巨匠線上真人

Java Web OCE JWCD元件系統開發認證

巨匠線上真人

Java Web OCE JWCD元件系統開發認證

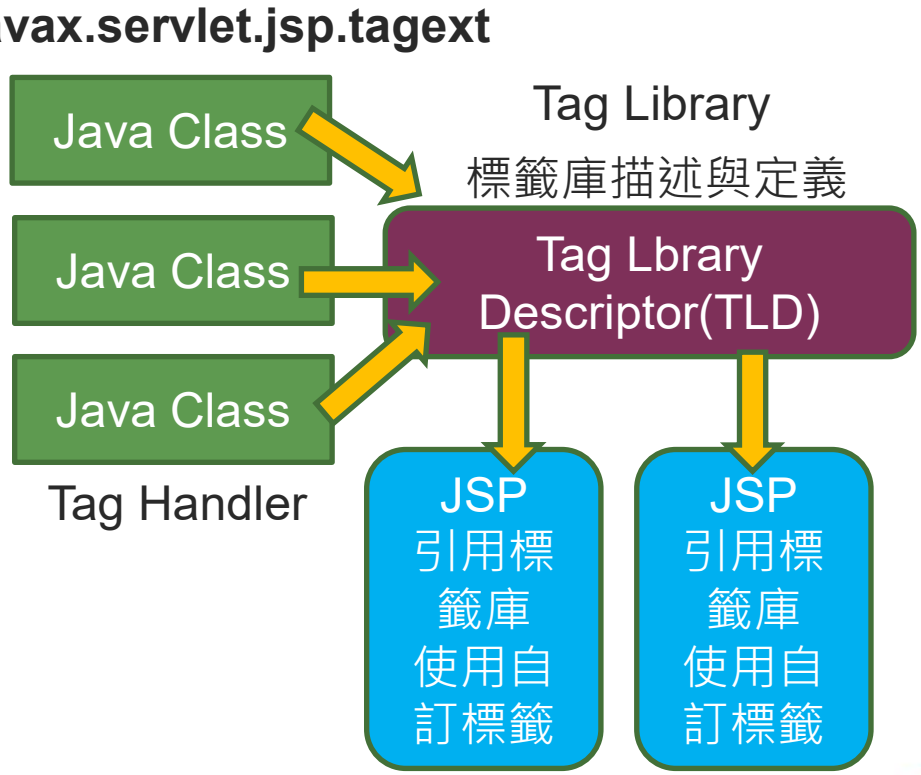
第十四堂：
JSP 自訂標籤開發與應用

本堂教學重點

- 自訂標籤庫設計架構與API說明
- 簡易型自訂標籤設定與Tag Library Descriptor組態
- Page引用Custom Tag架構與應用
- SimpleTagSupport與BodyTagSupport應用

自訂標籤庫設計架構與API說明

- 何謂自訂標籤(Custom Tag)
  - 避免在JSP直接撰寫Java Scriptlet，畢竟程式語言與Page中的Tag呈現隔閡距離。最後還是整個轉譯成Servlet Source。
  - 透過自訂標籤如同實現JavaBean元件概念，可以重複使用這些自訂標籤，呈現Page設計上的彈性與擴展性。
  - 因為是採用Tag方式，所以在可攜性被強化。
  - 採用XML語法，因此具有XML namespace語法延展。

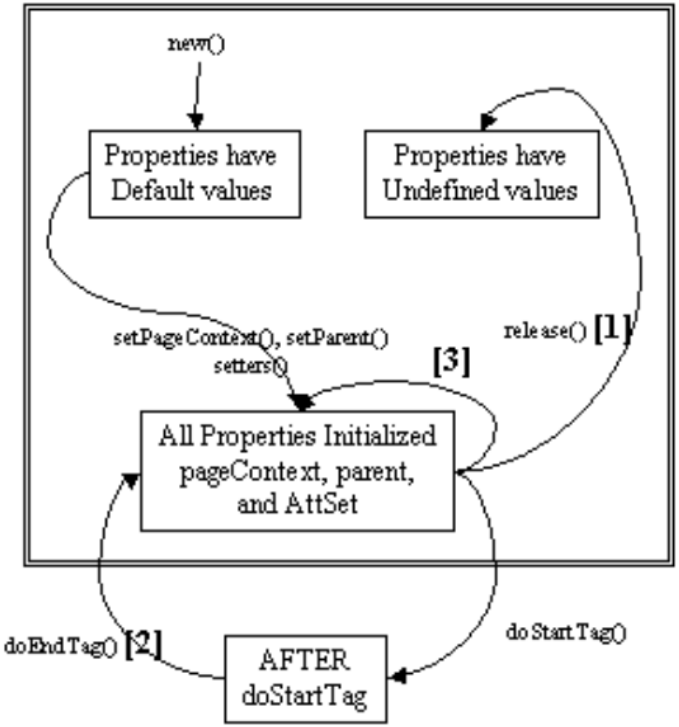


Tag Handler基準API

- 需要一個擴充標籤的API，來自Servlet API。
- javax.servlet.jsp.tagext
- 所有的自訂標籤的Base介面
  - javax.servlet.jsp.tagext. JspTag介面

簡易型自訂標籤設定與Tag Library Descriptor組態

- 實作javax.servlet.jsp.tagext.Tag介面
  - 開頭標籤與結束標籤等程序明確實作。
  - 明確注入PageContext物件形成與自訂標籤的互動。
  - 明確注入Parent Tag與上層標籤物件進行互動。



HelloTag Handler-實作Tag Interface

- 實作Tag介面，設計一個HelloTag Handler類別 。
- 透過Tag Library Descriptor進行描述。
- 使用JSP引用Custom Tag。

```java
@Override
public void setPageContext(PageContext pageContext) {
    this.pageContext=pageContext;

}

@Override
public void setParent(Tag parent) {
    this.parent=parent;

}
```

HelloTag.java

```java
//自訂標籤
public class HelloTag implements Tag{
    //attribute
    private Tag parent;
    private PageContext pageContext;

    @Override
    public int doEndTag() throws JspException {
        //透過PageContext取出Writer
        JspWriter out=this.pageContext.getOut();
        //寫出資訊
        try {
            out.println("<font size='5' color='blue'>大家好，我離開了!!</font>");
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return EVAL_PAGE;
    }

    @Override
    public int doStartTag() throws JspException {
        //透過PageContext取出Writer
        JspWriter out=this.pageContext.getOut();
        //寫出資訊
        try {
            out.println("<font size='5' color='blue'>大家好，我來了!!</font><br/>");
        } catch (IOException e) {
```

TLD描述

- 在WEB-INF/lib資料夾中新增一個TLD File。
- <uri>描述一個被JSP引用的URL或tld檔案相對位址。
- 透過<tag>元素定義Tag
- <body-content>具有四種定義類型:
  - empty
  - scriptless
  - JSP
  - tagdependent

customtag.tld

```
<?xml version="1.0" encoding="UTF-8"?>
<taglib version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
      web-jsptaglibrary_2_1.xsd">
    <tlib-version>1.0</tlib-version>
    <short-name>custom tag</short-name>
    <uri>http://www.gjun.com.tw/simple</uri>
    <tag>
        <!-- 說明 -->
        <description>打招呼</description>
        <!-- 自訂標籤名 -->
        <name>hello</name>
        <!-- Tag Handler類別 -->
        <tag-class>com.gjun.tag.HelloTag</tag-class>
        <!-- 標籤採用空元素架構 -->
        <body-context>empty</body-context>
    </tag>
</taglib>
```

<tag>相關子元素

Element	Description
description	(optional) Tag標籤說明。
display-name	(optional) 工具中顯示的名稱.
icon	(optional) 工具中顯示的Icon.
name	自訂標籤名，具有唯一性。
tag-class	完整敘述的類別來源，必須敘述package到類別(Tag Handler)
body-content	Tag中的body-content設定。區分empty/JSP/scriptless 支援 EL/tagdependent
attribute	描述自訂標籤中的attribute.
example	(optional) 自訂標籤的使用範例.

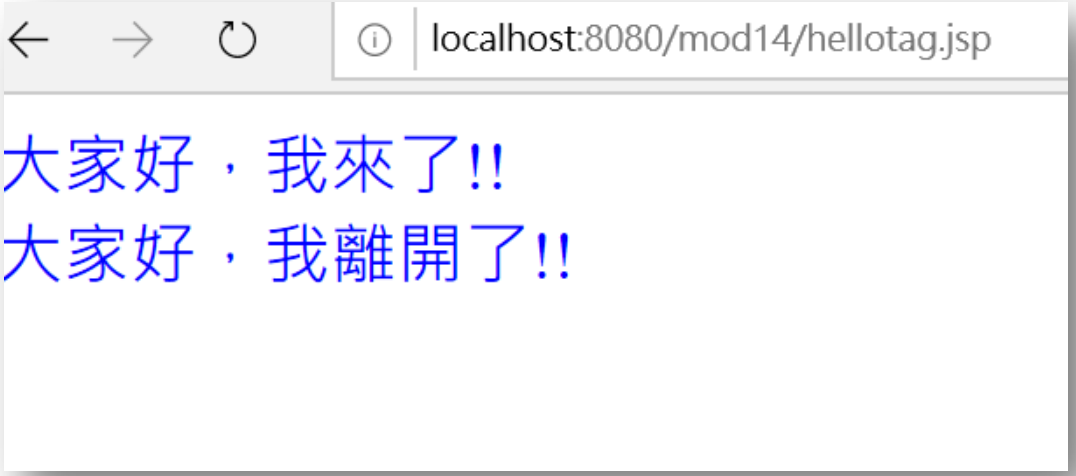
JSP引用CUSTOM Tag

- JSP使用<%@taglib%> Directive引用Tag Library。
- 使用uri attribute進行指定。
- 使用prefix attribute設定XML Namespace(前置詞)，用來提供引用Tag Library來源，須避開保留字，如jsp,java,javax等。

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="http://www.gjun.com.tw/simple" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Tag自訂標籤</title>
</head>
<body>
<div><h:hello/></div>
</body>
</html>
```

```xml
<taglib version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    web-jsptaglibrary_2_1.xsd">
  <tlib-version>1.0</tlib-version>
  <short-name>custom_tag</short-name>
  <uri>http://www.gjun.com.tw/simple</uri>
  <tag>
      <!-- 說明 -->
      <description>打招呼</description>
      <!-- 自訂標籤名 -->
      <name>hello</name>
      <!-- Tag Handler類別 -->
      <tag-class>com.gjun.tag.HelloTag</tag-class>
      <!-- 標籤採用空元素架構 -->
      <body-context>empty</body-context>
  </tag>
</taglib>
```

Demo



Page引用Custom Tag架構與應用

- 直接使用tld所在檔案目錄進行引用

  <%@ taglib
  uri="/WEB-INF/tld/hellotag.tld"
  prefix="h" %>

- 透過透過uri指定自訂的URL引用。
  - <%@taglib prefix="h"
    uri="http://www.gjun.com.tw/simple"
    %>
- 標籤須採用xml namespace寫法，讓JSP Actions引用到正確的Tag Lbrary來源。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="http://www.gjun.com.tw/simple" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Tag自訂標籤</title>
</head>
<body>
<div><h:hello/></div>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="/WEB-INF/tld/customtag.tld" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Tag自訂標籤</title>
</head>
<body>
<div><h:hello/></div>
</body>
</html>
```

設計一個具有Attribute的Custom Tag

- <xxx:hello attr1="值" attr2="${…}"/>
- 可以借助Attribute注入互動資訊，內容可以是固定值，亦可以設定為運算式結果。
- 強化Custom Tag設計階段運作注入互動資訊。
- Tag Handler類別規劃採用JavaBean setter/getter架構設計(Property注入)。

MsgHelloTag Handler:TagSupport

- 在Custom Tag Handler中必須繼承Tag介面與IterationTag介面。
- InterationTag主要是針對Tag Body執行之後可以引發處理的程序。
  - <xxx:tag>body…</xxx.tag>
- 所以API提供一個Adapter Pattern的TagSupport 類別支援。我們可以直接繼承TagSupport類別設計一個基底的Tag Handler。

MsgHelloTag Handler 設計

MsgHelloTag.java

```java
public class MsgHelloTag extends TagSupport{
    //attributte
    private String message;
    //<xxx>body</xxx> body執行之後
    @Override
    public int doAfterBody() throws JspException {
        // TODO Auto-generated method stub
        return super.doAfterBody();
    }

    //<xxx></xxx>結束標籤執行之後
    @Override
    public int doEndTag() throws JspException {
        //參考出PageContext物件
        PageContext pageContext=this.pageContext;
        //取出Writer
        JspWriter out=pageContext.getOut();
        String msg=String.format("<font size='6' color='red'>%s 您好!!</font>",message);
        try {
            out.println(msg);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return EVAL_PAGE; //回應旗標 繼續往下執行
    }
```

Setter and getter

```java
//getter and setter
public String getMessage() {
    return message;
}

public void setMessage(String message) {
    this.message = message;
}
```

TLD佈署與描述

- Tld File中使用<tag><attribute>元素進行屬性描述。
- <name>屬性名稱</name>，必須與類別中定義的Property同名稱(setName/getName,Property為name)。
- <required>true</require>元素描述該屬性必須在標籤中設定與否。選項項元素，預設值為false。
- <type>類型</type>描述該屬性資料型別。選項性元素，預設為java.lang.String。
- <rtexprvalue>false</rtexprvalue>元素設定是否支援運算式或EL。選項元素，預設值為false。

```
<!-- MsgHelloTag -->
<tag>
    <!-- 說明 -->
    <description>具有參數傳遞的打招呼</description>
    <!-- 自訂標籤名 -->
    <name>hellomsg</name>
    <tag-class>com.gjun.tag.MsgHelloTag</tag-class>
    <body-context>empty</body-context>
    <attribute>
        <name>message</name>
        <required>false</required>
        <type>java.util.Boolean</type>
        <rtexprvalue>false</rtexprvalue>
    </attribute>
</tag>
```

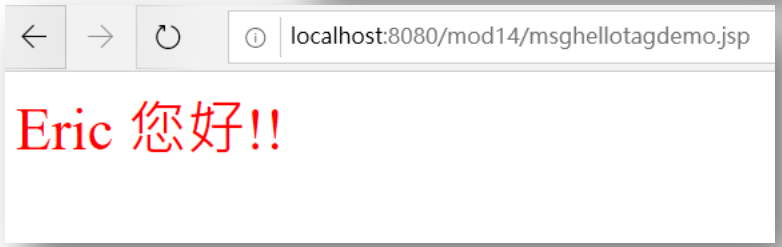
```
//getter and setter
public String getMessage() {
    return message;
}

public void setMessage(String message) {
    this.message = message;
}
```

JSP設計-具有Attribute Custom Tag

- 使用<%@taglb...%>引用Custom Tag。

msghellotagdemo.jsp

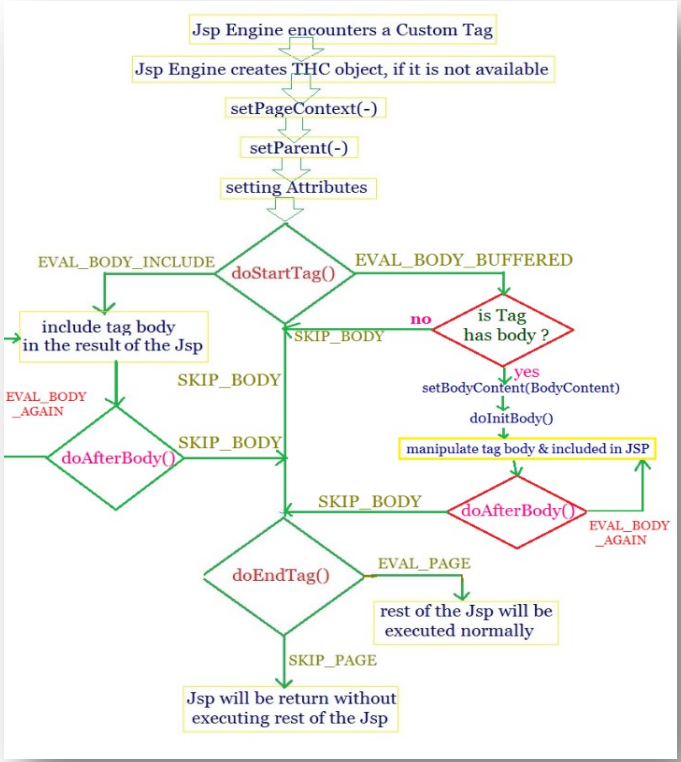
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://www.gjun.com.tw/simple" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>帶有屬性設定的Custom Tag</title>
</head>
<body>
<c:hellomsg message="Eric"/>
</body>
</html>
```



實作BodyTag Interface

- BodyTag介面，用來執行進入標籤的Body初始化與設定Body內容。以及執行Body之後相關處理。

方法	說明
	進入Body範圍之前的執行程序。
setBodyContent(BodyContent b)	注入BodyContent物件，透過BodyContent物件可以透過JspWriter物件進行Body輸出處理。
doAfterBody()	在Body處理之後可以透過這一個方法進行處理。



BodyTag介面範例

- 透過 doStartTag 回應 EVAL_BODY_INCLUDE，評量Body內容進入串流輸出。
- 在doAfterBody回應 EVAL_BODY_AGAIN，可以走訪Body內容進入串流輸出(迴圈處理)。

```
@Override
public int doStartTag() throws JspException {
    JspWriter writer=this.pageContext.getOut();
    try {
        writer.print("<h1>開始標籤</h1>");
    } catch (IOException e) {
        e.printStackTrace();
    }
    return EVAL_BODY_INCLUDE;
}
```

MyBodyTagHandler.java

```
public class MyBodyTagHandler implements BodyTag {
    private PageContext pageContext;
    private BodyContent bodyContent;
    private String message;
    private int counter=1;

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    @Override
    public int doAfterBody() throws JspException {
        //衡量執行次數
        if(counter>3) {
            return EVAL_PAGE;
        }else {
            counter++;
            return EVAL_BODY_AGAIN;
        }
    }
```

TLD佈署與描述

customtag.tld

```
<!-- BodyTag demo -->
<tag>
    <!-- 說明 -->
    <description>具有參數傳遞BodyTag</description>
    <!-- 自訂標籤名 -->
    <name>mybody</name>
    <tag-class>com.gjun.tag.MyBodyTagHandler</tag-class>
    <body-context>JSP</body-context>
    <attribute>
        <name>message</name>
         <required>true</required>
         <type>java.util.Boolean</type>
        <rtexprvalue>true</rtexprvalue>
    </attribute>
</tag>
```

支援運算式

bodytagdemo.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="/WEB-INF/tld/customtag.tld" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>BodyTag自訂標籤</title>
</head>
<%
    String message="Hello World";
    pageContext.setAttribute("message",message);
%>
<body>
<div><h:mybody message="${message}">
    <h2 style="font-size:48px;color:red">巨匠您好</h2>
</h:mybody></div>
</body>
</html>
```

BodyTag- EVAL_BODY_INCLUDE Demo

BodyTag- *EVAL_BODY_BUFFERED*應用

- 在doStartTag()回應EVAL_BODY_BUFFERED，可以透過注入的BodyCotent介面，取得JspWriter子類別進行Body輸出處理。

```java
@Override
public int doStartTag() throws JspException {
    JspWriter writer=this.pageContext.getOut();
    try {
        writer.print("<h1>開始標籤</h1>");
    } catch (IOException e) {
        e.printStackTrace();
    }
    return EVAL_BODY_BUFFERED;

}
```

```java
@Override
public void setBodyContent(BodyContent bodyContent) {
    this.bodyContent=bodyContent;

}
```

MyBodyTagHandler2.java

```java
public class MyBodyTagHandler2 implements BodyTag {
    private PageContext pageContext;
    private BodyContent bodyContent;
    private String message;
    private int counter=1;


    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    @Override
    public int doAfterBody() throws JspException {
        //取得Closed JspWriter
        JspWriter writer=this.bodyContent.getEnclosingWriter();
        try {
            writer.println(String.format("<h2>您好...%s</h2>",message));
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
```

JSP Custom Tag Demo

bodytagdemo2.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="/WEB-INF/tld/customtag.tld" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>BodyTag自訂標籤</title>
</head>
<%
    String message="Hello World";
    pageContext.setAttribute("message",message);
%>
<body>
<div><h:mybody2 message="${message}">
    <h2 style="font-size:48px;color:red">巨匠您好</h2>
</h:mybody2></div>
</body>
</html>
```

SimpleTagSupport 與BodyTagSupport應用

- *JSP 2.0* 增加了一個SimpleTagSupport類別，簡化流程無須透過繁瑣的Method，進行回應值(旗標)，如 SKIP_PAGE、EVAL_BODY_INCLUDE 、EVAL_PAGE、EVAL_BODY_AGAIN等，讓自訂標籤可以更精簡與方便設計。

方法	說明
doTag()	透過PageContext參照出JspWriter，直接在這一個Method撰寫輸出資訊。
getPageContext()	參照出注入的PageContext物件。

自訂SimpleTagSupport Tag handler-圖片呈現設定

MySimpleTagHandler.java

```java
public class MySimpleTagHandler extends SimpleTagSupport {
    //Attribute屬性
    private String url = "";
    private boolean showBorder = false;
    private boolean showUrl = false;
    //設定圖片來源
    public void setUrl(String url) {
        this.url = url;
    }
    //呈現邊線效果
    public void setShowBorder(boolean showBorder) {
        this.showBorder = showBorder;
    }
    //呈現圖片位址
    public void setShowUrl(boolean showUrl) {
        this.showUrl = showUrl;
    }
```

```java
    //Overriding doTag
    public void doTag() throws JspException {

        PageContext pageContext = (PageContext) getJspContext();
        JspWriter out = pageContext.getOut();

        try {
            StringBuffer sb = new StringBuffer();
            if (showUrl) {
                sb.append("<h3>");
                sb.append(url);
                sb.append("</h3>\n");
            }
            sb.append("<img width='200px' height='200px' ");
            if (showBorder) {
                sb.append("border=\"1\" ");
            }
            sb.append("alt=\"\" src=\"");
            sb.append(url);
            sb.append("\"/>");

            out.println(sb.toString());

        } catch (Exception e) {
            e.printStackTrace();
        }
```

TLD描述

customtag.tld

```
<!-- SimpleTagSupport Handler -->
<tag>
    <name>img</name>
    <tag-class>com.gjun.tag.MySimpleTagHandler</tag-class>
    <body-content>empty</body-content>
    <description>Display Image</description>
    <attribute>
        <name>url</name>
        <required>true</required>
        <rtexprvalue>true</rtexprvalue>
    </attribute>
    <attribute>
        <name>showBorder</name>
        <required>false</required>
        <rtexprvalue>false</rtexprvalue>
        <type>boolean</type>
    </attribute>
    <attribute>
        <name>showUrl</name>
        <required>false</required>
        <rtexprvalue>false</rtexprvalue>
        <type>boolean</type>
    </attribute>
</tag>
```

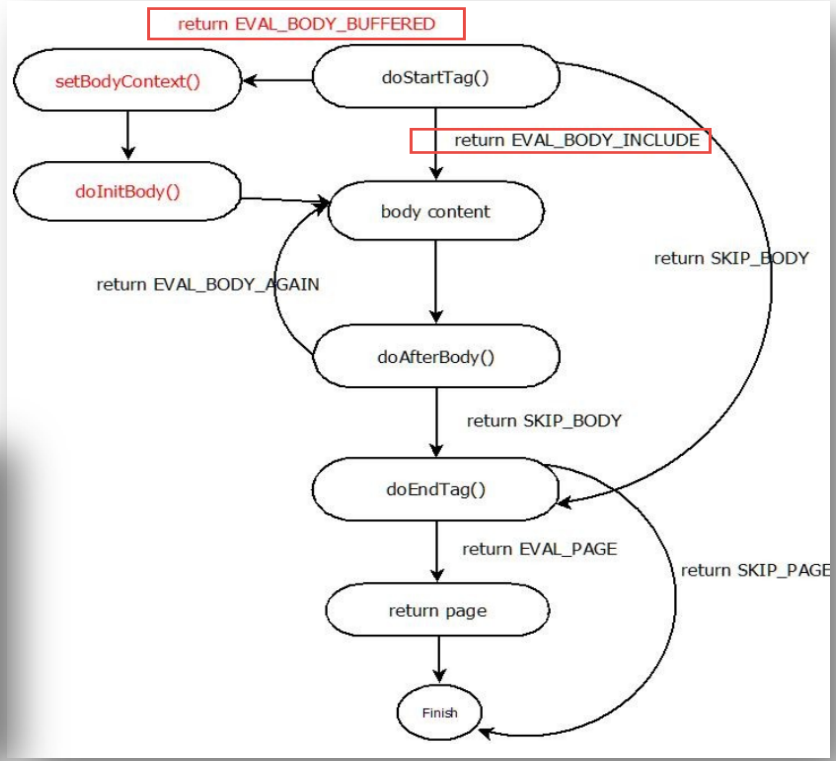
JSP Demo

simpletagsupportdemo.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="h" uri="/WEB-INF/tld/customtag.tld" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>SimpleTagSupport自訂標籤</title>
</head>

<body>
<div>
<table border="1" cellpadding="2" cellspacing="0">
    <tr>
        <td><h:img url="p1.jpg" showUrl="true" showBorder="true"/></td>

    </tr>
</table>
</div>
</body>
</html>
```

BodyTagSupport應用

- BodyTagSupport父類別為TagSupport。
- 實作了BodyTag Interface。
- 因此在處理Body方式與BodyTag Interface規範一樣。



Class BodyTagSupport

java.lang.Object
  └ javax.servlet.jsp.tagext.TagSupport
    └ javax.servlet.jsp.tagext.BodyTagSupport

All Implemented Interfaces:

BodyTag, IterationTag, JspTag, java.io.Serializable, Tag

BodyTagSupport 常用Method與預設實作狀態

方法	實作與回應值
int doStartTag()	繼承來自於Tag，預設回應值 EVAL_BODY_BUFFERED
int doAfterBody()	繼承來自於 IteratorTag，預設回應值 SKIP_BODY
int doEndTag()	繼承來自 Tag，預設回應值 EVAL_PAGE

BodyTagSupport應用-Body處理

- 讀取Body之後處理，Overriding doAfterBody() Method。
- 並且借助doStartTag()實作預設回應的EVAL_BODY_BUFFERED，可透過注入的pageContext取出JspWriter子類別進行輸出處理。

MyBodyTagSupportHandler.java

```
public class MyBodyTagSupportHandler extends BodyTagSupport {

    //Overriding doAfterBody
    public int doAfterBody() throws JspException
    {
        try {
            //參考出注入進來的BodyContent物件
            BodyContent bodycontent = getBodyContent();
            //取出Body內容
            String body = bodycontent.getString();
            //取出JspWriter子類別物件
            JspWriter out = bodycontent.getEnclosingWriter();
            //判斷Body已經有設定內容
            if(body != null) {
                //轉換成大寫輸出
                out.print(body.toUpperCase());
            }
        } catch(IOException ioe) {
            throw new JspException("錯誤:"+ioe.getMessage());
        }
        return SKIP_BODY;
    }
```

TLD佈署與描述

```
<!-- BodyTagSupport應用 -->
 <tag>
  <name>stringToUpper</name>
  <tag-class>com.gjun.tag.MyBodyTagSupportHandler</tag-class>
  <bodycontent>JSP</bodycontent>
  <info>轉換body內容為大寫</info>
</tag>
```

JSP應用

bodytagsupportdemo.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://www.gjun.com.tw/simple" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>BodyTagSupport應用</title>
</head>
<body>
<c:stringToUpper>
     I think that we all have times when life can get overwhelming and we need to lean on someone
</c:stringToUpper>
</body>
</html>
```

Lab

- 請說明自訂標籤設計過程與應有的架構為何
- Tag Library Descriptor如何描述一個具有Attribute定義的Tag
- JSP如何引用Custom Tag
- 撰寫SimpleTagSupport Tag handler-呈現Image