

BAB 7

Pengenalan Arsitektur MVC

7.1 Pengenalan Arsitektur Model-View-Controller

Arsitektur Model-View-Controller adalah sebuah pola yang terbukti membangun proyek secara lebih efektif. Hal itu dilakukan dengan memilah komponen antara Model, View dan Controller pada bagian – bagian dalam proyek.

7.1.1 Motivasi

Aplikasi apapun, bagian dalam kode yang sering mengalami perubahan adalah bagian user interface. User interface adalah bagian yang paling terlihat oleh user dan bagaimana ia berinteraksi dengan aplikasi, membuatnya menjadi titik fokus perubahan berdasar kemudahan penggunaan.

Business-logic yang rumit pada user-interface membuat perubahan pada user interface menjadi lebih kompleks dan mudah terjadi kesalahan. Perubahan pada satu bagian memiliki potensi keterkaitan dengan keseluruhan aplikasi.

7.1.2 Solusi

Pola MVC menyediakan sebuah solusi terhadap permasalahan tersebut dengan membagi aplikasi menjadi bagian – bagian tersendiri, Model, View dan Controller, memisahkan antar bagian tersebut dan membuat tata interaksi diantaranya.

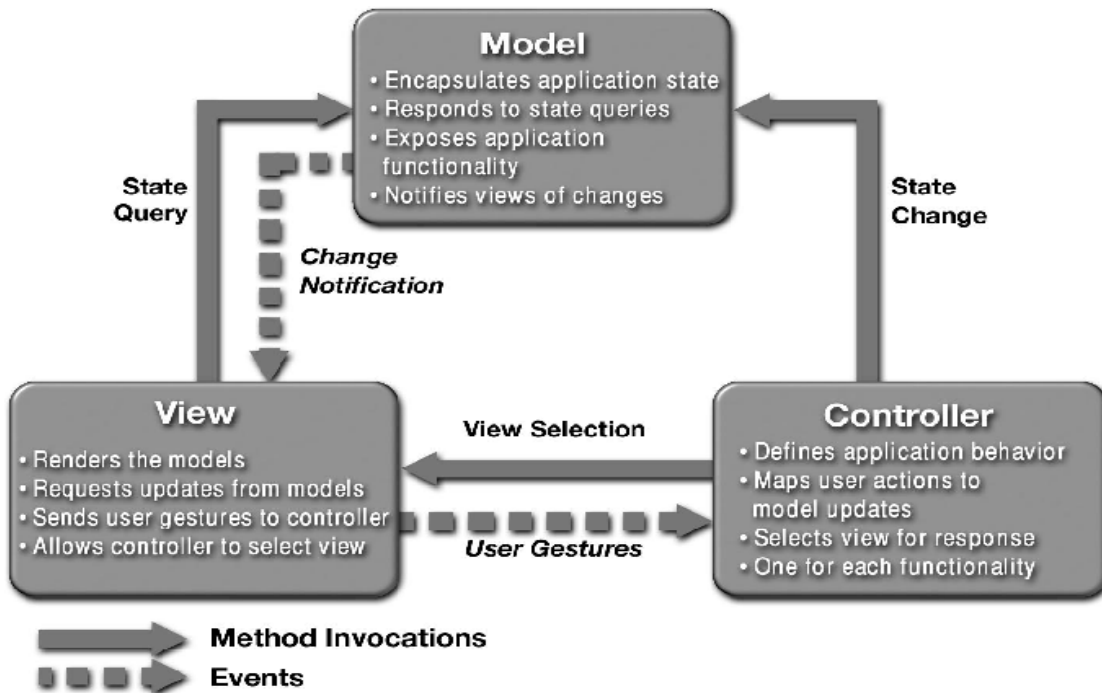


Diagram di atas menunjukkan 3 komponen yang terdapat dalam pola MVC dan interaksi yang terjadi.

7.2 MODEL

Pola MVC memiliki *layer* yang disebut dengan Model yang merepresentasikan data yang digunakan oleh aplikasi sebagaimana proses bisnis yang diasosiasikan terhadapnya. Dengan memisahkannya sebagai bagian terpisah, seperti penampungan data, persistence, serta proses manipulasi, terpisah dari bagian lain aplikasi.

Terdapat beberapa kelebihan dalam pendekatan ini. Pertama, membuat detail dari data dan operasinya dapat ditempatkan pada area yang ditentukan (Model) dibanding tersebar dalam keseluruhan lingkup aplikasi. Hal ini memberikan keuntungan dalam proses maintenance aplikasi.

Kedua, dengan pemisahan total antara data dengan implementasi interface, komponen model dapat digunakan kembali oleh aplikasi lain yang memiliki kegunaan yang hampir sama.

7.3 VIEW

Layer ini mengandung keseluruhan detail dari implementasi user interface. Disini, komponen grafis menyediakan representasi proses internal aplikasi dan menuntun alur interaksi user terhadap aplikasi. Tidak ada layer lain yang berinteraksi dengan user, hanya View.

Penggunaan layer View memiliki beberapa kelebihan : Pertama, memudahkan penggabungan divisi desain dalam development team. Divisi desain dapat berkonsentrasi pada style, look & feel, dan sebagainya, dalam aplikasi tanpa harus memperhatikan lebih pada detail yang lain.

Dan juga, memiliki layer View yang terpisah memungkinkan ketersediaan multiple interface dalam aplikasi. Jika inti dari aplikasi terletak pada bagian lain (dalam Model), multiple interfaces dapat dibuat (Swing, Web, Console), secara keseluruhan memiliki tampilan yang berbeda namun mengeksekusi komponen Model sesuai fungsionalitas yang diharapkan.

7.4 CONTROLLER

Terakhir, arsitektur MVC memiliki layer Controller. Layer ini menyediakan detail alur program dan transisi layer, dan juga bertanggungjawab akan penampungan events yang dibuat oleh user dari View dan melakukan update terhadap komponen Model menggunakan data yang dimasukkan oleh user.

Kelebihan dalam penggunaan layer Controller secara terpisah : Pertama, dengan menggunakan komponen terpisah untuk menampung detail dari transisi layer, komponen view dapat didesain tanpa harus memperhatikan bagian lain secara berlebih. Hal ini memudahkan team pengembang multiple interface bekerja secara terpisah dari yang lain secara simultan. Interaksi antar komponen View terabstraksi dalam Controller.

Kedua, dengan menggunakan layer terpisah yang melakukan update terhadap komponen Model, detail tersebut dihapus dari layer presentasi. Layer presentasi kembali pada fungsi utamanya untuk menampilkan data kepada user. Detail tentang bagaimana data dari user mengubah ketetapan aplikasi disembunyikan oleh Controller. Hal ini memisahkan dengan jelas antara presentation logic dengan business logic.

Tidak dapat disimpulkan bahwa pola MVC hadir dengan kelebihan – kelebihan tanpa ada efek samping. Pembagian aplikasi dalam 3 bagian terpisah meningkatkan kompleksivitas. Pada aplikasi kecil yang tidak membutuhkan loose coupling pada Model, hal ini dapat menjadi blok penghalang dalam penggunaan pola ini. Bagaimanapun, yang terbaik adalah untuk meyakini bahwa sebuah aplikasi umumnya dimulai dari aplikasi sederhana, dan berkembang menjadi sistem yang kompleks., sehingga factor loose coupling harus selalu diutamakan dan diperhatikan.

7.5 Arsitektur MVC Untuk Web : Arsitektur Model 2

Arsitektur MVC secara sederhana dirancang dan diadaptasi dalam penggunaan dalam Web-Application. Arsitektur yang dihasilkan kemudian disebut dengan *Model 2 Architecture*.

Aplikasi Model 2 umumnya memiliki :

- o Servlet Controller yang menyediakan akses tunggal terhadap keseluruhan aplikasi. Controller ini bertanggungjawab menyediakan manajemen terpusat terhadap alur aplikasi dan juga service lain seperti penanganan security dan user management.
- o Controller Servlet umumnya menggunakan konfigurasi XML untuk menentukan alur aplikasi dan pemrosesan perintah. Hal itu juga membuat helper components yang berfungsi sebagai Command objects. Hal ini berarti helper components terasosiasi dengan user actions dan dibuat/dipanggil untuk menangani actions yang terjadi, memanggil komponen Model sebagaimana diperlukan. Hal ini berfungsi untuk memisahkan untuk memisahkan antara controller servlet dari Model.

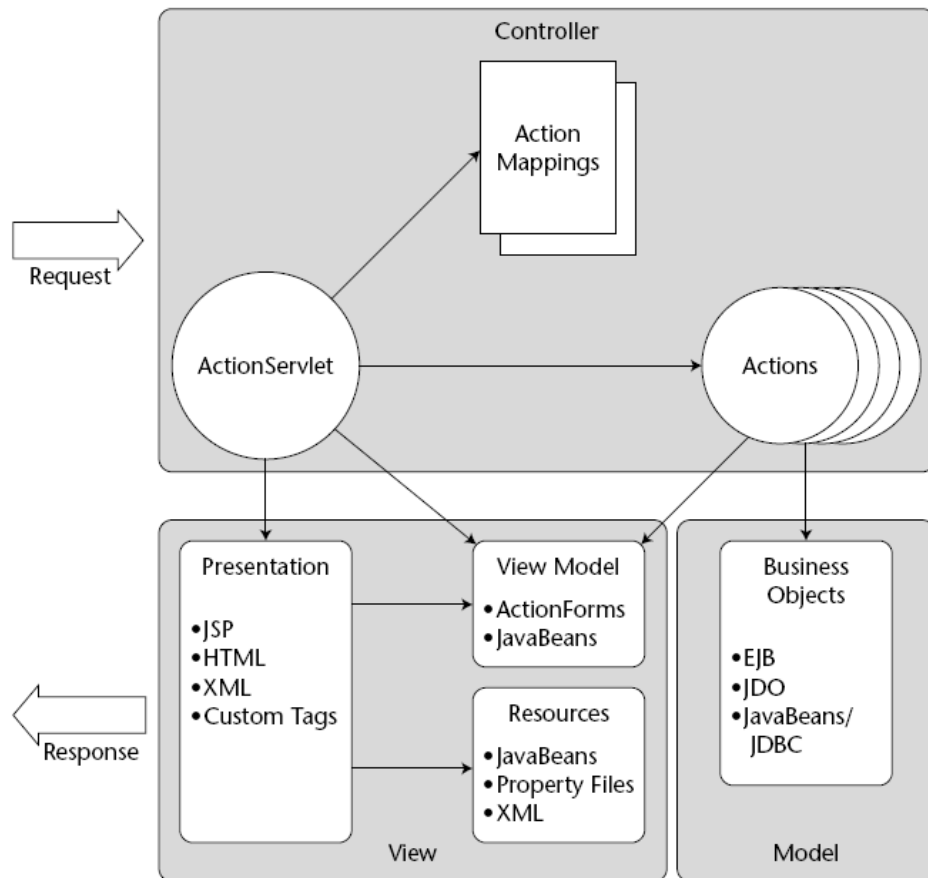
7.5.1 Model 2 Diagram

Implementasi sebuah pola dapat dipermudah dengan menggunakan third-party framework. Frameworks tersebut menyediakan detail terkait (request, konfigurasi, dan sebagainya) sehingga kita dapat berkonsentrasi pada hal lain yang lebih penting. Frameworks tersebut juga menyediakan fungsi - fungsi tambahan.

Pada pembelajaran ini, akan dibatasi pada dua framework yang populer, Struts dan JavaServerFaces (JSF). Struts akan dibahas pada bagian ini, sedangkan JSF pada bagian selanjutnya.

7.5.2 STRUTS

Struts adalah open-source framework yang dibuat oleh Apache Software Foundation. Dibawah ini menunjukkan bagaimana penanganan Model 2 Architecture oleh Struts.



Perhatikan object yang disediakan oleh framework pada komponen Model, View dan Controller.

7.5.3 CONTROLLER

7.5.3.1 ActionServlet

Pusat dari implementasi controller pada Struts adalah **ActionServlet**. Berfungsi sebagai Front Controller servlet dan menyediakan jalur tunggal untuk mengakses aplikasi. ActionServlet juga mengandung logic dalam penanganan request dari client – terlihat pada HTTP request dari client, kemudian diteruskan menuju sebuah halaman Web atau mengirimkan request tersebut object penerima yang disebut dengan **Actions** yang kemudian bertanggungjawab untuk menentukan respon yang akan dihasilkan.

ActionServlet memiliki detail – detail sebagai berikut, Action apa yang akan dipanggil untuk menangani request, komponen view mana yang akan dipanggil selanjutnya – dengan membaca konfigurasi XML, yang umunya bernama **struts-config.xml**.

Servlet ini menyediakan struts framework yang siap digunakan. Secara keseluruhan penting untuk menyertakannya pada aplikasi sebagai konfigurasi pada deployment aplikasi.

Berikut ini adalah potongan dari web.xml yang menunjukkan konfigurasi ActionServlet.

```
...
<servlet>
<servlet-name>action</servlet-name>
<servlet-class>
org.apache.struts.action.ActionServlet
</servlet-class>
<init-param>
<param-name>application</param-name>
<param-value>ApplicationResources</param-value>
</init-param>
<init-param>
<param-name>config</param-name>
<param-value>/WEB-INF/struts-config.xml</param-value>
</init-param>
</servlet>
...
<servlet-mapping>
<servlet-name>action</servlet-name>
<url-pattern>*.do</url-pattern>
</servlet-mapping>
```

7.5.3.2 Action

Seperti yang telah disebutkan sebelumnya, beberapa request dari client diteruskan menuju Action yang sesuai oleh front controller servlet. Seluruh action objects mendefinisikan sebuah method yang disebut `execute()`, dan method inilah yang dipanggil oleh `ActionServlet` untuk menangani request yang terjadi.

Aktifitas umum dalam aplikasi web adalah user log-in. Ditunjukkan di bawah ini adalah implementasi dari `LoginAction` class yang berfungsi untuk menangani request tersebut.

```
import org.apache.struts.action.*

public class LoginAction extends Action {
    public ActionForward execute(ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
        // casting ActionForm

        LoginForm loginForm = (LoginForm)form;

        // Menampung data dari user.
        String loginName = form.getLoginName();
        String password = form.getPassword();

        // Membuat business object yang akan menangani request
        UserService service = new UserService();
        user = service.login(loginName, password);

        // Jika user tidak ada, diteruskan pada error page
        if (user == null) {
            return mapping.findForward("failure");
        }

        // Simpan pada session untuk digunakan pada proses aplikasi selanjutnya
        HttpSession session = request.getSession();
        session.setAttribute(ApplicationConstants.USER_OBJECT, user);

        // User telah log-in dengan sukses.
        return mapping.findForward("success");
    }
}
```

Perhatikan bahwa implementasi di atas menggunakan sebuah business object yang disebut dengan **UserService**, untuk menentukan otentifikasi user dan tidak menyediakan implementasi secara langsung pada method eksekusinya. **Action** instances harus dibuat dengan alur sebagai berikut – fungsi utama harus di teruskan kembali pada business object (yang dapat ditemukan pada bagian Model), tidak

diimplementasikan dalam Action itu sendiri. Proses yang akan dijalankan pada Action adalah sebagai berikut :

- o Menampung informasi user dari ActionForm bean terkait
- o Menerjemahkan data dari form menjadi parameter yang diperlukan oleh business object yang mengimplementasikan fungsionalitasnya
- o Menampung hasil operasi dari business object dan menentukan View selanjutnya yang akan ditampilkan pada user
- o Secara opsional, menyimpan data hasil dari operasi bisnis ke dalam session atau request objects untuk digunakan dalam proses aplikasi selanjutnya

Hal lain yang perlu diperhatikan pada saat membuat kode instances dari Action objects adalah bahwa framework hanya akan menginstansiasi satu salinan object dan menggunakannya untuk memfasilitasi seluruh request. Hal ini berarti kita harus selalu membuat kode Action dalam thread-safe, dan memastikan bahwa hanya local variabel yang digunakan, bukan instance variabel.

Action instances berkemampuan untuk menginstruksikan ActionServlet dalam memilih komponen View mana yang akan digunakan dalam merespon dengan mengembalikan instances dari objek **ActionForward**. Actions memiliki akses terhadap ActionForward melalui penggunaan **ActionMapping** yang mengenkapsulasi data dari pemetaan logical path pada setiap Action. Pemetaan tersebut dibaca dari file konfigurasi oleh ActionServlet, yang kemudian bertanggungjawab untuk meneruskan ActionMapping pada Action yang sesuai. Sehingga menginstruksikan ActionServlet untuk meneruskan pada logical map dengan hasil "Success", Action bekerja sesuai pernyataan berikut :

```
return mapping.findForward("success");
```

7.5.3.3 ActionForm

Struts framework menyediakan sebuah class yang disebut dengan ActionForm. Instances dari class ini digunakan untuk memfasilitasi penampungan data yang berasal dari form yang dikumpulkan dari user pada Action instances yang menangani events pada form tersebut.

Tiap instances dari ActionForm merepresentasikan sebuah form atau rangkaian forms. Instances tersebut mendefinisikan properties dari elemen – elemen form, dan mempublikasikannya dengan menggunakan getters dan setters yang dapat diakses secara public. Action yang memerlukan data dari form kemudian memanggil method getter dari instance ActionForm.

Struts menyertakan definisi class dasar; developer bertanggungjawab penuh atas implementasi buatan mereka sendiri.

Kode dibawah menunjukkan ActionForm yang digunakan pada contoh sebelumnya :

```
import org.apache.struts.action.*;
public class LoginForm extends ActionForm {
    private String loginName;
    private String password;

    public String getLoginName() {
        return loginName;
    }

    public void setLoginName(String loginName) {
        this.loginName = loginName;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

Beberapa hal yang perlu diingat pada saat pengkodean ActionForm :

- o Mendefinisikan properties (dengan asosiasi terhadap method get dan set) untuk tiap elemen yang akan direpresentasikan pada form
- o Hindari penempatan business logic pada ActionForm. ActionForm ditujukan untuk mentransfer data antara komponen View dan Controller, dan bukan ditujukan untuk business logic.
- o Opsi lain, sertakan sebuah method untuk memvalidasi data sebelum diteruskan pada Action. Validasi akan dibahas lebih mendalam selanjutnya.

7.5.3.4 struts-config.xml

File ini berfungsi sebagai file konfigurasi komponen – komponen yang terdapat pada Struts. Disini, kita dapat mendefinisikan Action mana yang akan dipanggil untuk tiap request, form mana yang akan digunakan pada tiap Action, dan memetakan logical names pada actual path, diantara hal – hal lain. Berikut ini adalah potongan dari struts-config.xml yang digunakan pada contoh di atas :

```
<?xml version="1.0"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts
Configuration 1.1//EN" "http://jakarta.apache.org/struts/dtds/struts-
config_1_1.dtd" >
<struts-config>
  <form-beans>
    <form-bean name="loginForm" type="login.LoginForm"/>
  </form-beans>
  <action-mappings>
    <action name="loginForm"
      path="/login"
      scope="request"
      type="login.LoginAction">
      <forward name="success" path="/success.jsp"/>
      <forward name="failure" path="/failure.jsp"/>
    </action>
  </action-mappings>
</struts-config>
```

Mari kita bahas tiap elemen tersebut.

<!DOCTYPE ...>

Elemen ini mendefinisikan file XML sebagai file konfigurasi dalam penggunaan Struts framework. Terlewat dan kesalahan dalam penulisan kode, akan menghasilkan error pada saat aplikasi dijalankan.

<struts-config>

Elemen dasar dalam file konfigurasi. Seluruh elemen lain adalah elemen turunan dari elemen ini.

<form-beans>

Elemen ini menandai awal dan akhir definisi dari ActionForm instances. Elemen <form-bean> harus diposisikan sebagai turunan dari elemen ini.

<form-bean>

Menjelaskan instance dari ActionForm yang dapat digunakan oleh aplikasi. Elemen ini memiliki dua atribut :

- o name – nama logik yang akan diasosiasikan dengan ActionForm class
- o type – desripsi kualifikasi lengkap nama class dari ActionForm class

<action-mappings>

Elemen ini menandai awal dan akhir definisi Action dan pemetaannya. Seluruh elemen <action> harus diposisikan sebagai titik turunan dari elemen ini.

<action>

Menjelaskan instance dari Action object yang akan digunakan oleh aplikasi. Secara umum, elemen action akan mengimplementasikan atribut - atribut sebagai berikut :

- o path – context relative path yang digunakan oleh Action. Request apapun pada path ini menghasilkan Action dipanggil
- o type – deskripsi lengkap nama class dari Action class
- o name – nama elemen <form-bean> yang akan digunakan oleh Action
- o scope – lingkup dimana ActionForm yang dapat diakses. Hal ini mengatur dimana ActionServlet akan menyimpan instance dari ActionForm

<forward>

Elemen ini mendefinisikan pemetaan logik antara sebuah name dan path pada aplikasi.

Elemen ini memiliki atribut sebagai berikut :

- o name – nama dari elemen forward yang akan digunakan oleh Action instance
- o path – path dari komponen View yang akan diasosiasikan pada forward

Hal – hal yang perlu dilakukan pada Controller layer :

Untuk sekali pengaturan :

- o Mengkonfigurasi ActionServlet pada deployment aplikasi

Untuk tiap form handler yang akan ditambahkan pada aplikasi :

- o Membuat object ActionForm yang akan menrepresentasikan seluruh data yang dikumpulkan dari form
- o Membuat object Action dimana pada method eksekusinya menjelaskan bagaimana form akan ditangani
- o Mengkonfigurasi object ActionForm dalam struts-config.xml, pada bagian <form-beans>
- o Mengkonfigurasi object Action dalam struts-config.xml, pada bagian <action-mappings>
- o Mengkonfigurasi seluruh elemen forward yang akan digunakan oleh Action, dengan menempatkan <forward> pada badan definisi <action>

7.5.4 MODEL

Struts framework secara eksplisit tidak menyediakan komponen apapun dalam elemen Model. Object mana yang akan digunakan sebagai komponen Model ditentukan sepenuhnya oleh developer, umumnya adalah JavaBeans, atau terkadang EJB.

7.5.5 VIEW

Struts dapat menggunakan teknologi presentation layer apapun, walaupun pada sebagian kasus yang presentation layer yang digunakan adalah JSP dan HTML. Apa saja yang disediakan oleh Struts, adalah serangkaian dari tag libraries yang memungkinkan penggunaan fitur – fitur dari Struts dalam pengumpulan dan validasi form secara otomatis.

7.5.5.1 struts-html

Struts menyertakan sebuah tag-library, disebut dengan struts-html, yang hampir menyerupai keseluruhan fungsionalitas standar HTML, namun ditambahkan dengan beberapa fitur tambahan.

Tag-library ini sering digunakan pada saat membuat form yang akan dipakai oleh aplikasi. Perhatikan contoh dibawah ini :

```
<%@ taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<html>
  <head><title>Login Page</title></head>
  <body>
    <h1> Login Page </h1>
    <br/>
    <html:form action="/login">
      User Name : <html:text property="loginName"/> <br/>
      Password : <html:password property="password"/> <br/>
      <html:submit>
    </html:form>
```

Ini adalah form yang digunakan pada contoh sebelumnya. Perhatikan bagaimana elemen standar HTML seperti `<form>`, `<input type="text">`, `<input type="password">`, serta `<input type="submit">` digantikan oleh tag dari library struts-html. Mari kita bahas tags tersebut satu-persatu :

`<html:form>`

Tag `<html:form>` menerjemahkan sebuah HTML form. Tiap form tag terasosiasi dengan sebuah action mapping yang didefinisikan oleh atribut **action**. Value dalam atribut action menspesifikasikan path dari action mapping yang sesuai.

Perhatikan kembali file konfigurasi pada contoh sebelumnya :

```
...  
<action name="loginForm"  
    path="/login"  
    scope="request"  
    type="login.LoginAction">  
...  

```

Dapat kita lihat bahwa value dari atribut path terdefinisi pada file konfigurasi sesuai dengan value atribut action pada tag `<html:form>`. Hal ini mengindikasikan bahwa dalam form ini, saat diserahkan, akan diteruskan pada `LoginAction` untuk ditangani.

Salah satu dari persyaratan dari tag `html:form` yang valid adalah tiap field yang mengandungnya harus tercantum pada `ActionForm` untuk dipetakan pada action mapping.

Kemungkinan atribut lain :

- o `method` – dapat berupa POST atau GET. Menjelaskan method HTTP yang akan digunakan pada pengiriman form

`<html:text>`

Tag ini menerjemahkan standar HTML text input field. Property attribute menentukan batasan property dalam `ActionForm` terkait. Sebagai contoh, value dari property attribute dari contoh di atas adalah `loginName`, text field ini terhubung dengan property `loginName` pada `LoginForm`.

Atribut lain yang tersedia pada tag ini :

- o `size` – mendefinisikan ukuran dari text field yang akan ditampilkan
- o `maxlength` – menentukan panjang maksimum value yang dapat dimasukkan oleh user

`<html:password>`

Tag ini menerjemahkan standard HTML password field. Property attribute dari tag ini menjelaskan batasan property dalam `ActionForm` yang berkaitan. Dalam contoh diatas, field ini dibatasi oleh `password` property pada `LoginForm`.

Pada pembahasan sebelumnya tentang `html:text` dan `html:password` field, dijelaskan bahwa tag – tag tersebut dibatasi oleh properties pada `ActionForm` yang terhubung dengan form. Hal ini menjelaskan bahwa pada prakteknya value yang dimasukkan oleh user dalam field ini secara otomatis akan diatur sesuai batasan properties dalam object `ActionForm`. Terlebih lagi, jika terdapat value sebelumnya dalam `ActionForm` object (form tersebut telah diakses sebelumnya), field pada form secara otomatis pula akan terisi dengan values dari `ActionForm` object.

Tag-tag lain yang terdapat pada `struts-html` tag library :

`<html:hidden>`

Menerjemahkan HTML hidden pada form field.

Contoh penggunaan :

```
<html:hidden property="hiddenField"/>
```

`<html:radio>`

Menerjemahkan HTML radio check box.

Contoh penggunaan :

```
<html:radio property="radioField"/>
```

`<html:select>`, `<html:option>`

`html:select` digunakan untuk menerjemahkan sebuah drop-down list box. Pilihan – pilihan yang terdapat pada list box didefinisikan menggunakan tag `html:option`.

Contoh penggunaan :

```
<html:form action="/sampleFormAction">
  <html:select property="selectField" size="5">
    <html:option value="0 value">0 Label</html:option>
    <html:option value="1 value">1 Label</html:option>
    <html:option value="2 value">2 Label</html:option>
  </html:select>
</html:form>
```

Contoh di atas akan membuat sebuah list box dengan 5 item berukuran 5. Perhatikan bahwa body dari `html:option` tag berperan sebagai label dari item pada list, sedangkan value dari value attribute menentukan value yang akan diberikan pada Action.

`<html:checkbox>`, `<html:area>`

Digunakan untuk menerjemahkan check box field dan textarea.

Seperti yang terlihat pada pembahasan sebelumnya (Advanced JSP), terdapat beberapa hal yang perlu dilakukan sebelum memakai bermacam tag libraries dalam aplikasi.

Pertama, letakkan file JAR implementasi dari fungsionalitas tag dalam direktori WEB-INF/lib dalam aplikasi.

Beberapa hal yang harus dilakukan terhadap layer View :

Untuk satu kali pengaturan :

- Konfigurasi tag library yang akan digunakan dalam pengembangan aplikasi
- Tempatkan file JAR yang memiliki implementasi tag libraries pada direktori WEB-INF/lib dalam aplikasi

Untuk setiap form yang akan dibuat :

- Buat sebuah direktive taglib pada halaman JSP untuk menggunakan struts-html tag library
- Pada peletakan standar `<form>` tag, gunakan `<html:form>`, tentukan dalam action attribute dari path Action yang akan menangani form
- Pada posisi peletakan HTML field tags (`<input type="text">`, dan sebagainya), gunakan tags yang terdapat pada struts-html tag library yang memiliki fungsi yang sama (`<html:text>`, dan sebagainya).
 - Pastikan bahwa seluruh input fields yang terdapat pada form terdefinisi sebagai properties pada ActionForm object yang terhubung dengan request. Tidaklah perlu bahwa seluruh properties pada object ditampilkan sebagai field, namun dipersyaratkan bahwa seluruh fields juga terdefinisi sebagai properties.
 - Menutup tag `<html:form>`

7.6 MEMANDANG HAL-HAL YANG ADA SECARA KESELURUHAN

Untuk memahami bagaimana Struts framework berfungsi secara menyeluruh, perhatikan scenario yang terdapat pada contoh diatas : user logging.

Sebelum seorang user memasuki situs, ActionServlet mengambil konfigurasi file dan menelusuri detail yang ada. Sehingga, pada saat user mengakses form login, framework telah mengetahui ActionForm terkait yang akan menyimpan detail dan Action yang akan menanganinya.

Saat halaman login dijalankan, struts html tags yang telah digunakan berupaya untuk menerjemahkan HTML fields. Jika ActionForm untuk form ini tidak ada, halaman tersebut tidak akan ditampilkan. Jika terdapat fields dalam form yang melebihi properties pada ActionForm untuk menopangnya, halaman tersebut juga tidak akan ditampilkan. Jika ActionForm tersebut memang telah terdefinisi sebelumnya, tags – tags yang ada akan ditampilkan jika terdapat values sebelumnya yang tersimpan dalam ActionForm, dan juga form fields akan ditampilkan beserta data yang ada. Sebaliknya, form fields akan dibiarkan kosong dan user akan memperoleh tampilan form tanpa isian.

Pada saat form diserahkan, values pada form fields secara otomatis diatur ke dalam object ActionForm oleh Struts Framework. Object ini kemudian diteruskan menuju Action handler yang sesuai, selama ActionMapping object yang merefleksikan detail pemetaan tercantum pada file konfigurasi.

Action object memerankan penanganannya, kemudian memberi tahu ActionServlet kemana langkah selanjutnya dengan menentukan salah satu elemen forward yang terkonfigurasi pada pemetaan/mapping. ActionServlet kemudian meneruskan user menuju halaman yang ditentukan.