StrutsとVelocityを統合する

5つの簡単なステップで、JSPよりも柔軟なVelocityに置き換える

George Franciscus (george.franciscus@nexcel.ca)
Principal
Nexcel

2005年 9月 20日

「Struts Recipes」の共著者George Franciscusが、StrutsアプリケーションにVelocityテンプレート・エンジンを統合する方法を、順を追って解説します。これを習得すれば、Strutsから期待される安心感はそのままで、JSRの置き換えとして非常に高速で柔軟なものを使えるようになります。

JSP（Java™ ServerPages）技術はあまりにも一般的になったため、Webページを作るに当たって他の選択肢があることを忘れそうになります。しかし最近では一部の開発者は、JSPに欠けている柔軟性を求めて、テンプレート・エンジンに関心を向けるようになっています。JSPでもテンプレート・エンジンでも、HTMLにデータを埋め込むことはできますが、どちらの技術にも独特な方法があります。Velocityテンプレート・エンジンは、JSPに代わるものとして、特に人気があります。Velocityは短期間で学ぶことができ、使い方も非常に簡単です。簡潔な構文は開発者にとって魅力的であり、パフォーマンス解析によればJSPを上回ることが知られています。またVelocityは、簡単にStrutsアプリケーションの中に統合することができます。

この記事では、Strutsアプリケーションの中にVelocityテンプレート・エンジンを統合する方法と、使い方について説明します。説明としては、最初に公式を示し、その後から順を追って手順を説明します。できあがるアプリケーションは、StrutsとVelocityとの組み合わせ、つまり輝くような組み合わせとなり、皆さんは自分がJSPに抱く忠誠に疑問を持つようになるでしょう。

最初に、ダウンロード・セクションから、この記事で使用しているソースコードやStruts、Velocity、Velocityのツール・パッケージなどをダウンロードしてください。この記事では、皆さんがStrutsを使ったMVCプログラミングに慣れていることを想定していることに注意してください。

テンプレート・エンジンについて

StrutsとVelocityの統合という単純な仕事に取りかかる前に、テンプレート・エンジンと、ビューを生成する上でのテンプレート・エンジンの役割について、よく理解しておいてください。一般的にテンプレート・エンジンは、特にVelocityは、HTMLの外にあるものです。Velocityは、テキス

ト・ボディーの様々な部分の中に、データを融合させます。このテキストはレターであったり、eメールであったり、XMLやHTMLであったりします。こうした面から見ると、Velocityテンプレート・エンジンは、Microsoft Wordの「差し込み（mail merge）」機能の名残のようなものです。差し込みを使うと、動的データ（名前やアドレス、電話番号など）をレターの中に容易に入れ込むことができます。初期の頃は、組織はこの機能を使って膨大な数のレターを生成し、郵便局に運び込んでいたのです。これがスパムの始まりというわけです。

Velocityとは何か

VelocityはJavaベースのテンプレート・エンジンであり、スクリプト風の方法でオブジェクトを参照するための、単純な、テンプレート・ベース言語を提供しています。Velocityでは、チームのメンバー間での責任の分離が可能となります。つまりWebデザイナーはビュー（ページのルック・アンド・フィール）に専念でき、Javaプログラマーはバックエンド・コードに専念することができます。ページ・レイアウトからJavaコードを取り去ることによって、Webアプリケーションは、後々の維持管理が簡単なものになります。StrutsのようなMVC開発フレームワークと組み合わせる場合、JSPやPHPに代わるものとして、Velocityは非常に有力です。

Webアプリケーションに使用する場合、Velocityの目的はJSPとほとんど同じです。つまり、HttpServletResponseのOutputStreamに送り出す前にHTMLを生成するために使うことができます。Strutsアプリケーションの中でのVelocityの使い方として、Struts Action内からレスポンスに書き込み、ヌルのActionForwardを返す、という使い方があります。この手法は確かに動作しますが、深刻な欠陥があります。つまり、struts-config.xmlファイルを使ってレスポンスを抽象化することができないのです。Action内部にビューを置くということは、レスポンスを変更したい時にはActionを変更する必要がある、ということを意味します。

この手法では、Strutsの最大の特長（つまりビューからフォーカスを抽象化できる）が生かせません。私の好みの方法としては、すべてのレスポンスをサーブレット（Velocityテンプレートにアクセスすることだけをします）に振り向け、データをコンテキストにマージし、レスポンスを生成し、それをブラウザーに向かって送り出します。この後を読むとすぐ分かると思いますが、Velocityを開発している人達は、こうしたステップを既に一まとめに同梱しています。ですから皆さんは、私が以下に説明する手順をたどるだけなのです。まだダウンロード・セクションを見ていない人は、ここでぜひダウンロードを行ってください。

Velocityに至る5つのステップ

StrutsとVelocityテンプレート・エンジンは非常に簡単に、素直に組み合わせることができます。実際、次の5つのステップを踏めば良いだけです。

1. VelocityのJARをクラスパスに置く
2. Velocityサーブレットを認識するためにweb.xmlファイルを変更する
3. Velocityのtoolbox.xmlをアプリケーションのWEB-INFディレクトリーの下に置く
4. ビューとしてJSPではなくVelocityテンプレートを指すようにstruts-configを変更する
5. 描画したいページそれぞれに対してVelocityテンプレートを作る

ここでは、StrutsとVelocityの統合処方を、おなじみの検索での使い方として説明しましょう。この例では、ユーザーは単純なアプリケーションを使って、ISBN番号で本を検索することができます。そして検索結果として、ISBN番号に一致する本が表示されます。

Strutsのタグは・・・不要です！

皆さんはこの段階で、これまでコーディング削減にあれほど役立った、あの素敵なStrutsタグを全部あきらめるのか、と思っているかも知れません。しかしJSPを使わないのであれば、Struts JSPタグも使っていないはずなのです。代わりに、Velocityツールがあります。VelocityのStrutsツールは、皆さんがStrutsで慣れ親しんだもの全てを、Velocityの柔軟性を加えて提供してくれるのです。

ステップ1. VelocityのJARをクラスパスに置く

まだVelocityをダウンロードしていなければ、今ダウンロードしてください。Velocityはそれ自体でも素晴らしいものですが、そのツール・パッケージを使うと、仕事をより良く、より速く行うことができます。特にStrutsツールは、皆さんご存じのStrutsタグを真似ています。Velocityテンプレート・エンジンとVelocityツールのダウンロードに関しては、ダウンロード・セクションを見てください。

必要なjarは、場合によって少しずつ変わることに注意してください。ここではJARのリストを公開する代わりに、皆さんがVelocityのホームページ（参考文献）を調べ、インストールの説明を読むようにお勧めします。必要なJARを手にしたら、それらを単純にWEB-INF\libの下に置きます。

ステップ2. Velocityサーブレットを認識するためにweb.xmlを変更する

次のステップとして、Velocityサーブレットを認識するためにStrutsのweb.xmlファイルを変更し、.vmで終わるリソース・リクエストをすべて、Velocityサーブレットに振り向けます（リスト1）。

リスト1. Velocityサーブレットを宣言するようにweb.xmlを変更する

```
<servlet>
  <servlet-name>velocity</servlet-name> |(1)
  <servlet-class> |(2)
    org.apache.velocity.tools.view.servlet.VelocityViewServlet
  </servlet-class>

  <init-param> |(3)
    <param-name>org.apache.velocity.toolbox</param-name>
    <param-value>/WEB-INF/toolbox.xml</param-value>
 </init-param>

 <load-on-startup>10</load-on-startup> |(4)
</servlet>

<!-- Map *.vm files to Velocity -->
<servlet-mapping> |(5)
  <servlet-name>velocity</servlet-name>
  <url-pattern>*.vm</url-pattern>
</servlet-mapping>
```

リスト1で何が行われているかを明確にしましょう。

- (1) では、Velocityサーブレットを宣言し、それにvelocityのハンドルを与えます。
- (2) では、Velocityサーブレットに対するクラス名を宣言します。

Velocityサーブレットは、「toolbox」というパラメーターを持ちます。このtoolboxが、そのアプリケーションで利用できるツールを宣言する場所です。従ってリスト1では、下記も行っています。

- (3) では、toolboxコンフィギュレーションがどこにあるかをVelocityServletに伝えています。
- (4) では、Velocityサーブレットが正しい時間にロードされるように、load-on-startupタグを設定しています。0以上の任意の値であれば、コンテナーはそのinit()メソッドを呼ぶことによって、サーブレットをロードします。load-on-startupタグのボディーの中に置かれた値が、様々なサーブレットを呼ぶ順番を決定します。例えば0は1の前に呼ばれ、1は2の前に呼ばれます。タグを省略すると、あるいはマイナスの値を使うと、サーブレット・コンテナーはいつでも自分が好きな時にサーブレットをロードできるようになります。
- (5) では、.vmという接尾辞を持つリソースのすべてのリクエストを、強制的にVelocityサーブレットに振り向けるように、サーブレット・マッピングを宣言しています。(5)の<servlet-name>は、(1) の<servlet-name>に一致する必要があることに注意してください。宣言とマッピングをインターリーブすると、ログでエラーが起きます。

ステップ3. toolbox.xmlをWEB-INFの下に置く

Velocityでは、ツールボックスを使う（あるいは作る）ことができます。ツールボックスは、何度も使うような有用な機能を含むクラスを登録するために使います。幸いVelocityでは、事前構築されたツールが幾つか用意され、使えるようになっています。しかもオリジナルのStrutsタグを真似た、数多くのStrutsツールが作られていますが、独自のツールを作る必要がある場合には、自由に作ることができます。リスト2はtoolbox.xmlを示しています。これはVelocity Toolsダウンロードの中にあるものです。このファイルは、Velocity JARと共にWEB-INFの下に置きます。

リスト2. toolbox.xml

```
 <?xml version="1.0"?>
<toolbox>
  <tool>
     <key>link</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.StrutsLinkTool
     </class>
  </tool>
  <tool>
     <key>msg</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.MessageTool
     </class>
  </tool>
  <tool>
     <key>errors</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.ErrorsTool
     </class>
  </tool>
  <tool>
     <key>form</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.FormTool
     </class>
  </tool>
  <tool>
     <key>tiles</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.TilesTool
```

```
    </class>
  </tool>
  <tool>
    <key>validator</key>
    <scope>request</scope>
    <class>
      org.apache.velocity.tools.struts.ValidatorTool
    </class>
  </tool>
</toolbox>
```

ステップ4.　struts-configを変更する

次のステップは、struts-config.xmlを変更して、JSRではなくVelocityビューを指すようにすることです。新しいconfigファイルをリスト3に示します。

リスト3.　Velocityビュー用に変更したstruts-config.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE struts-config PUBLIC
          "-//Apache Software Foundation//DTD
             Struts Configuration 1.0//EN"
          "http://jakarta.apache.org/struts/dtds/
            struts-config_1_0.dtd">

<struts-config>
    <form-beans>
        <form-bean name="searchForm" type="app.SearchForm"/>
    </form-beans>

    <global-forwards>
        <forward name="welcome" path="/welcome.do"/>
    </global-forwards>

   <action-mappings>
        <action
            path="/welcome"
            type="org.apache.struts.actions.ForwardAction"
            parameter="/pages/search.vm"/> |(1)

        <action
            path="/search"
            type="app.SearchAction"
            name="searchForm"
            scope="request"
            input="/pages/search.vm"> |(2)
            <forward name="success"
              path="/pages/results.vm"/> |(3)
        </action>
    </action-mappings>
</struts-config>
```

リスト3は、非常に典型的なStrutsアプリケーションのように見えますが、1つだけ小さな違いがあります。つまりクライアントをJSPに転送するのではなく、レスポンスが.vmファイルに振り向けられています（リスト3の参照1、2、3を見てください）。StrutsアプリケーションをJSPからVelocityビューに移行する場合、必要なことはグローバル・サーチを行って.jspから.vmに置き換えるだけ、ということがほとんどです。他は全て、そのままで良いのです。テンプレートを保存する場所は、通常JSPを保存する場所と同じです。必要なことは、JSPタグの代わりにVelocityコマンドを使うことだけです。

ステップ5. Velocityテンプレートを作る

リスト4は、サンプル・アプリケーションのサーチ・ページ用のVelocityテンプレートです。

リスト4. サーチ・ページ用のVelocityテンプレート

```
<HTML>
  <HEAD>
    <TITLE>Search</TITLE>
  </HEAD>
  <BODY>
    $!errors.msgs()||(1)
    <FORM method="POST"
      action="$link.setAction('/search')"> |(2)
      <h2>Book Search</h2>
      ISBN:<INPUT type="text" name="isbn">
      <INPUT type="submit" value="Submit" name="submit">
    </FORM>
  </BODY>
</HTML>
```

リスト4は、典型的なHTMLページですが、JSPタグもStrutsタグもありません。下記の要素は、あまりなじみのないものかも知れませんが、このようなことを行っています。

- (1) では、エラー・メッセージ・キュー上の任意のエラー・メッセージを取得するために$!errors.msgs() を使っています。
- (2) では、順方向検索用のURLを取得するために$link.setAction('/search') を使っています。

そして、それだけです。このテンプレートの残り部分は、皆さんが見慣れたものとほとんど同じに見えるはずです。リスト5は、このアプリケーションの結果ページ用のVelocityテンプレートを示しています。

リスト5. 結果ページ用のVelocityテンプレート

```
<html>
  <body>

  <h1>Book Details</h1>
  <a href="$link.setForward("searchEntry")">Search
    again</a> |(1)

  <h3>$book.title</h3> |(2)

    <b>ISBN:</b>$book.isbn<br>|(3)
    <b>Title:</b>$book.title<br>|(4)
    <b>Author:</b>$book.author<br>|(5)
    <b>Price:</b>$book.price<br>|(6)
    <b>No Pages:</b>$book.pages<br>|(7)
    <b>Description:</b>$book.description<br>|(8)
    <b>Publisher:</b>$book.publisher<br>|(9)
  </body>
<html>
```

リスト5には、JSPタグもStrutsタグも無いことに注意してください。さらに詳しく見てみましょう。

- (1) では、StrutsのLinkツールを使って、<a>タグのhrefをStrutsのforwardに設定しています。

- (2) では、$book titleプロパティーにアクセスしています。
- (3) では、$book isbnプロパティーにアクセスしています。
- (4) では、$book titleプロパティーにアクセスしています。
- (5) では、$book authorプロパティーにアクセスしています。
- (6) では、$book priceプロパティーにアクセスしています。
- (7) では、$book pagesプロパティーにアクセスしています。
- (8) では、$book descriptionプロパティーにアクセスしています。
- (9) では、$book publisherプロパティーにアクセスしています。

解説

StrutsとVelocityテンプレート・エンジンの統合に関して、必要なことはこれだけです。表面上は非常に単純（そして非常に素直です）ですが、この統合動作がどのように行われているかを少し考えてみましょう。

Strutsのアクション・マッピングは、単にJSPだけではなく、任意のビューを定義することができます。この記事では単純にアクション・マッピングを変更して、接尾辞としてjspではなくvmを持つファイルを返すようにしました。次にVelocityサーブレットを宣言し、Servletコンテナーに対して、vmという接尾辞を持つ任意のファイルを、VelocityViewServletに送るように伝えています。

VelocityViewServletはVelocityコマンドをHTMLレスポンスへと変換します。こうすることによって、VelocityViewServletは、ビュー・レスポンスに対するインターセプターとして動作します。Strutsコントローラーがそのビューを VelocityViewServletに転送すると、VelocityViewServletは、クライアントにレスポンスを送る前にvmファイルを処理します。Strutsアプリケーションの中にVelocityビューがどのように統合されるかについては、参考文献を見てください。

まとめ

ここで見た通り、StrutsとVelocityの統合は非常に単純です。この記事では、たった5つのステップで、必要なことがすべて分かるように解説しました。JSPの代わりにテンプレート・エンジンを使う利点は、エンジンによって、またシナリオによって異なります。Velocityの場合では、単純なこと、簡単に習得できること、そしてパフォーマンスが高いことが主な利点として挙げられるでしょう。

ダウンロード

内容	ファイル名	サイズ
Sample code	j-sr1-source.zip	3 MB

著者について

George Franciscus

George Franciscusは、Java Enterpriseのコンサルタントであり、Strutsの権威です。またManningから出版されている、Struts RecipesやStruts in Actionの共著者でもあります。彼の技術的、管理的コンサルティング・サービスはnexcel.caで受け付けています。