Dịch <http://www.techscore.com/tech/Java/ApacheJakarta/Struts/index/>

**Lời giới thiệu**

Trong loạt bài viết này, chúng ta sẽ đi nói về 1 framework J2EE được phát triển bởi Apache Software Foundation có tên là “Struts”.

Struts là 1 open source framework được sử dụng để xây dựng web application bằng cách sử dụng nền tảng kĩ thuật Java Servlet và JSP. Đối với những bạn chưa có hiểu biết về Java Servlet và JSP, hay là chưa từng có kinh nghiệm phát triển web application thì sẽ gặp khá nhiều khó khăn khi làm quen với Struts.

Tuy nhiên, sau khi đã làm quen được với Struts thì nó lại có thể giúp người phát triển giải quyết rất nhiều vấn đề trong việc phát triển web application thông qua việc sử dụng các kĩ thuật như tag library, actionForm hay validation,…　Nào, chúng ta hãy cùng nhau bắt tay vào chinh phục struts nào!

# Bài 1. ActionServlet và Action

## **1.1. Struts là gì ?**

StrutsはWebアプリケーションの開発フレームワークで

* Java Servlet, Java BeansなどJ2SE又はJ2EEで提供されている標準的なテクノロジーを採用している
* MVCモデル2を採用している
* ビューの部分は、JSP, Jakarta Velocity, XSLTなど様々なものの中から1つ選択できる

といった特徴を持っています。Struts自身は基本的にMVCのControlerの部分を提供するフレームワークで、プログラマは主にModelとViewの部分を開発する事になります。

開発においてはまずJakarta StrutsプロジェクトからStrutsのライブラリをダウンロードしてください。ここではバージョン1.1 Release Candidate1を基に説明していきます。またダウンロードした圧縮ファイルを解凍すると「lib」ディレクトリの中に

struts.jar

commons-beanutils.jar

commons-collections.jar

commons-digester.jar

commons-logging.jar

がありますので、それを開発するWebアプリケーションの「WEB-INF/lib」ディレクトリにコピーしてください。最初の「struts.jar」以外はJakarta Commonsプロジェクトのライブラリなのですが、Strutsで開発したアプリケーションを実行する上で最低限必要なものです。

また他にもたくさんのjarファイルが「Struts」の「lib」ディレクトリにはありますが、それぞれ拡張機能を利用する場合に必要になってきます。必要な場合には、その都度、説明します。

## **1.2. ActionServlet**

Strutsで開発されたWebアプリケーションでは、サーブレットクラスは唯一つで「org.apache.struts.action.ActionServlet」のみです。（Tilesを使用する場合は他のサーブレットクラスを利用します。Tilesについては後の節で説明します。）Webアプリケーションに対する全ての処理を、このActionServletが受け取ります。プログラマは「WEB-INF/web.xml」にActionServletを使用する設定を行うだけで、サーブレットクラスの実装は行いません。実際のサーバでの細かい処理はActionクラスで行われ、プログラマはこのクラスの実装を行う事によりWebアプリケーションを構築します。それに関する詳細は後述します。ここではActionServletの設定について説明します。

|  |
| --- |
| <web-app>  ...  <servlet>  <servlet-name>Action</servlet-name>  <servlet-class>org.<a href="#">apache</a>.struts.action.ActionServlet</servlet-class>  <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>  ...  </web-app> |

<servlet>タグでorg.apache.struts.ActionServletを使用する設定をします。その際の<servlet-name>に特に規定はありません。ActionServletは初期設定値として幾つか設定しなければならないものがありますが、以下に重要なものを示します。

|  |  |
| --- | --- |
| config | ActionServletの振る舞いを記述した設定ファイルをアプリケーションルートからの位置で指定します。実際の処理を担当するActionクラスとの関係や、Viewもこのファイルで指定します。デフォルト値は「WEB-INF/struts-config.xml」です。 |
| debug | デバッグ出力のレベルを指定します。0から6の数字で指定し、6の場合は全てのログが出力されます。0の場合は出力されません。デフォルト値は0です。 |

この内configが重要で、指定された設定ファイルの内容によって、Webアプリケーションの振る舞いが決定されます。今後「Struts設定ファイル」と記載した場合には、「ActionServletのconfigで指定された設定ファイル」の事だと考えるようにして下さい。

最後に<servlet-mapping>の設定をします。この設定によってどのURLでアクセスがあった場合にActionServletが呼び出されるか決定されます。サンプルでは拡張子が「.do」の場合に呼び出される設定をしていますが、Strutsではこの設定が推奨されています。

## **1.3. Thực thi Action class và chỉnh sử file cấu hình của Struts**

「WEB-INF/web.xml」の編集が終了したら、Struts設定ファイルの編集とActionクラスの実装に取り掛かります。

ActionServletは呼び出されると、その時のURLとStruts設定ファイルに基づいて実行するActionクラスを決定します。以下の単純な設定ファイルを用いて説明します。

|  |
| --- |
| <?xml version="1.0" encoding="ISO-8859-1" ?>  <!DOCTYPE struts-config PUBLIC  "-//<a href="#">Apache</a> Software Foundation//DTD Struts Configuration 1.1//EN"  "http://<a href="#">jakarta</a>.<a href="#">apache</a>.org/struts/dtds/struts-config\_1\_1.dtd">    <struts-config>  <action-mappings>  <action path="/sample/test" type="sample.TestAction">  <forward name="test" path="/sample/test.jsp" />  </action>  </action-mappings>  </struts-config> |

最初のxml宣言およびDOCTYPE宣言はそのまま真似て下さい。DTDはDOCTYPE宣言に記載されているURLにありますが、strutsのライブラリの中（libディレクトリ）にも含まれているのでそれを参考にして下さい。

<struts-config>内の<action-mapping>が、URLと呼び出されるActionクラスの設定を記述する部分です。具体的な設定は<action>タグで行われ「path」属性でURL、「type」属性で呼び出されるActionクラスが指定されます。例の場合、

|  |
| --- |
| http://（サーバ名）/（Webアプリケーションのルートpath）/sample/test.do |

というURLにアクセスがあると、「sample.TestAction」クラスがActionServletによって呼び出され、実行されます。

<action>タグの中にある<forward>タグは、リクエストをフォワードする先を指定するものです。Actionクラスは自身で表示を行わず、他にフォワードします。（Actionクラス自身で処理する事も可能ですが、それについては後に説明します。）その先の内容を指定するものが<forward>タグで、サンプルの場合は「/sample/test.jsp」ファイルを指定しています。「name」属性はフォワード先のキーを意味するもので、複数<forward>タグを指定した中から１つ転送先を選択する場合に使用します。

さて続いてActionクラスの実装について説明します。Actionクラスは「org.apache.struts.action.Action」クラスを継承する形で実装します。

|  |
| --- |
| package sample;    import org.<a href="#">apache</a>.struts.action.Action;  import org.<a href="#">apache</a>.struts.action.ActionMapping;  import org.<a href="#">apache</a>.struts.action.ActionForm;  import org.<a href="#">apache</a>.struts.action.ActionForward;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;    public class TestAction extends Action{  public ActionForward execute(ActionMapping mapping, ActionForm form,　HttpServletRequest request, HttpServletResponse response) throws Exception{  ....  return(mapping.findForward("test"));  }  } |

Actionクラスの実装は「execute」メソッドのオーバーライドが中心です。このメソッドはActionServletによってActionクラスが呼び出される際に実行されるもので、この中にクライアントからアクセスがあった場合の具体的な処理を記述します。引数・返り値の意味・利用目的は以下の通りです。

引数

|  |  |
| --- | --- |
| ActionMapping | ActionServletの情報を管理するクラス。主に<forward>タグと<exception>タグの情報を取得するために利用します。 |
| ActionForm | HTMLの<form>の情報を取得するために利用します。詳細については後の章で説明します。 |
| HttpServletRequest | 「ActionForm」で取得できない情報を取得するために使用します。基本的には使用しません。 |
| HttpServletResponse | forwardを行わずに、このクラス内でHTTPレスポンスを行う場合に使用します。基本的には使用しません。 |

返り値

|  |  |
| --- | --- |
| ActionForward | <forward>タグの情報を管理するクラス。executeメソッドがこのインスタンスを返す事により、<forward>タグで指定されたリソースにリクエストがフォワードされます。メソッドの返り値がnullである場合には、他への転送を行いません。 |

Actionクラスはexecuteメソッドの中で、ActionFormやHttpServletRequestからHTTPリクエストのパラメータを読み取り、何らかの処理を行います。処理が正常終了した場合には、ActionMappingのfindForwardメソッドでActionForwardインスタンスを１つ取得し、それを返します。findForwardメソッドの引数は文字列で、<forward>タグのname属性で指定した値を指定します。また返り値は、指定した値に該当する<forward>タグに該当するActionForwardインスタンスです。したがってfindForwardメソッドで検索された、<forward>タグのpath属性で指定されたリソースに、リクエストは転送される事になります。

executeメソッドの返り値をnullにした場合には他への転送が行われませんので、Actionクラス内で表示処理を行う必要があります。

処理が異常終了した場合には「javax.servlet.ServletException」を始め、何らかの例外をスローします。executeメソッドは「throws Exception」と定義されているので、任意の例外クラスをスローする事ができます。

（実習課題1）

以下のWebアプリケーションを、Strutsを用いて作成しなさい。

Internet Explorerでアクセスしてきた場合には403エラーを表示する事。

それ以外のブラウザでアクセスしてきた場合には時刻を表示する事。

時刻の表示はフォワード先のJSPページで行う事。

解答例

▼ディレクトリ構成は以下の通り

|  |
| --- |
| .  ├─com  │ └─techscore  │ └─struts  │ └─chapter1  │ └─exercise1　　　　　time.jsp  └─WEB-INF　　　　　　　　　　　　　　web.xml,struts-config.xml  ├─classes  │ └─com  │ └─techscore  │ └─struts  │ └─chapter1  │ └─exercise1　BrowserCheckAction.class  └─lib　　　　　　　　　　　　　　strutsライブラリjarファイル  ※strutsライブラリjarファイル  struts.jar,  commons-beanutils.jar,  commons-collections.jar,  commons-digester.jar,  commons-logging.jar |

▼BrowserCheckAction

|  |
| --- |
| /\*\*  \* BrowserCheckAction.java  \* TECHSCORE Java JakartaStruts 1章 実習課題1  \*  \* Copyright (c) 2004 Four-Dimensional Data, Inc.  \*  \*/    package com.techscore.struts.chapter1.exercise1;    import org.apache.struts.action.Action;  import org.apache.struts.action.ActionMapping;  import org.apache.struts.action.ActionForm;  import org.apache.struts.action.ActionForward;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;    public class BrowserCheckAction extends Action{  public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response) throws Exception{  //ブラウザの種類で処理を分岐　MSIEならエラーとする  String value = request.getHeader("user-agent");  if (value.indexOf("MSIE") >= 0) {  response.sendError(HttpServletResponse.SC\_FORBIDDEN);  return null;  } else {  return mapping.findForward("time");  }  }  } |

▼time.jsp

|  |
| --- |
| <!-- time.jsp -->  <!-- TECHSCORE Java JakartaStruts 1章 実習課題1 -->  <!-- Copyright (c) 2004 Four-Dimensional Data, Inc. -->    <%@ page contentType="text/html; charset=Windows-31J"  session="false"  pageEncoding="Windows-31J" %>  <html>  <head>  <title>TECHSCORE Java JakartaStruts 1章 実習課題1</title>  </head>  <body>  <%=new java.util.Date()%>  </body>  </html> |

▼struts-config.xml

|  |
| --- |
| <?xml version="1.0" encoding="Windows-31J" ?>  <!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>  <action-mappings>  <action path="/com/techscore/struts/chapter1/exercise1/BrowserCheck" type="com.techscore.struts.chapter1.exercise1.BrowserCheckAction">  <forward name="time" path="/com/techscore/struts/chapter1/exercise1/time.jsp" />  </action>  </action-mappings>  </struts-config> |

▼web.xml

|  |
| --- |
| <?xml version="1.0" encoding="Windows-31J" ?>  <!DOCTYPE web-app PUBLIC  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"  "http://java.sun.com/j2ee/dtds/web-app\_2\_3.dtd">    <web-app>  <servlet>  <servlet-name>Action</servlet-name>  <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>  <init-param>  <param-name>config</param-name>  <param-value>/WEB-INF/struts-config.xml</param-value>  </init-param>  <init-param>  <param-name>debug</param-name>  <param-value>6</param-value>  </init-param>  </servlet>    <servlet-mapping>  <servlet-name>Action</servlet-name>  <url-pattern>\*.do</url-pattern>  </servlet-mapping>    </web-app> |

▼起動URLは以下の通り

|  |
| --- |
| WEB\_ROOT/com/techscore/struts/chapter1/exercise1/BrowserCheck.do |

（実習課題2）

以下のWebアプリケーションを、Strutsを用いて作成しなさい。

パラメータ「op」の値が「time」である場合には、時刻を表示する事。

パラメータ「op」の値が「ip」である場合には、接続元のIPアドレスを表示する事。

いずれも表示はフォワード先のJSPページで行う事。

解答例

|  |
| --- |
| .  ├─com  │ └─techscore  │ └─struts  │ └─chapter1  │ └─exercise2　　time.jsp,ip.jsp  └─WEB-INF　　　　　　　　　　　web.xml（実習課題1とじ）,struts-config.xml  ├─classes  │ └─com  │ └─techscore  │ └─struts  │ └─chapter1  │ └─exercise2　ParameterCheckAction.class  └─lib　strutsライブラリjarファイル  ※strutsライブラリjarファイル  　　struts.jar,commons-beanutils.jar,commons-collections.jar,commons-digester.jar,commons-logging.jar |

# Bài 2. ActionForm

## 2.1. Thực thi ActionForm

## 2.2. Cấu hình ActionForm

## 2.3. Sử dụng lớp Action

## 2.4. Cách tiếp nhận giá trị parameter

## 2.5. Hàm reset của ActionForm

# Bài 3. Bean Tag Library (P1)

## 3.1. Bean Tag Library là gì ?

## 3.2. Tag define

## 3.3. Tag parameter / Tag cookie / Tag header

# Bài 4. Bean Tag Library (P2)

## 4.1. Tag message

## 4.2. Tag write

## 4.3. Tag include

# Bài 5. HTML Tag Library(P1)

## 5.1. Hàm validate của ActionForm Bean

## 5.2. Hiển thị Error Message

## 5.3. HTML Tag Library

## 5.4. form Tag / submit Tab / cancel Tag / reset Tag

# Bài 6. HTML Tag Library(P2)

## 6.1. text Tag／password Tag／textarea Tag

## 6.2. imageTag

## 6.3. radioTag／hiddenTag

## 6.4. checkboxTag／multiboxTag

# Bài 7. HTML Tag Library(P3)

## 7.1. selectTag／optionTag／optionsTag／optionsCollectionTag

## 7.2. htmlTag

## 7.3. linkTag／frameTag／imgTag

# Bài 8. Logic Tag Library (P1)

## 8.1. Logic Tag Lbrary

## 8.2. iterateTag

## 8.3. equalTag／notEqualTag／greaterEqualTag／greaterThanTag／lessEqual

## Tag／lessThanTag

# Bài 9. LogicTag Lbrary (P2)

## 9.1. matchTag／notMatchTag

## 9.2. emptyTag／notEmptyTag

## 9.3. presentTag／notPresentTag

## 9.4. messagesPresentTag／messagesNotPresentTag

# Bài 10. NestedTag Library

## 10.1. NestedTag Library

10.2. rootTag

10.3. nestTag

Bài 11. DynaActionForm

11.1. org.apache.struts.action.DynaActionForm

11.2. Kế thừa DynaActionForm

Bài 12. ValidatorForm／ValidatorActionForm

12.1. org.apache.struts.validaton Package

12.2. org.apache.struts.validator.ValidatorForm

12.3. Chỉnh sửa validation.xml

12.4. org.apache.struts.validator.ValidatorActionForm

Bài 13. validatorで提供されている検証ルール

13.1. 提供されている検証ルール

13.2. byte, short, integer, long, float, double, intRange, floatRangeルール

13.3. minlength, maxlengthルール

13.4. mask, date, creditCard, emailルール

13.5. required, requiredifルール

Bài 14. validation.xmlの設定

14.1. 定数

14.2. 変数

14.3. デフォルトのエラーメッセージ

14.4. indexedListProperty

Bài 15. Javascriptでの検証／検証ルールの追加

15.1. Javascriptでのチェック

15.2. 独自の検証ルールの追加

15.3. 独自の検証ルールに対するJavascriptの作成

Bài 16. DynaValidatorForm／DynaValidatorActionForm

16.1. DynaValidatorForm

16.2. DynaValidatorActionForm

16.3. プロパティ値のコピー

Bài 17. その他のActionクラス

17.1. ForwardAction, IncludeAction

17.2. DispatchAction

17.3. LookupDispatchAction

17.4. アプリケーションのモジュール化

Bài 18. Tiles

18.1. org.apache.struts.tiles

18.2. Tiles設定ファイルの作成

18.3. Tilesで定義したページの表示

Bài 19. レイアウト

19.1. Tileの挿入

19.2. Tileの置き換え

19.3. 組み合わせの継承

Bài 20. メニューの作成

20.1. メニューの作成

20.2. メニューの改良

20.3. 文字列・Tileのリスト

Bài 21. 表示するTileの動的な変更

21.1. org.apache.struts.tiles.Controller

21.2. org.apache.struts.tiles.actions.TilesAction

21.3. 表示するTileの動的な変更

Bài 22. TilesTagライブラリ

22.1. useAttributeTag／importAttributeTag

22.2. getAsStringTag／insertTag／getTag

Bài 23. TilesTagライブラリ2

23.1. definition Tag / put Tag

23.2. putList Tag / add Tag

Bài 24. Struts 1.2.x

24.1. ワイルドカード・マッピング

24.2. MappingDispatchAction

24.3. stopOnFirstError

24.4. キャンセル・ハンドラ (DispatchAction#cancelled)

24.5. validator の変更

24.6. ActionRedirect

24.7. DownloadAction

24.8. セッション・スコープの ActionMessages

24.9. エラーのハイライト

24.10. その他の追加/変更

実習課題

Apache / Jakarta TOP

Java TOP