Pengantar: Standard Action

Chocolove Mic
chocolove_mic@yahoo.co.uk
http://www.mycgiserver.com/~chocolove2003

Standard Action adalah tag yang berfungsi untuk menjalankan suatu operasi yang spesifik, seperti :

- Mem-forward dari suatu halaman JSP ke halaman JSP yang lain.
- Mennyisipkan suatu halaman / operasi pada halaman JSP dari halaman JSP yang lain.
- Penanganan Applet atau Java Bean pada suatu halaman JSP (Java Bean akan dijelaskan lebih jauh pada materi yang akan datang).
- Dan lain-lain.

Standar action cara penulisannya mirip dengan aturan penulisan pada XML. Sintak dari Standard Action ini adalah :

```
<jsp:nama_aksi atribut1 atribut2 .... />
```

Atau :

```
<jsp:nama_aksi atribut1 atribut2 .... />
   /*
   pada bagian ini bisa diisi dengan JSP Action yang lain
   */
</jsp:nama_aksi>
```

Berikut adalah Standard Action yang terdapat dan dikenali oleh JSP:

- param action (`jsp:param`)
- forward action (`jsp:forward`)
- include action (`jsp:include`)
- plugin action (`jsp:plugin`)
- use bean action (`jsp:useBean`)
- set property action (`jsp:setProperty`)
- get property action (`jsp:getProperty`)

`jsp:param`

Standar Action ini berfungsi untuk mendefinisikan suatu variabel dan nilainya. Sintak yang digunakan adalah :

```
<jsp:param name="nama_parameter" value="nilai_parameter" />
```

Atau :

```
<jsp:param name="nama_parameter" value="<%= nilai_parameter %>" />
```

jsp:forward

Fungsi dari Standar Action ini adalah untuk mengirimkan (mem-forward) suatu permintaan dari suatu halaman JSP ke halaman JSP yang lain, halaman HTML atau Servlet yang masih berada pada konteks aplikasi web tersebut. Misalnya pada server.xml dikonfigurasi suatu konteks aplikasi web sebagai berikut :

```
<context path="/jsp" docBase="C:\app\JSP\" debug="0" reloadable="true"/>
```

Maka halaman JSP pengirim dan HTML, JSP atau servlet penerima harus berada pada satu direktori yaitu C:\app\JSP.

Sintaknya :

```
<jsp:forward page="url_tujuan" />
```

Atau :

```
<jsp:forward page="<%= url_tujuan %>" />
```

Atau :

```
<jsp:foward page="url_tujuan">
   <jsp:param name="nama_parameter1" value="nilai_parameter"  />
   <jsp:param name="nama_parameter2" value="<%= nilai_parameter %>" />
</jsp:forward>
```

Atau :

```
<jsp:foward page="<%= url_tujuan %>">
   <jsp:param name="nama_parameter1" value="nilai_parameter"  />
   <jsp:param name="nama_parameter2" value="<%= nilai_parameter %>" />
</jsp:forward>
```

Contoh kasus :

forward01.jsp

```
<jsp:forward page="/jsp_param02.jsp">
  <jsp:param name="nama_user" value="Chocolove Mic" />
  <jsp:param name="email" value="chocolove_mic@yahoo.co.uk" />
</jsp:forward>
```

Sumber : -

forward02.jsp

```
<%
String nama=null;
String email=null;

nama = request.getParameter("nama_user");
email = request.getParameter("email");

out.println("Nama : "+nama);
out.prinln("<br>");
out.println("Email : "+email);
%>
```

Sumber : -

Dari contoh ini, halaman `forward01.jsp` akan mengirimkan user ke halaman `forward02.jsp` sambil mengirimkan nilai dari parameter dengan nama `nama_user` dan `email`. Pada halaman forward02 digunakan method `getParameter("nama_parameter")` untuk menangkap nilai-nilai tersebut.

jsp:include

Standard action berfungsi untuk menyisipkan halaman yang bersifat statik maupun dinamik. Sintaks dari tag ini adalah :

```
<jsp:include page="nama_file" flush="true" />
```

Atau :

```
<jsp:include page="url" flush="true">
   <jsp:param name="nama_parameter" value="nilai_parameter" />
   . . . . .
</jsp:include>
```

Contoh :

include.jsp

```
<html>
<head>
<title>JSP</title>
</head>

<body>
<%@ include file="/halaman.html" %><br>
<%@ include file="/halaman.jsp" %>
<p>
<jsp:include page="/halaman.html" flush="true" /><br>
<jsp:include page="/halaman.jsp" flush="true" />
</body>

</html>
```

Sumber : -

halaman.html

```
<a href="halaman.html"> Ini link ke halaman.html </a>
```

Sumber : -

halaman.jsp

```
<%@ page language="java" import="java.util.Date" %>
<%= new Date() %>
<br>
<%
for (int i=1; i<3; i++)
{
out.println("Tulisan ke-"+i+"<br>");
}
%>
```

Sumber : -

Pada gambar di bawah bisa dilihat hasilnya. Kalau dilihat sekilas tidak terlihat perbedaanya. Tapi pada prosesnya, waktu yang digunakan directive include untuk menampilkan file-file yang disisipkan lebih cepat dibandingkan dengan penggunaan standard action jsp:include. Hal ini

bisa dibuktikan dengan melihat bagian detik pada gambar di bawah.

Hasilnya :

jsp:useBean

Tag ini berfungsi untuk menggunakan JavaBean atau class Java. Sintaks yang digunakan adalah :

```
<jsp:useBean id="nama_identifier"
scope="page|request|application|session"
[ class="nama_class" | type="tipe" | class="nama_class" type="tipe"|
beanName="nama_bean|
<%= expression %>" type="tipe"]
/>
```

Atau :

```
<jsp:useBean id="nama_identifier"
scope="page|request|application|session"
[ class="nama_class" | type="tipe" | class="nama_class" type="tipe"|
beanName="nama_bean|
<%= expression %>" type="tipe"] >
 . . . . . .
</jsp:useBean>
```

Atribut yang dimiliki oleh tag ini adalah :

`id`	untuk memberikan nama untuk identitas objek yang dibuat.
`scope`	atribut ini mendefinisikan ruang batas (scope) dari objek yang dibuat. nilai yang dibolehkan adalah page, request, session dan application. nilai default dari scope adalah page
`class`	nama class yang akan digunakan untuk membuat objek.
`beanName`	nama class JavaBean yang akan digunakan
`type`	atribut ini digunakan untuk menentukan class lain yang merupakan class, super-class atau interface dari class yang ditentukan pada atribut class.

Berikut contoh berbagai beberapa cara penggunaannya :

```
<%-- membuat object dari class --%>
<jsp:useBean id="user" class="com.chocolove.kelas.User" scope="session" />

<%-- membuat objek dari JavaBean --%>
<jsp:useBean id="user" beanName="com.chocolove.bean.User" scope="session"
/>
```

Bisa dilihat id dari objek adalah "`user`" sehingga pada scriplet bisa ditulis kode sebagai berikut :

```
<%
   String namaUser = user.getName();
   out.println(namaUser);
%>
```

jsp:setProperty

Fungsi tag ini adalah untuk men-set nilai dari properti pada objek yang dibuat dengan tag `jsp:useBean`.
Sintaksnya adalah :

```
<jsp:setProperty name="id_objek"
  [ property="*"
  | property="nama_properti"
  | property="nama_properti" param="nama_parameter"
  | property="nama_properti" value="nilai_properti|<%= expression %>"]
/>
```

jsp:getProperty

tag ini berfungsi untuk mengambil dan menampilkan ke layar nilai suatu properti yang diinginkan pada objek yang dibuat dengan tag `java:useBean`. Sintaks yang digunakan adalah :

```
<jsp:getProperty name="id_objek" property="nama_property_pada_objek" />
```

Berikut adalah contoh penggunaan Standar Action :

- `jsp:useBean`
- `jsp:setProperty`
- `jsp:getProperty`

Berikut adalah file-file yang diperlukan :

jsp_bean.jsp

```
<html>
<head>
<title>JSP</title>
</head>

<body>
<jsp:useBean id="cacah" class="com.chocolove.CounterBean" scope="page">
  Nilai dari Action getProperty :
  <jsp:getProperty name="cacah" property="count" /><br>
  <jsp:setProperty name="cacah" property="count" value="8" />
</jsp:useBean>

<%
       out.print("Nilai awal : "+cacah.getCount());
```

```
        cacah.setCount(5);
        cacah.increaseCount();
        out.println("<br>");
        out.println("Nilai setelah dicacah :"+cacah.getCount());
%>
<br>
Nilai dari Action getProperty :
<jsp:getProperty name="cacah" property="count" /><br>
</body>
</html>
```

Sumber : -

CounterBean.java

```
/* paket dari class */
package com.chocolove;

public class CounterBean {
        int count = 3; /* nilai awal properti */

        /* method untuk mengambil nilai properti */
        public int getCount() {
                return count;
        }

        /* method untuk mengisi nilai properti */
        public void setCount(int c) {
                count = c;
        }

        /* method untuk menaikkan nilai properti */
        public int  increaseCount() {
                count++;
                return count;
        }
}
```

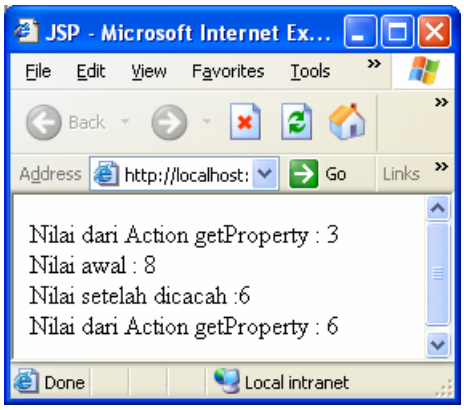
Sumber : -

class hasil kompilasi CounterBean.class akan disimpan pada direktori sesuai packagenya yaitu :

```
com
 |
 ---chocolove
          |
          --- CounterBean.class
```

Direktori com ini disimpan pada `WEB-INF\classes` pada direktori yang menjadi konteks.

Hasilnya dari contoh di atas adalah:



Keluaran pada baris pertama adalah hasil dari baris berikut :

```
Nilai dari Action getProperty :
  <jsp:getProperty name="cacah" property="count" />
```

Bisa dilihat kalau nilai awal properti count pada objek hasil dari class CountBean adalah 3.

Kemudian untuk keluaran baris kedua, prosesnya adalah nilai properti count pada objek diset menjadi 8 dengan baris berikut :

```
<jsp:setProperty name="cacah" property="count" value="8" />
```

Dan selanjutnya ditampilkan dengan scriplet dengan perintah berikut ini :

```
out.print("Nilai awal : "+cacah.getCount());
```

Keluaran baris ketiga adalah hasil baris berikut :

```
cacah.setCount(5);
cacah.increaseCount();
out.println("Nilai setelah dicacah :"+cacah.getCount());
```

Sedangkan keluaran pada bari terakhir adalah pengambilan nilai properti dengan tag jsp:getProperty seperti di bawah ini :

```
Nilai dari Action getProperty :
<jsp:getProperty name="cacah" property="count" />
```

Keterangan Artikel:
Isi artikel ini hanya merupakan pengantar untuk sebagai materi pengenalan tag-tag Standard Action pada JSP. Contoh-contoh akan dibahas pada bagian tersendiri.

Catatan Penulis :
Penulis adalah pemula dalam mempelajari Java, jadi mohon masukkannya bagi pembaca yang menemukan kesalahan konsep atau asumsi yang digunakan penulis.