với các thẻ chuẩn tắc: standard tags)
- Đóng gói các tác vụ phải thực hiện nhiều lần
- Lấy từ thư viện thẻ (tag library) tự định nghĩa

Custom Tags có thể

- Được tùy biến thông qua các thuộc tính truyền từ trang JSP gọi chúng
- Có thể truyền lại tham số cho trang gọi
- Truy cập được tất cả các đối tượng có trong trang JSP
- Được lồng vào nhau, giao tiếp thông qua các biến cục bộ

Ví dụ các Custom Tag

- Thiết lập/truy cập các Implicit objects
- Xử lý forms
- Truy cập database
- Điều khiển (rẽ nhánh, vòng lặp)

Các thư viện thẻ (Custom Tag Library) đã có sẵn

- Java Standard Tag Library (JSTL)
- Tags for setting/getting attributes, iteration, etc
- Tags truy cập database
- Tags for internationalized formatting
- Tags for XML Jakarta-Taglibs

TAG LIBRARY

- Là tập các thẻ cùng chung mục đích
- Một hoặc nhiều thẻ có thể được đóng gói thành thư viện thẻ

TAG LIBRARY

- Các thành phần liên quan đến Tag Library
  - Tag Handler Class
    - Cách thức Tag xử lý
  - Tag Library Descriptor File
    - Mô tả Tag
  - JSP Page
    - Khai báo và sử dụng Tag

TAG LIBRARY

- Tag Handler Class
  - Tag Handler chịu trách nhiệm xử lý trên Tag, chuyển Tag thành mã nguồn Java
  - Tag Handler phải cài đặt lại javax.servlet.jsp.tagext.Tag
  - Thường kế thừa từ TagSupport hoặc BodyTagSupport
  - Tag Handler để trong Source Packages của ứng dụng web giống như Servlets, Java Beans
-

TAG LIBRARY

- Tag Library Descriptor File
  - File XML mô tả
    - Tag Name
    - Các Attribute
    - -Chỉ định Tag Handler Class
  - Thư viên được khai báo và sử dụng các Tag trong trang JSP
-

TAG LIBRARY

- JSP Page
  - Import Tag Library - Tham chiếu đến URL của TLD
  - Khai báo Tag Prefix Sử dụng các Tag
-

SIMPLE CUSTOM TAG

- Đặc điểm
  - Không có thuộc tính
  - Không có thân
- Định dạng
  - <prefix:TagName/>

SIMPLE CUSTOM TAG

- Các bước xây dựng và sử dụng
- Bước 1: Tạo lớp Tag Handler Class kế thừa TagSupport trong Source Packages
- Bước 2: Tao Tag Library Descriptor (TLD) mô tả thông tin về Simple Custom Tag
  - - Có thể mô tả thông tin nhiều Tag
- Bước 3: Khai báo và sử dụng Simple Custom Tag trong JSP
-

SIMPLE CUSTOM TAG

- B1:Tạo Tag Handler Class kế thừa TagSupport

```
1  package packageName;
2  import java.io.IOException;
3  import java.util.logging.Level;
4  import java.util.logging.Logger;
5  import javax.servlet.jsp.JspWriter;
6  import javax.servlet.jsp.tagext.TagSupport;
7
8  public class TenTag extends TagSupport {
9      @Override
10     public int doStartTag() {
11         . . .
12         return SKIP_BODY;
13     }
14     @Override
15     public int doEndTag() {
16         return EVAL_PAGE;
17     }
```

SIMPLE CUSTOM TAG

- Bước 1: Tạo lớp Tag Handler Class
- Trong Tag Handler Class
  - Cài đặt lại doStartTag()
    - Xử lý khi bắt đầu mở Tag
    - Return SKIP_BODY
  - Cài đặt lại doEndTag()
    - Xử lý sau khi kết thúc thân Tag
    - Return EVAL_PAGE
- Trong doStartTag và doEndTag có thể sử dụng
  - thuộc tính pageContext kế thừa được từ TagSupport
-

SIMPLE CUSTOM TAG

- B1:Tạo Tag Handler Class kế thừa TagSupport
  - Một số phương thức hay dùng với pageContext
-

Đối tượng	Phương thức
out	JspWriter getOut()
session	HttpSession getSession()
request	ServletRequest getRequest()
response	ServletResponse getResponse()
exception	Exception getException()
errorData	ErrorData getErrorData()
page	Object getPage()
config	ServletConfig getServletConfig()
context	ServletContext getServletContext()

SIMPLE CUSTOM TAG

- B1:Tạo Tag Handler Class kế thừa TagSupport
  - Include và Forward với pageContext

Thao tác	Phương thức
Include	void include(String relativeURLPath)
Forward	void forward(String relativeURLPath)

SIMPLE CUSTOM TAG

- **B2: Tạo Tag Library Descriptor (TLD)**
  - File XML mô tả
    - Tag Name
    - Các Attribute
    - -Chỉ định Tag Handler Class
  - Tạo TLD trong Web Pages
-

SIMPLE CUSTOM TAG

- **B2: Tạo Tag Library Descriptor (TLD)**

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <taglib version="2.1"
3  xmlns="http://java.sun.com/xml/ns/javaee"
4  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
6  http://java.sun.com/xml/ns/javaee/web-
7  jsptaglibrary_2_1.xsd">
8     <tlib-version>1.0</tlib-version>
9     <short-name>Short Name</short-name>
10    <uri>URI</uri>
11    <tag>
12          <name>Tag Name</name>
13          <tag-class>Tag Handler Class</tag-class>
14          <body-content>empty</body-content>
15    </tag>
16 </taglib>
17
18
```

SIMPLE CUSTOM TAG

- **B2: Tạo Tag Library Descriptor (TLD)**
  - <body-content>empty</body-Content> : Không có thân
-

SIMPLE CUSTOM TAG

- **B3: Khai báo và sử dụng trong JSP**
- Khai báo
  - <%@taglib
    prefix=nprefixName"
    uri="uri" %>
- Sử dụng
  - <prefixName:TagName/>
-

Ví dụ: SIMPLE CUSTOM TAG

- **B1:Tạo Tag Handler Class kế thừa TagSupport**

```
1  public class HelloTag extends TagSupport {
2      @Override
3      public int doStartTag() {
4          JspWriter out = this.pageContext.getOut();
5          try {
6              out.println("<b>Xin chào Nguyễn Hoàng Anh</b>");
7              out.println("<b>This is a Simple Custom Tag</b>");
8          } catch (IOException ex) {
9              Logger.getLogger(HelloTag.class.getName())
10                     .log(Level.SEVERE, null, ex);
11         }
12         return SKIP_BODY;
13     }
14     @Override
15     public int doEndTag() {
16         return EVAL_PAGE;
17     }
18 }
```

Ví dụ: SIMPLE CUSTOM TAG

- **B2: Tạo Tag Library Descriptor (TLD)**

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <taglib version="2.1"
3  xmlns="http://java.sun.com/xml/ns/javaee"
4  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
6  http://java.sun.com/xml/ns/javaee/web-
7  jsptaglibrary_2_1.xsd">
8     <tlib-version>1.0</tlib-version>
9     <short-name>customtaglibrary</short-name>
10    <uri>/WEB-INF/tlds/CustomTagLibrary</uri>
11    <tag>
12          <name>HelloTag</name>
13          <tag-class>customs.HelloTag</tag-class>
14          <body-content>empty</body-content>
15    </tag>
16 </taglib>
```

Ví dụ: SIMPLE CUSTOM TAG

- **B3: Khai báo và sử dụng trong JSP**
- Khai báo

```
<%@taglib
        prefix=nnhanh"
        uri=n/WEB-INF/tlds/CustomTagLibrary" %>
```

- Sử dụng

```
<nhanh:HelloTag/>
```

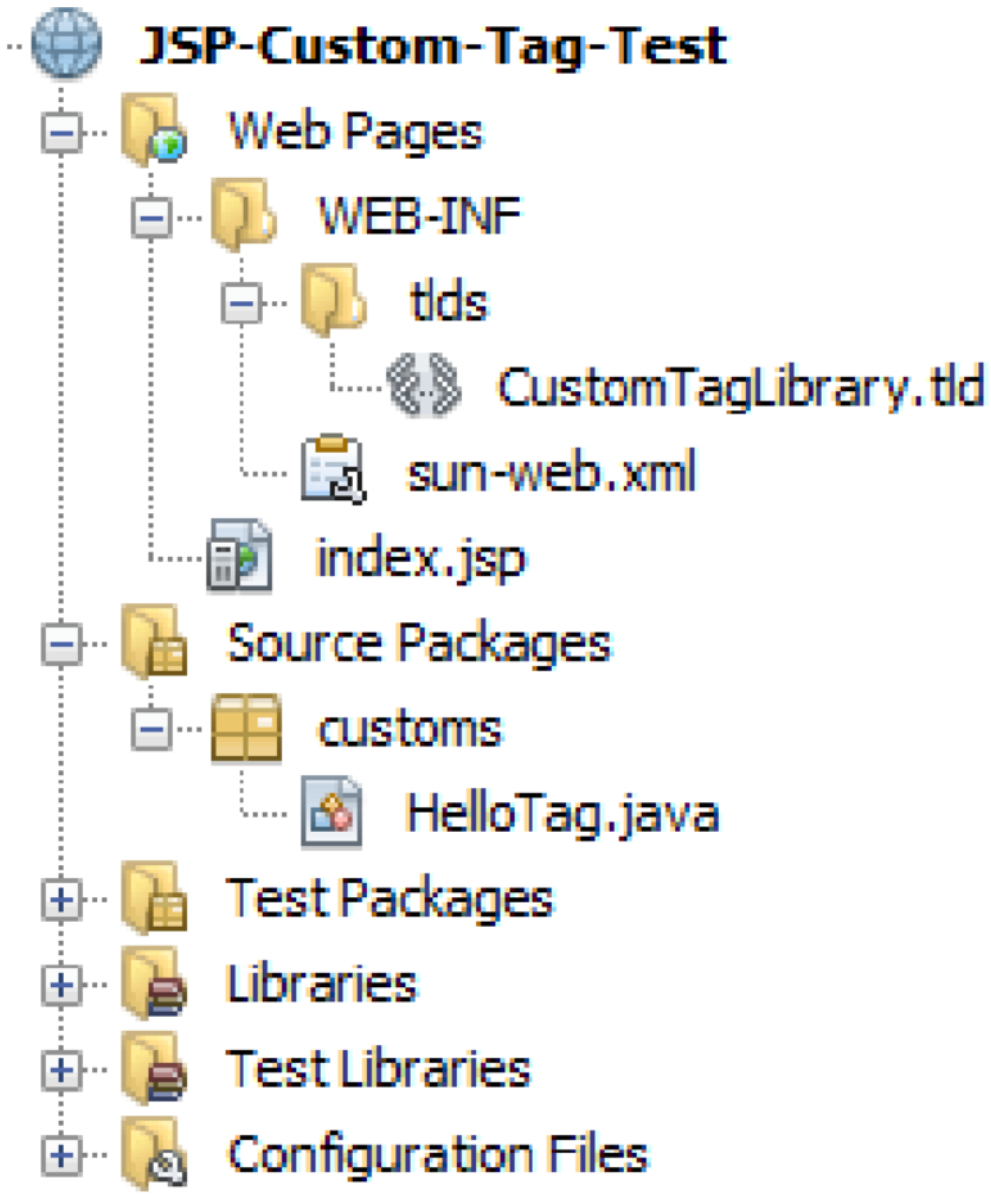
Ví dụ: SIMPLE CUSTOM TAG

- **B3: Khai báo và sử dụng trong JSP**

```
1  <%@page contentType="text/html" pageEncoding="UTF-8"%>
2  <%@taglib prefix="nhanh"
3            uri="/WEB-INF/tlds/CustomTagLibrary.tld" %>
4  <html>
5      <head>
6          <meta http-equiv="Content-Type"
7                content="text/html; charset=UTF-8">
8              <title>JSP Page</title>
9          </head>
10     <body>
11         <nhanh:HelloTag/>
12     </body>
13 </html>
```

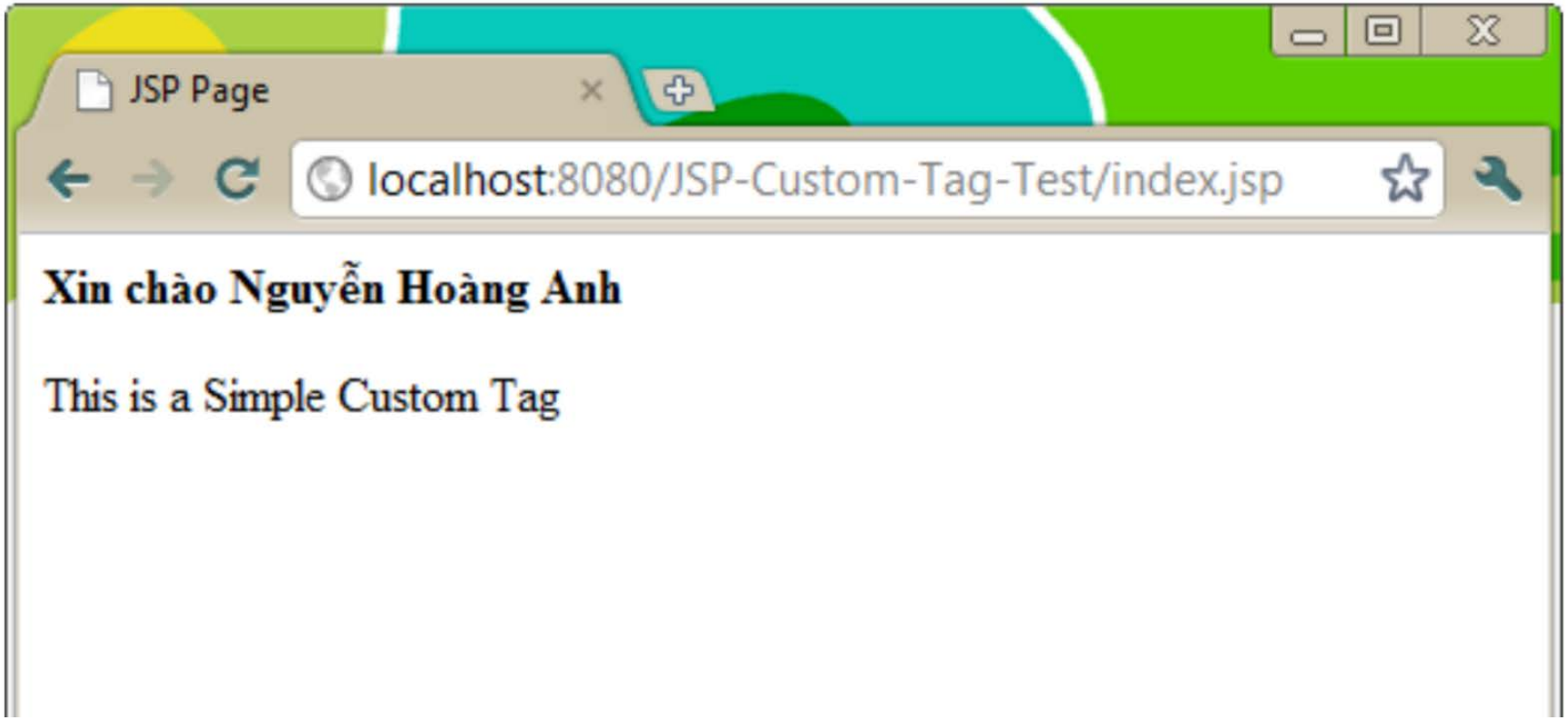
Ví dụ: SIMPLE CUSTOM TAG

Kết quả



Ví dụ: SIMPLE CUSTOM TAG

Kết quả



CUSTOM TAG WITH ATTRIBUTES

- Custom Tag có thuộc tính
- Đặc điểm
  - Có một hay nhiều thuộc tính
  - Không có thân
- Định dạng
  - <prefix:TagName attribute1 =“... ”
    attrhbute2=“...”
    attributeN=“...”
    />
-

CUSTOM TAG WITH ATTRIBUTES

- Các bước xây dựng và sử dụng
- Bước 1: Tạo lớp Tag Handler Class kế thừa TagSupport trong Source Packages
- Bước 2: Tao Tag Library Descriptor (TLD) mô tả thông tin về Simple Custom Tag
  - Có thể mô tả thông tin nhiều Tag
- Bước 3: Khai báo và sử dụng Custom Tag có thuộc tính trong JSP
-

CUSTOM TAG WITH ATTRIBUTES

- **B1:Tạo Tag Handler Class kế thừa TagSupport**
  - Để tạo được Custom Tag có thuộc tính

<prefix:TagName attribute1="..." attribute2="..."
... attributeN="..."/>

  - Trong Tag Handler Class
    - Thêm thuộc tính attributel, attribute2,...,atrributeN
    - Thêm các phương thức:

      public void setAttributel (String value1){...}
      public void setAttribute2 (String value2){...}
      public void setAttributeN(String valueN){...}

- - Cài đặt lại phương thức doStartTag, doEndTag
-

CUSTOM TAG WITH ATTRIBUTES

- **Bước 1: Tạo lớp Tag Handler Class**
  - Trong Tag Handler Class
    - Cài đặt lại doStartTag()
      - Xử lý khi bắt đầu mở Tag
      - Return SKIP_BODY
    - Cài đặt lại doEndTag()
      - Xử lý sau khi kết thúc thân Tag
      - Return EVAL_PAGE
  - Trong doStartTag và doEndTag có thể sử dụng -thuộc tính pageContext kế thừa được từ TagSupport
-

CUSTOM TAG WITH ATTRIBUTES

- **B1:Tạo Tag Handler Class kế thừa TagSupport**
- Một số phương thức hay dùng với pageContext
-

Đối tượng	Phương thức
out	JspWriter getOut()
session	HttpSession getSession()
request	ServletRequest getRequest()
response	ServletResponse getResponse()
exception	Exception getException()
errorData	ErrorData getErrorData()
page	Object getPage()
config	ServletConfig getServletConfig()
context	ServletContext getServletContext()

CUSTOM TAG WITH ATTRIBUTES

- **B1:Tạo Tag Handler Class kế thừa TagSupport**
-

Thao tác	Phương thức
Include	void include(String relativeURLPath)
Forward	void forward(String relativeURLPath)

CUSTOM TAG WITH ATTRIBUTES

- **B1:Tạo Tag Handler Class kế thừa TagSupport**

```
1  public class TagName extends TagSupport {
2      private String attribute1;
3      private String attribute2;
4      ...
5      private String attributeN;
6      public TagName(){
7          this.attribute1=...
8          this.attribute2=...
9          . . .
10         this.attributeN=...
11     }
12     public void setAttribute1(String value1){attribute1=value1;}
13     public void setAttribute2(String value2){attribute2=value2;}
14     ...
15     public void setAttributeN(String valueN){attributeN=valueN;}
16     ...
17 }
```

CUSTOM TAG WITH ATTRIBUTES

- **B1:Tạo Tag Handler Class kế thừa TagSupport**

```
1  public class TagName extends TagSupport {
2      @Override
3      public int doStartTag() {
4          //Sử dụng các thuộc tính
5          ...
6          return SKIP_BODY;
7      }
8      @Override
9      public int doEndTag() {
10         return EVAL_PAGE;
11     }
12     @Override
13     public void release(){
14         this.attribute1=...
15         this.attribute2=...
16         . . .
17         this.attributeN=...
18         //Release ...
19     }
20 }
```

CUSTOM TAG WITH ATTRIBUTES

- **B2: Tạo Tag Library Descriptor (TLD)**

```
1  <taglib> <tlib-version>1.0</tlib-version>
2     <short-name>customtaglibrary</short-name><uri>URI TLD</uri>
3     <tag>
4        <name>Tag Name</name>
5        <tag-class>Tag Handler Class</tag-class>
6        <attribute>
7           <name>attribute1</name>
8           <required>true/false</required>
9           <rtexprvalue>true/false</rtexprvalue>
10       </attribute>
11       . . .
12       <attribute>
13          <name>attributeN</name>
14          <required>true/false</required>
15          <rtexprvalue>true/false</rtexprvalue>
16       </attribute>
17       <body-content>empty</body-content>
18    </tag>
19 </taglib>
```

CUSTOM TAG WITH ATTRIBUTES

- **B2: Tạo Tag Library Descriptor (TLD)**
  - <required>true</required> : Thuộc tính bắt buộc sử dụng trong Tag
  - <required>false</required> : Thuộc tính tùy chọn sử dụng trong Tag
  - <rtexprvalue>true</rtexprvalue> : Thuộc tính có thể gán giá trị bằng JSP Expression
  - <rtexprvalue>false</rtexprvalue> : Thuộc tính không được gán giá trị bằng JSP Expression
  - <body-content>empty</body-Content> : Không có thân
-

CUSTOM TAG WITH ATTRIBUTES

- **B3: Khai báo và sử dụng trong JSP**
- Khai báo

```
<%@taglib prefix=″prefixName"
                    uri="URI TLD″ %>
```

- Sử dụng

```
<prefixName:TagName attribute1="..."
                              attribute2="..."
                                        • • •
                              attributeN="..."/>
```

-

CUSTOM TAG WITH ATTRIBUTES

- **Lưu ý**
- Truyền dữ liệu là POJO hay mảng POJO cho Custom Tag có thuộc tính là POJO hoặc mảng POJO sử dụng

<%= Expression %>

- POJO

<% DanhMuc dm = . . . %>

<prefix:tagName attribute='<%= dm %>' . . .

- Danh sách POJO

<% ArrayList<DanhMuc> ds = ... %>

<prefix:tagName attribute='<%= ds %>'. . .

XIN CẢM ƠN!