東京ITスクール
TOKYO IT SCHOOL

Action と設定ファイル

目次

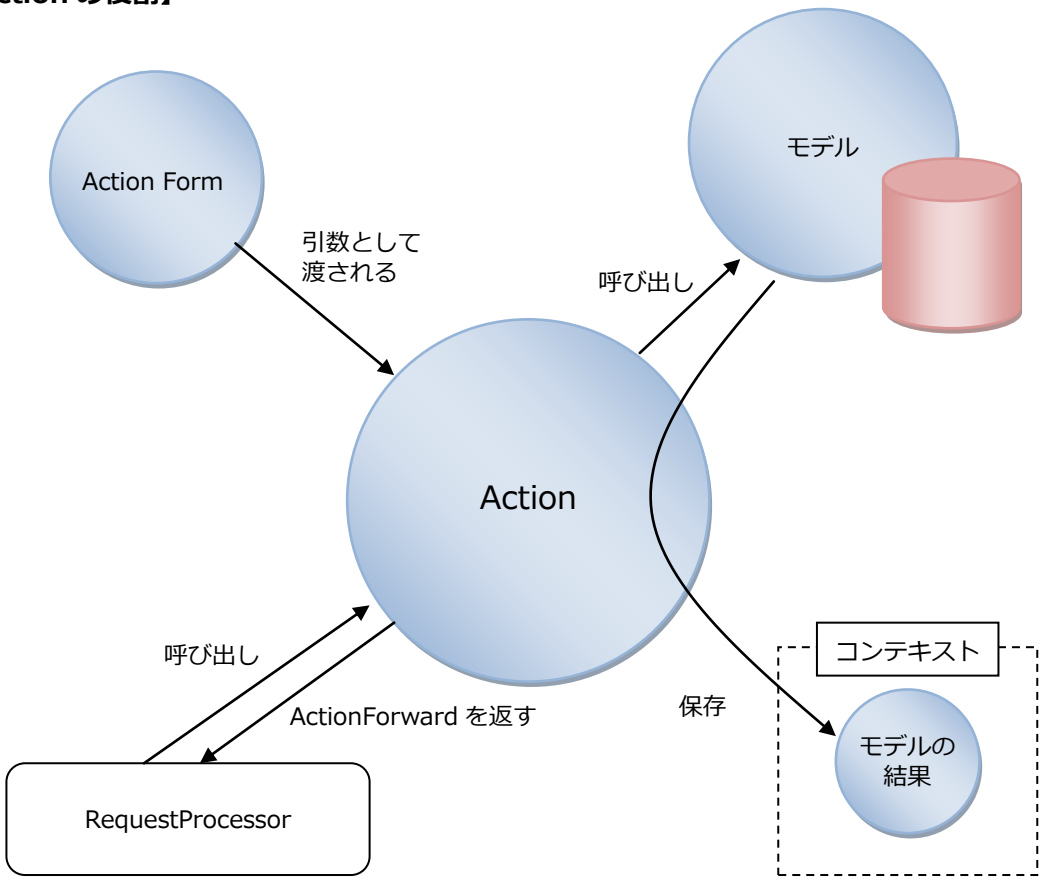
1. Action

1　Action の役割

Action の役割はモデルの呼び出しです。HTTP リクエストに従ってモデルを呼び出し、終了後はモデルにより生成された結果をコンテキストに保存します。

また、Action を作成する際は、org.apache.struts.action.Action を継承する必要があります。

【Action の役割】



Struts では ActionServlet が唯一のサーブレットですが、Action は ActionServlet から RequestProcessor 経由で処理を引き継ぐため、サーブレットでできることは Action でもほぼ実現可能です。

2　Action の execute()メソッド

前述の通り、Action は org.apache.struts.action.Action を継承したクラスを作ります。それに当たり、execute()メソッドをオーバーライドする必要もあります。このメソッドに、ビジネスロジックの呼び出しを実装し、フォーワード先を指定するため、Action の作成と execute()メソッドの作成はほぼ同義であると考えてお差し支えありません。

以下に execute()メソッドのシグニチャを示します。

```
public ActionForward execute(ActionMapping mapping,
                             ActionForm form,
                             HttpServletRequest request,
                             HttpServletResponse response)
    throws Exception
```

execute()メソッドの引数と戻り値を以下に示します。

execute()メソッドの引数

クラス名	説明
ActionMapping	org.apache.struts.action パッケージのクラスで、Action に関する設定を管理します。
ActionForm	org.apache.struts.action パッケージのクラスで、リクエストパラメータの値を管理します。
HttpServletRequest	javax.servlet.http パッケージのクラスで、ユーザーからのリクエストを表します。
HttpServletResponse	javax.servlet.http パッケージのクラスで、ユーザーへのレスポンスを表します。

execute()メソッドの戻り値

クラス名	説明
ActionForward	org.apache.struts.action パッケージクラスで、遷移先に関する設定を管理します。

3　ActionMapping と ActionForward

(1) ActionMapping

ActionMapping は、Struts 設定ファイルに記述された内容を管理するクラスです。開発者が作成するクラスではありません。

ActionMapping はリクエストされた URL と Action の関連付けを担当し、RequestProcessor においてリクエストの URL から Action を呼び出すときに利用されます。どの Action を実行するのかを識別するものと考えてもよいでしょう。

また、Action の execute()メソッドの引数にも ActionMapping がありますので、このメソッドの中で ActionMapping が持つデータを取り出し、操作することが可能です。

ActionMapping のインスタンスは、Struts 設定ファイルの action 要素の情報から自動生成されます。

以下のような Struts 設定ファイルを例に ActionMapping のインスタンスと Struts 設定ファイルの関連を確認しましょう。

struts 設定ファイル

```
<action-mappings>
    <action path="/actionA"                                    ①
            name="sampleForm"
            type="sample.SampleActionA">
        <forward name="forward_A" path="/WEB-INF/a.jsp"/>
        <forward name="forward_B" path="/WEB-INF/b.jsp"/>
    </action>
    <action path="/actionB"                                    ②
            type="sample.SampleActionB">
        <forward name="forward_C" path="/WEB-INF/c.jsp"/>
        <forward name="forward_D" path="/WEB-INF/d.jsp"/>
    </action>
</action-mappings>
```

ActionMapping のインスタンスは action-mappings 要素の中にある、action 要素数の分だけ作成されます。action 要素に必須の属性は type のみで、ここにはこの ActionMapping から呼び出される Action のクラス名を指定します。

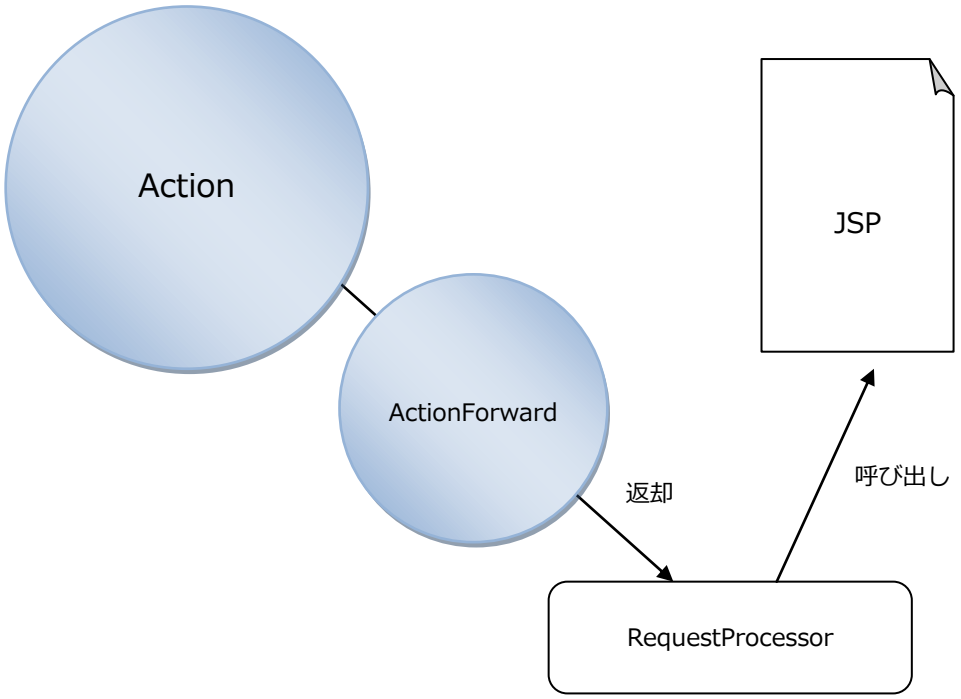
ActionMapping のもう 1 つの重要な役割として、ActionForm と Action の関連付けがあります。ActionForm は HTTP リクエストパラメータの値を保持する JavaBean ですが、これを Action に引き渡す場合は、action 要素の name 属性に ActionForm の名前を指定します。

(2) ActionForward

ActionForward は、ActionMapping と同様に Struts 設定ファイルに記述された内容を管理するクラスで、開発者が作成するクラスではありません。

ActionForward は、Action 終了後の遷移先を保持するクラスです。ActionForward のインスタンスは Struts 設定ファイルの記述に従って自動生成され、最終的にどこに遷移するかは Action 内に記述します。

【ActionForward の役割】



Struts 設定ファイルにおいて ActionForward を設定するのは、global-forwards 要素と action 要素にネストされる forward 要素です。

global-forwards 要素で指定される ActionForward は複数の Action から参照され、これをグローバルフォーワードと呼びます。global-forwards 要素に関しては、後述の Struts 設定ファイルの説明でも改めて触れます。

一方、forward 要素で指定される ActionForward は、親の action 要素で呼び出される Action からのみ参照され、これをローカルフォーワードと呼びます。ActionMapping の説明の際に使用した例の場合、①の action 要素は 2 つの forward 要素を持っていますので、SampleActionA という Action は、終了後に遷移する先の候補として、forward_A と forward_B の 2 つがあるということになります。forward 要素、global-forwards 要素では、name と type の 2 つ属性は必須で、name 属性ではこの ActionForward の論理名（Action の中で使用される）を指定し、type 属性で遷移先のリソースを指定しています。①の例ではどちらも JSP が遷移先となっていますが、これは別の Action である場合もあります。

(3) ActionMapping のメソッドと Action

ActionMapping は org.apache.struts.config.ActionConfig のサブクラスで、Action Config で用意されているメソッドを利用することにより、action 要素の各属性値を取得することができます。具体的には path, type, input, name, scope, validate などの属性値を、getPath() getType() getInput() getName() getScope() getValidate()などのメソッドで取得することができます。

また action 要素にネストされている forward 要素を ActionForward のインスタンスとして返すメソッドもあります。以下に Action 内でよく利用される、ActionMapping のメソッドを示します。

ActionMapping のメソッド

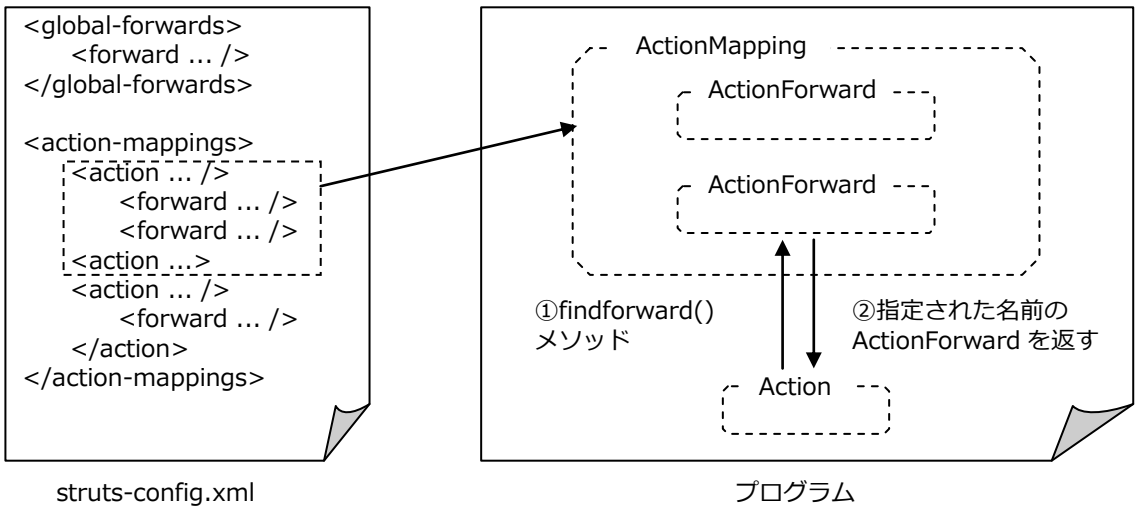
メソッド	説明
findForward(String name)	引数で指定した文字列を name 属性に持つ forward 要素の情報を、ActionForward インスタンスとして返します。
getInputForward()	action 要素の input 属性で指定されるリソースへの ActionForward インスタンスを返します。

上記のメソッドは、どちらも戻り値は ActionForward となりますので、Action で execute() メソッドを実装する際の戻り値として設定することも多々あります。

findForward()メソッドは、まず action 要素にネストされた forward 要素の name 属性を調べ、引数と一致するものがあれば、それを ActionForward として返します。見つからない場合には、global-forwards 要素にネストされた forward 要素の name 属性を調べます。そして引数と一致するものがあれば、その forward 要素を ActionForward として返します。action 要素、global-forwards 要素のいずれからも適した forward 要素が見つからない場合、null が返されます。

このように ActionMapping と ActionForward は、Struts 設定ファイルに書かれた情報をラッピングして保持し、Action などでの利用を容易にしています。

【ActionMapping と ActionForward】



4　Action の実装

Action クラスの仕組みをある程度理解したところで、ここでは、実際の Action の実装について考えていきます。冒頭で Action の役割はビジネスロジックの呼び出しであると述べましたが、実際には以下のような処理を行います。

- ActionForm を使用可能なように型変換を行う
- ビジネスロジックを呼び出す
- ビジネスロジックの処理結果をコンテキストに登録
- 遷移先の ActionForward を RequestProcessor に返す

「Struts の概要と動作」で作成したサンプル Action で考えてみましょう。execute()メソッドを見てみてください。

execute()メソッド

```
public ActionForward execute(ActionMapping mapping,
                             ActionForm form,
                             HttpServletRequest request,
                             HttpServletResponse response)
        throws Exception {

    // ActionForm を LoginForm にキャスト
    LoginForm loginForm = (LoginForm) form;

    //フォームに入力された ID を取得する
    String id =loginForm.getId();

    //フォームに入力されたパスワードを取得する
    String password =loginForm.getPassword();
    String message = " ID : " + id + "   パスワード : " + password;

    // message を request スコープに登録する
    request.setAttribute("message", message);

    // ActionForward を返却
    return mapping.findForward("success");
}
```

また、この Action を呼び出すための Struts 設定ファイル内の定義は、以下のようになっています。

```
<action path="/sample01/login"
        type="sample01.LoginAction"
        name="sample01_loginForm"
        scope="request">
    <forward name="success" path="/WEB-INF/jsp/sample01/success.jsp"/>
</action>
```

このメソッドでは、まず ActionForm を LoginForm にキャストしています。ActionForm のインスタンスについては「ActionForm」の資料にて触れますので、ここでは説明を割愛します。

次に loginForm から id, password の値を取得し、それらを連結して message という新しい String を作成しています。そして message を request スコープに登録しています。この例の場合は、この Action の中で処理が完結していますが、本来であれば、ここにビジネスロジックを呼び出すための処理、たとえばデータベースに問い合わせをして、結果セットをコンテキストに登録するなどを記述します。

最後に、ActionMapping の findForward メソッドを使用して、ActionForward インスタンスを返しています。findForward()メソッドは、Struts 設定ファイルに記述された forward 要素から、name 属性の値が引数に指定された文字列と同じものを探します。同じものがあれば、forward 要素の情報を ActionForward として返します。上記の場合、findForward()メソッドの引数には success という文字列が指定されていますので、name 属性が success である forward 要素の情報が返されます。これにより、execute()メソッドの処理が終わるとき、「/WEB-INF/jsp/sample01/success.jsp」へ処理がフォーワードされます。

5　Action 設計時のガイドライン

Action の設計について、本家の Web サイトでは「Action Class Design Guidelines」が示されています。以下はそちらを要約したものとなります。

(1) スレッドセーフを意識する

RequestProcessor により生成される Action のインスタンスは、各クラスに 1 つです。つまり、Web アプリケーション利用者からのリクエストがあるたびに、新しいインスタンスが生成されることはありません。

Action にインスタンス変数を定義し、それを利用してしまうと、複数のスレッドからアクセスされることになります。その結果、あるスレッドでセットした値が、別のスレッドにより読み込まれる、書き換えられるといったことが起こります。インスタンス変数は極力使用せずに、ローカル 変数で処理を行うようにしましょう。

(2) 例外は Action でキャッチする

Action の execute()メソッドではビジネスロジックを呼び出すため、様々な例外が発生する可能性があります。

execute()メソッドは、あらゆる例外をスローすることができるのですが、例外は極力スローせずに execute()メソッド内でキャッチするようにしましょう。

(3) Action は小さく、シンプルにする

Action の execute()メソッドでは、遷移先を決定するまでの部分、つまり ActionForward を返すまでの部分をできるだけシンプルに、短く書くことが推奨されています。

MVC モデルのコントローラは、ビジネスロジックはモデルに任せ、どのロジックを呼び出すかを制御するだけにするのが望ましいと言われています。同様に Action も、どのクラスを呼び出すか制御するだけで、複雑な処理は行わない方が良いとされています。

2. Struts 設定ファイル

1 Struts 設定ファイルとは

Struts 設定ファイルは、Struts を利用したアプリケーションの動作を決定する非常に重要なファイルです。Struts 設定ファイルは、ActionServlet が初期化されるタイミングと、ActionServlet から RequestProcessor が起動されるタイミングに読み込まれ、その設定内容に応じてモジュールの準備が整えられます。

Struts 設定ファイルの配置場所は、web.xml に記述します。一般的には、/WEB-INF/ディレクトリに配置し、その名称を struts-config.xml とします。

また、Struts 設定ファイルに設定できる情報は、およそ以下のとおりです。

- URL と起動する Action の関連情報
- Action 実行後に使用されるリソースの情報
- ActionForm の情報
- メッセージリソースの情報

Struts 設定ファイルは XML 形式です。まずはその構造について確認していきましょう。次の Struts 設定ファイルの例を見てください。

struts-config.xml の例

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.2//EN"
    "http://struts.apache.org/dtds/struts-config_1_2.dtd">

<struts-config>
    <action-mappings >
        <action path="/sample01/sample" type="sample01.SampleAction">
            <forward name="success" path="/jsp/sample.jsp" />
        </action>
    </action-mappings>
</struts-config>
```

まず、Struts 設定ファイルは XML 形式ですので、XML 宣言を最初に記述します。続けて DOCTYPE 宣言を記述します。DOCTYPE 宣言では、この設定ファイルの DTD を指定しています。実際にこの URL から得られる DTD を確認して設定方法を知ることもできますが、Struts のライブラリの中にも含まれているので、そちらを参考にしていただくこともできます。

Struts 設定ファイルのルート要素は、struts-config 要素です。この中にモジュールの動作を制御するための、さまざまな設定を行います。

2　主要な要素について

Struts 設定ファイルのルート要素 struts-config に記述する主要な要素は以下の通りです。

- form-beans 要素
- global-exceptions 要素
- global-forwards 要素
- action-mappings 要秦
- controller 要素
- message-resources 要素
- plug-in 要秦

(1) form-beans 要素

アプリケーションで使用する ActionForm に関する情報を記述します。この要素に重要な属性はありませんが、form-bean 要素を子要素として持ちます。form-bean 要素は、この struts アプリケーションで使用する ActionForm のクラス名とインスタンス名を関連付ける重要な要素です。

form-bean 要素の属性

属性名	必須	説明
name	○	ActionForm のインスタンスを識別するための名前を指定します。
type	○	ActionForm のクラス名を完全限定名で指定します。

form-bean 要素は、form-property 要素を子要素として持つ場合があります。これは、DynaActionForm と呼ばれる ActionForm か、そのサブクラスを使用した場合に必要となるものです。form-property 要素の属性を以下に示します。

form-property 要素の属性

属性名	必須	説明
initial		この要素で指定するプロパティの初期値を指定します。
name	○	この要素で指定するプロパティの名称を指定します。
size		このプロパティが配列の場合、この属性の値の長さで配列を初期化します。
type	○	この要素で指定するプロパティの型を指定します。

form-beans 要素の記述例

```
<form-beans>
    <form-bean name="sampleForm" type="sample.SampleForm"/>
    <form-bean name="dynaForm"
                type="org.apache.struts.action.DynaActionForm">
        <form-property name="id" type="java.lang.String" />
    </form-bean>
</form-beans>
```

(2) global-exceptions 要素

global-exceptions 要素では、Action により例外がスローされたときの振る舞いについての設定を行います。global-exceptions 要素は、exception 要素を子要素として持ち、詳細な設定は exception 要素で行います。

exception 要素の属性

属性名	必須	説明
path		例外発生時のフォーワード先リソースを、アプリケーションルートからの相対パスで指定します。
key	○	例外発生時にエラーメッセージを検索するキーの値を指定します。対応する値がメッセージリソースに存在する必要があります。
type	○	ハンドリングする例外クラスの完全限定名を指定します。
handler		例外ハンドラクラスの完全限定名を指定します。

global-exception 要素の使用例

```
<global-exceptions><exception type="java.io.IOException"
        key="error.io"
        path="/WEB-INF/jsp/ioError.jsp" />
</global-exceptions>
```

(3) global-forwards 要素

グローバルフォーワードに関する情報を設定します。すべての Action から使用される ActionForward をこの要素で設定します。forward 要素を子要素として持ちます。

forward 要素の属性

属性名	必須	説明
module		フォワードするモジュール名を指定します。モジュール名はスラッシュから始まります。
name	○	ActionForward を識別するための名前です。
path	○	フォーワードする JSP などのリソースをアプリケーションルートからの相対パスで指定します。
redirect		リダイレクトかフォーワードかを設定します。true ならリダイレクト、false ならフォーワード。false が初期値。

global-forwards 要素の使用例

```
<global-forwards>
    <forward name="success"
             path="/WEB-INF/jsp/success.jsp"/>
    <forward name="fail"
             path="/WEB-INF/jsp/fail.jsp"/>
</global-forwards>
```

(4) action-mappings 要素

この要素で設定された内容に基づき、ActionMapping が生成されます。action-mappings 要素には特に重要な属性はなく、主な設定は子要素である action 要素で指定します。

action 要素の属性

属性名	必須	説明
path	○	Action を起動する URI を指定します。先頭は/ (スラッシュ) から始まります。
type	※	RequestProcessor より呼び出される Action の完全限定名を指定します。
forward	※	Action 以外のリソースをアプリケーションルートからの相対パスで指定します。Action を経由せずに、他のリソースへフォーワードするときに使用します。
include	※	Action 以外のリソースをアプリケーションルートからの相対パスで指定します。Action を経由せずに、他のリソースをインクルードするときに使用します。
parameter		Action に追加情報を渡すときに指定します。
roles		この Action へのアクセスを許可されたセキュリティロール名のカンマ区切りのリストを指定します。
name		使用する ActionForm のインスタンス名を指定します。対応する form-bean 要素の name 属性の値と一致させます。
attribute		name 属性で指定した ActionForm をスコープに登録する際の名前を指定します。デフォルトは name 属性の値です。
scope		name 属性で指定した ActionForm をどのスコープで登録するのかを request、session のいずれかで指定します。
validate		name 属性で指定した ActionForm において、validate()メソッドによる検証を行うかどうかを指定します。true としたとき、validate()メソッドが呼び出されます。デフォルトは false。
input		validate 属性が true の場合に、validate() メソッドが返す ActionErrors のサイズがゼロでないとき（検証が失敗したとき）に表示されるページを指定します。
unknown		この Action をデフォルトの Action とする際に true を指定します。いずれの Action にも関連付けられていないリクエストがあった場合に、この unknown 属性に true を指定している Action が実行されます。1 つのモジュール中、unknown 属性に true を指定できる Action は 1 つだけです。

※いずれか 1 つ指定する必要があります。

action 要素は、子要素として forward 要素と exception 要素を持ちます。forward 要素は、global-forwards 要素の子要素と同じように、Action が返す ActionForward を表すものです。exception 要素は、global-exceptions 要素の子要素と同じで、Action が例外をスローしたときの振る舞いが設定されます。

global-exceptions 要素、および global-forwards 要素の中に書かれたものは、全 Action から参照されるのに対して、action 要素の中に書かれたものは、この要素の type 属性で指定された Action のみが利用可能ということになります。

なお、action の子要素として記述する forward 要素、および exception 要素に指定可能な属性は、global-forwards 要素、global-exception 要素の場合と同じです。

ワイルドカードの利用

Struts1.2 からは action 要素の path 属性にワイルドカード"*"を使用することができるようになりました。例えば以下のような設定がされていたとします。

【ワイルドカードの使用例】

```
<action path="/regist*"
        type="sample.Regist{1}Action"
        name="{1}Form"
        scope="request"
        validate="false">
    <forward name="failure" path="/{1}Fail.jsp" />
    <forward name="success" path="/{1}.jsp" />
</action>
```

上記の設定は、/registUser、/registItem 等の"/regist"から始まる URL にマッチします。ただし、/registUser/update のように"/"を含むものにはマッチしません。また、上記の例にある{1}はマッチした文字列によって置き換えられます。つまり、/registUser という URL のリクエストがあった場合、*部分は User となるため、name 属性の値は”UserForm”を指定したものとみなされます。

また、ワイルドカードの*は複数個所に使用することができます。たとえば/*Regist*という path の指定に対して、/empRegistInput にアクセスした場合、{1}で emp、{2}で Input が呼び出せます。{0}には、URL の全体が格納されています。今回の例の場合は、{0}に"empRegistInput "が格納されます。

パターンにマッチした文字列を受け取ることのできる属性は、以下の通りです。

action 要素

attribute 属性	forward 属性	include 属性	input 属性
parameter 属性	roles 属性	type 属性	

forward 要素

path 属性

(5) controller 要素

controller 要素はモジュールごとのコントローラを設定するためのもので、RequestProcessor をデフォルトの設定で使用する場合は、省略することができます。

また、多少敷居が高くなりますが、RequestProcessor を拡張して独自の処理を行わせるということも可能です。このとき、独自の RequestProcessor を ActionServlet から利用できるように、controller 要素で設定します。

controller 要素の属性

属性名	必須	説明
processorClass		ActionServlet から呼び出すプロセッサクラスを指定します。デフォルトは org.apache.struts.action.RequestProcessor です。
contentType		ヘッダにセットするコンテントタイプを指定します。デフォルトは text/html です。
nocache		キャッシュを無効にするかどうかを指定します。true の場合、キャッシュは無効です。デフォルトは false です。
bufferSize		ファイルアップロードの際に使用する入力バッファのサイズ。デフォルトは 4096 です。
TempDir		ファイルアップロードの際に使用する作業ディレクトリ。デフォルトはサーブレットコンテナが提供するディレクトリです。

以下に独自の RequestProcessor を定義する例を示します。

controller 要素の使用例

```
<controller processorClass="sample.MyRequestProcessor" />
```

(6) message-resources 要素

メッセージリソースの設定を行う要素です。

message-resources 要素の属性

属性名	必須	説明
parameter	○	メッセージリソースファイルの名前を指定します。

resources パッケージに含まれるメッセージリソースファイル Messages.properties を指定する場合は以下のように記述します。

message-resources 要素の使用例

```
<message-resources parameter="resources.Messages" />
```

(7) plug-in 要素

自分で作成したプラグインや、Struts で提供されている Tiles プラグイン、Validator プラグインを利用するときは、この要素で指定します。

plug-in 要素の属性

属性名	必須	説明
className	○	プラグインクラス名を指定します。

また指定したプラグインが使用するプロパティに関する情報は、set-property 要素をネストすることで指定します。

set-property 要素の属性

属性名	必須	説明
property	○	プロパティ名を指定します。
value	○	プロパティに対応する値を指定します

例えば Tiles プラグインを使用する場合は、次のように記述します。

【Plugin 要素の使用例】

```
<plug-in className="org.apache.struts.tiles.TilesPlugin"
    <set-property property="definitions-config"
                  value="/WEB-INF/tiles-defs.xml"/>
</plug-in>
```

3. web.xml の設定

Struts を使用した Web アプリケーションでは、サーブレットコンテナから ActionServlet が利用されるように、web.xml に設定を行います。ここでは、Struts を使用するために必要な web.xml の設定について説明していきます。

1 web.xml の要素

web.xml の記述例を以下に示します。ここでは Struts に関連する要素のみについて解説します。

【web.xml】

```
<web-app>
・・・(略)・・・
    <servlet>
        <servlet-name>Action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet
        </servlet—class>                                          ①
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>  ②
        </init-param>
        ・・・(略)・・・
    <load-on-startup>0</load-on-starup>
    </servlet>
    ・・・(略)・・・
    <servlet-mapping>
        <servlet-name>Action</servlet-name>
        <url-pattern〉*.do</url-pattern>                           ③
    </servlet-mapping>
    ・・・(略)・・・
    <taglib>
        <taglib-uri>http://jakarta.apache.org/struts/tags-html</taglib-uri>
        <taglib-location〉/WEB—INF/lib/struts—html.tld</taglib-location>  ④
    </taglib>
    ・・・(略)・・・
</web-app>
```

(1) servlet 要素

Struts では、org.apache.struts.action.ActionServlet を Action というサーブレット名で登録するのが一般的です。なお、Struts では ActionServlet を拡張して、独自のものを作成することも可能です。その場合は、ここに自作のサーブレットを指定します。

init-param 要素ではサーブレットの起動時に引数として渡されるパラメータを設定し、子要素の param-name 要素でパラメータ名を、param-value 要素でパラメータの値を指定します。

Struts を使用する場合は、config というパラメータ名で、Struts 設定ファイルの場所をアプリケーションルートからの相対パスで指定します。

load-on-starup 要素は、Tomcat の起動時にサーブレットも起動させる場合に指定します。値はサーブレットの起動順となります。

(2) servlet-mapping 要素

servlet-mapping 要素には、サーブレットに転送する URL パターンを指定します。Struts では「do」という拡張子でアクセスされた場合に、それが ActionServlet へ転送されるように指定します。もし拡張子が do となっている Web アプリケーションを見つけたら、Struts が使用されていると考えてよいでしょう。

もし、Struts を使用して実装していることを隠蔽するため拡張子を変更したい場合は、web.xml の該当箇所を変更します。

(3) taglib 要素

これは特に Struts に限った話ではありませんが、カスタムタグを使用する場合は、taglib 要素で識別名と TLD ファイルの場所をアプリケーションルートからの相対パスで指定します。

2　Struts 設定ファイルの分割

アプリケーションの規模が大きくなり、見通しが悪くなった場合は、Struts 設定ファイルを分割することが出来ます。例えば、ActionForm の定義、フォーワードの定義、Action に関する定義を別のファイルとして保存する場合、param-value 要素のボディ部にそれらのファイル名をカンマで区切って並べれば、動作時には 1 つの Struts 設定ファイルとみなされます。

web.xml:Struts 設定ファイルの分割例

```
<init-param>
    <param-name>config</param-name>
    <param-value>/WEB-INF/form_beans.xml,
                 /WEB-INF/global_forwards.xml,
                 /WEB-INF/action_mappings.xml
    </param-value>
</init-param>
```

なお、この方法ではモジュールは分割されませんので注意してください。

4. モジュールの分割

アプリケーションが大規模になると、Struts 設定ファイルも肥大化しやすくなります。そのような場合に、Struts では、アプリケーションのモジュール化を行い、Struts 設定ファイルを複数に分割することができます。多くの場合、一般利用者と管理者ではアクセスできるページが異なります。そういった際に一般利用者と管理者のアクセスできるページをそれぞれ 1 つのモジュールとして取り出すことをモジュール化といいます。

モジュール化された Struts アプリケーションでは、それぞれのモジュールに対して 1 つの RequestProcessor が割り当てられ、リクエストをどちらのモジュールへ転送するのかを決定する処理は ActionServlet が行います。

1　モジュール分割の設定

モジュール化を行うには、まず web.xml の設定が必要です。以下にデフォルトモジュールに管理者用モジュールを追加した例を示します。

web.xml：モジュール分割の設定

```
<web-app>
・・・(略)・・・
    <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>                                                    ①
            <param-name>config/admin</param-name>
            <param-value>/WEB-INF/struts-config-admin.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
・・・(略)・・・
</web-app>
```

追加モジュールを設定するには、初期化パラメータのパラメータ名を

```
config/[モジュール名]
```

とします。ここで設定したモジュール名はモジュールにアクセスする際の URL の一部になります。ここでは param-name 要素に config/admin と指定していますので、"admin"がモジュール名です。

このモジュール配下の Action にアクセスする URL は

```
http://localhost:8080/StrutsSample/admin/****.do
```

となります。そして param-value 要素で、このモジュール用の Struts 設定ファイルのパスを指定します。

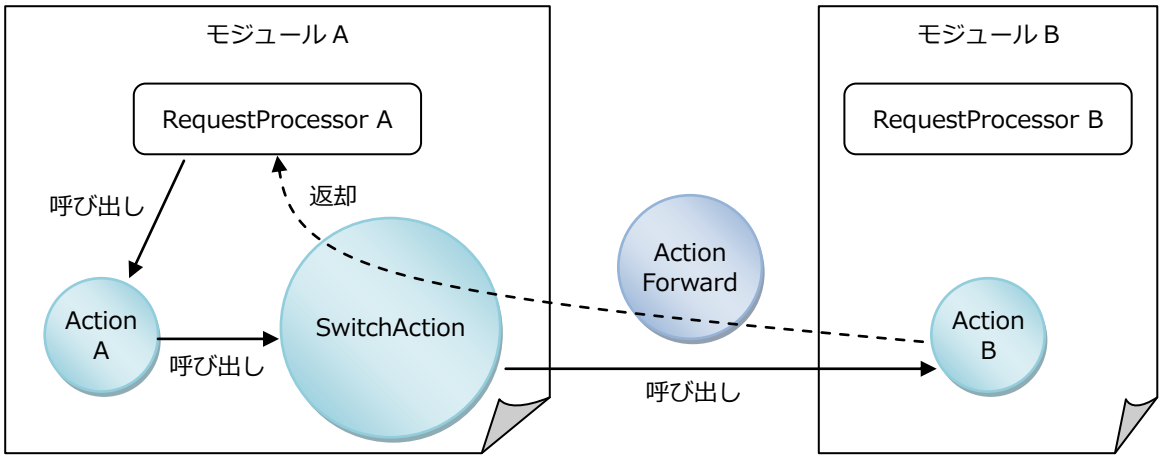
2　モジュール間のスイッチ

モジュールが異なる Action や ActionForm は別の RequestProcessor で管理されているため、通常の方法では相互に呼び出すことができません。これは、それぞれのモジュールが別の Struts 設定ファイルを参照しているため、ActionMapping や ActionForm の定義を検索することができないからです。

Struts は、これを可能にする方法を 2 つ用意しています。1 つはフォーワード（グローバル、ローカルを問わない）を使用する方法で、これは Struts1.2 で Struts 設定ファイルの forward 要素に module 属性が追加されたことにより可能になりました。

もう 1 つは SwitchAction を使用する方法です。SwitchAction は、モジュール間のコンポーネントの相互呼び出しのみを行う Action で、Struts 設定ファイルに記述するだけで使用できます。

【SwitchAction の動作】



以下にデフォルトモジュールから admin モジュールにスイッチする場合の、Struts 設定ファイルの記述例を示します。

【struts-config.xml：モジュール間呼び出しの例】

```
<struts-config>
    ・・・(略)・・・
    <!-- グローバルフォーワードを使用する場合-->
    <global-forwards>
        <forward name="admin_top"
                 module="/admin"
                 path="/index.do" />
    </global-forwards>

    <action-mappings>
        <!-- ローカルフォーワードを使用する場合 -->
        <action path="/sample02/toAdmin"
                type="sample02.SwitchModuleAction">
            <forward name="success"
                     module="/admin"
                     path="/index.do" />
        </action>
        <!-- SwitchAction を使用する場合 -->
        <action path="/sample02/switchAdmin"
                type="org.apache.struts.actions.SwitchAction" />
    </action-mappings>
    ・・・(略)・・・
</struts-config>
```

① ← module="/admin"

フォーワードを使用する場合は forward 要素の module 属性にスイッチするモジュールを指定します。指定する値はスラッシュ（"/")で始まります。

SwitchAction を使用する場合は、action 要素の type 属性に org.apache.struts.actions.SwitchAction を指定します。

SwitchAction を使用するにはこれだけでは不十分で、SwitchAction が呼び出される URL に以下の 2 つのリクエストパラメータを渡す必要があります。

パラメータ名	指定
prefix	呼び出したいモジュール名をスラッシュ付きで指定します。デフォルトモジュールにスイッチする場合は空文字列を指定します。
page	呼び出したいモジュールの URL を、モジュールのルートからの絶対パスで指定します。

admin モジュールにスイッチするには、以下のような URL を使用します。

```
http://localhost:8080/StrutsSample/sample02/switchAdmin.do?prefix=/admin&page=/index.do
```

以上がモジュールの分割方法となります。