1. **Custom tag là gì**

Bộ thư viện **JSTL** là bộ thư viện của Java phát triển dành cho việc xây dựng trang JSP. Bên cạnh bộ thư viện này, trang JSP còn cho phép người lập trình xây dựng một thư viện thẻ của mình: custom tag. **Custom tag** được mô tả chi tiết trong 1 file XML gọi là **tag library**, trong file XML chúng ta khai báo các thẻ có trong bộ custom tag và các class thực thi của từng thẻ. Các class thực thi được gọi là các tag handler; chúng ta có 2 loại **class handler**:

-         **Simple tag handler**: chỉ dùng cho các thẻ không có các thành phần chứa scripting và không có body

-         **Classic tag handler**: được sử dụng khi các thành phần của thẻ có chứa scripting.

Khi một trang JSP có chứa custom tag được chuyển đổi sang servlet, các custom tag sẽ được chuyển đổi sang các phương thức được khai báo trong tag handler. Khi Web container thực thi trang JSP, các phương thức này sẽ được gọi kèm theo.

1. **Ưu điểm của Custom tag:**

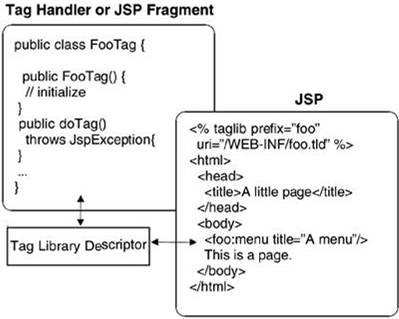
Custom tag có 3 ưu điểm chính sau:

-         **Tách rời các phương thức logic khỏi lớp View**: Custom tag và Scriptlet cung cấp các chức năng tương đương. Điểm khác biệt giữa 2 kỹ thuật này là chúng được viết ở nơi nào. Trong khi scriptlet cho phép nhúng trực tiếp 1 đoạn code vào trong trang JSP, custom tag là những đoạn code chạy phía sau để tổ chức lại HTML.

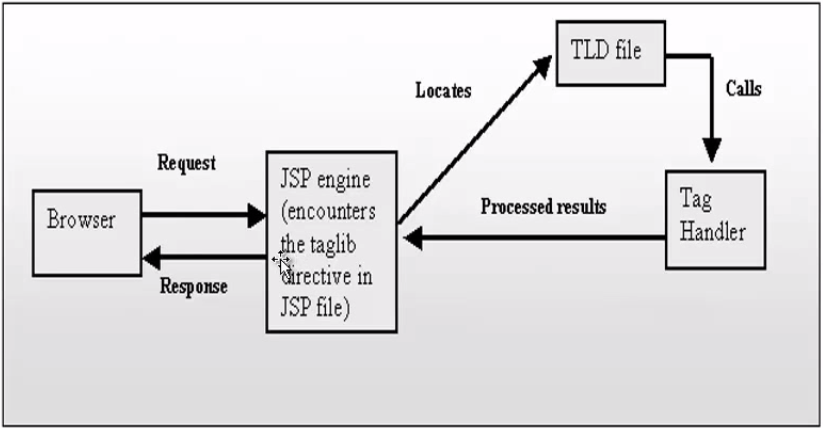
-         **Dễ sử dụng**: Custom tag rất dễ sử dụng cho người lập trình Java hay người không biết lập trình Java.

-         **Dễ cài đặt**: Custom tag sau khi xây dựng xong sẽ được đóng gói vào file jar và có thể sử dụng cho nhiều trang web khác nhau.

Hình sau mô tả các thành phần của Custom tag:



* 1. **Những thông tin nên biết về Custom Tags.**  
     Để có thể sử dụng tốt Custom Tags thì những thông tin sau đây sẽ rất có ích:  
     - JSP 1.0 không hỗ trợ “Tag Libralies”.  
     - JSP 1.1 hỗ trợ sự sáp nhập của người dùng được được tạo bởi Custom tags trong một file JSP.  
     - Cấu trúc của Custom Tags trong trang JSP cũng giống như XML tags.

1. **Mô hình làm việc của Custom Tags.**  
    [](http://2.bp.blogspot.com/-AazYFoR8qsI/U7lO7xYyElI/AAAAAAAAAQ4/BJQGTTwhOgc/s1600/ftrygv.png)

Như biểu đồ ta thây khi trình duyệt thực hiện Request lên server ,  request sẽ đi qua JSP engine sau đó. Trong trường hợp này JSP sẽ tìm trong TLD file để tìm hiểu đặc tả cho file đó.

Sau khi chúng ta có Tag rồi thì chúng ta sẽ biết chúng đượ c xử lấy ở đâu, bằng tag handler nào thì nó sẽ gọi đến tag handler đó  và xử lý,  sau đó trả kết quả về cho JSP Engine và JSP Engine trả về cho trình duyệt. Đó là một vòng đời của một JSP file với Custom Tags.

1. **Tag Library Descriptor (TLDs)**

Giải thích về Custom tag là một công việc khó khăn vì không biết bắt đầu từ đâu. **Tag Handler**, **.tag files** đều là những khái niệm quan trọng trong **Custom tag**; tuy nhiên, trước khi bắt tay vào các khái niệm này chúng ta cần nắm khái niệm về các file **TLD**. Vấn đề xuất hiện ở đây là một số khái niệm trong **TLD** sẽ không rõ nghĩa cho đến khi chúng ta nắm được khái niệm về **Tag Handler**. Chính vì vậy ở phần đầu tôi chỉ giải thích sơ lược về **TLD**, chúng ta sẽ bàn kỹ hơn ở các phần tiếp theo.

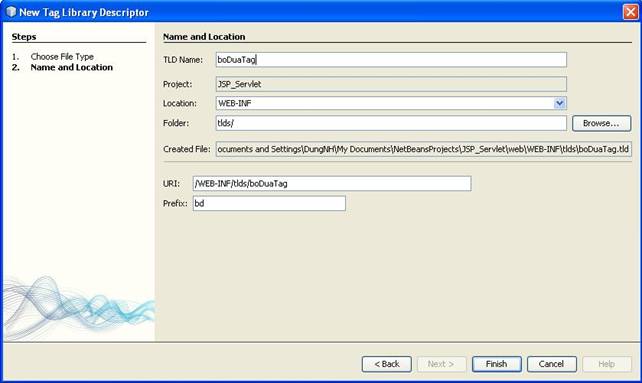
**TLDs là gì:**

Về cơ bản **TLD** là một kỹ thuật liên kết các đoạn code xử lý và các thẻ khai báo trên trang JSP. Để tạo 1 file **TLD**chúng ta làm theo các bước sau:

-         R.C vào tên project chọn **New/Other…**



-         Chọn **Tag Library Description** và **Next**, khai báo **TLD name**, **folde**r, **prefix**



Sau khi tạo file **TLD** chúng ta tiến hành khai báo các bước sau:

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

<**taglib** *version*="2.1" *xmlns*="http://java.sun.com/xml/ns/javaee"

*xmlns:xsi*="http://www.w3.org/2001/XMLSchema-instance"

*xsi:schemaLocation*="http://java.sun.com/xml/ns/javaee

          http://java.sun.com/xml/ns/javaee/web-jsptaglibrary\_2\_1.xsd">

            <**tlib-version**>**1.0**</**tlib-version**>

            <**short-name**>**bd**</**short-name**>

            <**uri**>**http://bodua.com**</**uri**>

            <**tag**>

                        <**name**>**table**</**name**>

                        <**tag-class**>**bodua.customTag.customTable**</**tag-class**>

                        <**body-content**>**empty**</**body-content**>

            </**tag**>

</**taglib**>

Qua đoạn code chúng ta có thể thấy file **TLD** thực chất chỉ là một tập hợp các khai báo về các thành phần của tag dưới dạng 1 file **XML**.

**Sử dụng Tag Library Descriptor:**

Để sử dụng file **TLD** chúng ta dùng 1 trong 2 cách sau:

-         Khai báo trực tiếp thông qua thành phần **directive taglig** của trang JSP:

**<%@ taglib** uri="**/WEB-INF/tlds/boDuaTag.tld**" prefix="**bd**" %>

-         Hoặc khai báo trong **web.xml**:

***<*jsp-config*>***

***<*taglib*>***

***<*taglib-uri**>**http://bodua.com*</*taglib-uri*>***

***<*taglib-location*>*/WEB-INF/tlds/boDuaTag.tld*</*taglib-location*>***

***</*taglib*>***

***</*jsp-config*>***

1. **Các loại thẻ Custom Tag:  
   Empty Tag**Vd. <td:welcome/>

Đây là thẻ không có body cũng như Attribute.  
**Custom tags with Attribute.**Vd: <td:welcome color= ‘blue”></tdwelcome>  
Đây là loại thẻ Custom Tags có Attribute color là “blue”.  
**Custom tags có body.**Vd:  
<td:welcome>  
 <%= today\_date %>  
</welcome>

**Custom tags lồng nhau.**

1. **Simple Custom Tag:**

Trong ví dụ đầu tiên tôi sẽ làm 1 ví dụ đơn giản là xuất ra 1 câu chào mừng một người nào đó và kèm theo ngày hiện hành. Chúng ta thực hiện theo 3 bước sau:

**Tạo file TLD:**

Khi tạo file **TLD** chúng ta cần hình dung tag của chúng ta có bao nhiêu property. Trong ví dụ này chúng ta sẽ có 2**property** là tên người cần chào mừng và kiểu thể hiện ngày tháng năm. Tạo 1 file tld có nội dung sau:

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

**<taglib** version**="2.1" xmlns="http://java.sun.com/xml/ns/javaee"**

xmlns:xsi**="http://www.w3.org/2001/XMLSchema-instance"**

xsi:schemaLocation**="http://java.sun.com/xml/ns/javaee**

**http://java.sun.com/xml/ns/javaee/web-jsptaglibrary\_2\_1.xsd">**

**<tlib-version>1.0</tlib-version>**

**<short-name>bd</short-name>**

**<uri>http://bodua.com</uri>**

**<tag>**

**<name>welcome</name>**

**<tag-class>bodua.customTag.customWelcome</tag-class>**

**<body-content>empty</body-content>**

**<attribute>**

**<name>ten</name>**

**</attribute>**

**<attribute>**

**<name>kieuNgay</name>**

**</attribute>**

**</tag>**

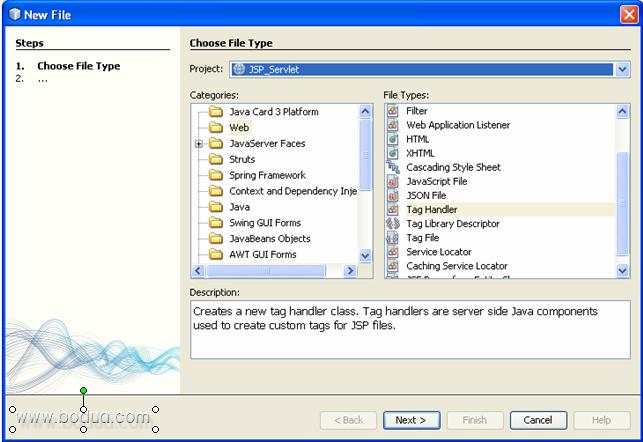
**</taglib>**

Để khai báo **attribute**cho **custom tag** chúng ta dùng thẻ **Attribute** và khai báo **name** cho Attribute.

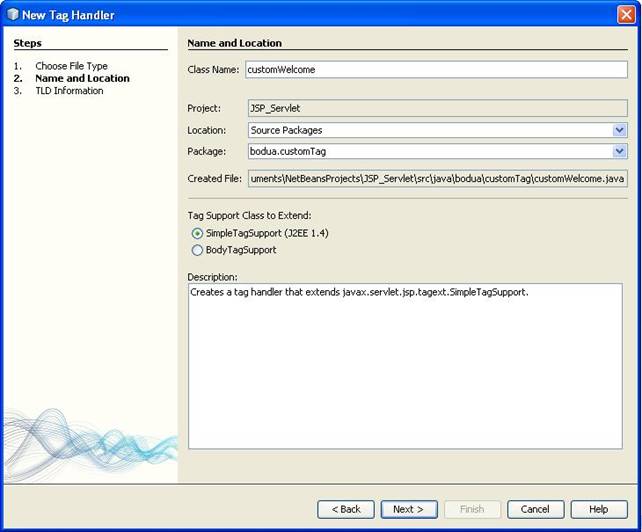
**Tạo Tag Handler:**

**Tag Handler** bản chất là 1 **Java class** **extends** từ 1 trong các class của Java dành cho Custom tag:**SimpleTagSupport** hay **BodyTagSupport**. Chúng ta tạo theo các bước sau:

-         R.C vào tên project chọn **New/Other**



-         Chọn Tag **Handler -> Next**:



Ở bước tiếp theo chúng ta bỏ chọn mục **Add Corresponding Tag to Tag Library Descriptor và Finish**. Viết đoạn code sau:

**public class** customWelcome **extends** **SimpleTagSupport** {

**private** String ten;

**private** String kieuNgay;

    /\*\*

     \* @return the ten

     \*/

**public** String getTen() {

**return** ten;

    }

    /\*\*

     \* @param ten the ten to set

     \*/

**public void** setTen(String ten) {

        this.ten = ten;

    }

    /\*\*

     \* @return the kieuNgay

     \*/

**public** String getKieuNgay() {

**return** kieuNgay;

    }

    /\*\*

     \* @param kieuNgay the kieuNgay to set

     \*/

**public** **void** setKieuNgay(String kieuNgay) {

        this.kieuNgay = kieuNgay;

    }

    /\*\*

     \* Hàm được gọi khi khai báo thẻ trên trang JSP

     \*/

    @Override

**public void** doTag() **throws** **JspException** {

**JspWriter** out = getJspContext().getOut();

**try** {

            // xét nếu tên không được khai báo thì dùng tên mặc định là "Không tên"

**if** (ten.isEmpty()) ten="Không tên";

            // xét nếu kiểu ngày không khai báo thì dùng kiểu ngày "dd/MM/yyyy"

**if** (kieuNgay.isEmpty()) kieuNgay="dd/MM/yyyy";

            // Đọc ngày hệ thống

            Date ngay= new Date();

            // chuyển đổi ngày sang dạng chuỗi

            String chuoiNgay= new SimpleDateFormat(kieuNgay).format(ngay);

            // dùng đối tượng out để xuất về phía client

            out.println("<h2>Hôm nay ngày "+chuoiNgay+"</h2>");

            out.println("<h1>Chào mừng "+ten+" đến với trang web Bó Đũa Group</h1>");

        } **catch** (Exception ex) {

**throw** new **JspException**("Error in customWelcome tag", ex);

        }

    }

}

**Viết trang JSP test thử:**

<%@**taglib** prefix="bd" uri="http://bodua.com"%>

<html>

    <head>

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

        <title>Welcome page</title>

    </head>

    <body>

        <**bd:welcome** ten="Nguyễn Minh Tân" kieuNgay="MM/dd/yyyy"/>

    </body>

</html>

Thi hành chúng ta thu kết quả sau:

