velocity模板引擎

模板引擎（web开发模板引擎）是为了使用用户界面与业务数据内容分离而产生的，它可以生成特定格式的文档，用于网站模板引擎会就生成一个标准的html文档。还是从MVC模式说起，M（Model，模型）包括后台的事务逻辑、真正处理事务的代码和商业逻辑等，它们是整个网站中最重要的工作部分。V（vew视图）是网页显示部分，这部分接受来自后台程序结果或数据，进行显示。C（controller，控制器），c在视图和模型之间传递控制，并根据要求调用相应的视图显示模型返回数据，主要负责调度工作。

这样职责划分明确，简化了开发过程中相关人员的工作，而我们平时使用JSP页面，可以在JSP页面中写java相关逻辑，这样在V中混淆了逻辑代码。所以java模板引擎出现，模板引擎的出现，它严格的区分程序开发功能和职责的划分。

# velocity模板介绍

Velocity是一个基于Java的模板引擎（template engine）。可以使用简单的模板语言（templatelanguage）来引用由java代码定义的对象Velocity将java代码从web页面中分离出来，这样为web站点的长期维护提供便利，同时也在JSP和PHP之间提供了一种可选的方案。

## Velocity优点

|  |
| --- |
| 1、容易集成在各种各样领域中。  2、为网页制作人员提供了一种清晰而简单的语法。  3、因为模板和代码是分离的，所以可以分别独立的开发和维护他们。  4、Velocity容易集成到一些java运行环境中，特别是servlet。  5、Velocity使得模板可以访问任何环境对象中的共有方法。  6、Java和SQL代码生成  7、XML处理和转换。  8、文字处理，比如生成trf文件等。 |

## Velocity与JSP,Freemarker的比较

|  |
| --- |
| **--Jsp**  优点：功能强大，可以写java代码；支持jsp标签（jsp tag）；支持表达式语言（el）； 官方标准，第三方jsp标签库，j2ee规范；性能良好。Jsp编译成class文件。  缺点： Jsp中可以写java代码，当jsp中写过多的java业务逻辑，容易破坏mvc结构。  **--Velocity**  优点：velocity不能编写java代码，可以实现严格的mvc分离。比jsp性能还好些。使用表达式语言，据说jsp表达式语言就是学velocity的。  缺点：不能编写java代码；用户群体和第三方标签库没有jsp多；对jsp标签支持的不够好。  **--FreeMarker**  同样和velocity一样，不能编写java代码，实现严格的mvc分离，性能不错，针对性能，**velocity>freemarker>Jsp**，但freemarker对JSP标签支持良好，内置大量常用功能，使用方便，宏定义非常方便。其缺点和velocity一样，不是官方标准，用户和第三方标签库没有jsp多。在复杂页面上（大量判断，日期金额格式化），freemarker的性能比较好。 |

# Velocity脚本基本语法

## velocity三种reference

|  |
| --- |
| **变量**：java对象的一种字符串化表示，返回值调用java的toString()方法的结果。  **方法**：调用的是对象的某个方法，该方法必须是public的，返回值也是toString ()，方法的参数也必须为String的。  **属性**：除了访问java的类属性外，等价于get..()方法。 |

## 基本符号

|  |
| --- |
| 1、"#"来标识velocity的脚本语句。  2、"$"来标识一个对象（或者变量）。  3、"{}"用来标识velocity变量。  4、"！"用来强制把不存在的变量显示为空白。  5、用双引号还是单引号表示，默认""，可以在stringliterals.interpolate=false改变默认处理方式 |

## 基本语法

### 变量

|  |
| --- |
| **#变量定义**  #($name="hello")  #($templateName = "index.vm")  #set($template = "$directoryRoot/$templateName")  #template  **#变量规范**  ${name}，也可以写成$name，但一般情况下使用${name}.  注意：当引用属性的时候不能加{}  **#变量赋值**  $name ="hello"  左侧：为变量或者属性的引用。  右侧：变量引用、字面字符串、属性引用、方法引用、字面数字、数组列表。  #set($monkey = $bill) ##变量引用  #set( $monkey.Friend = "monica" ) ## 字符串引用  #set($monkey.Blame = $spindoctory.weave($web) ## 方法引用  #set($monkey.Number = 123) ## 数字引用  #set($monkey.Say = ["Not",$my,"fault"] ##数组列表  ps：  1、右值是null，则左侧不会被赋值，会保留原来的值。  2、velocity 模板中未被定义的将被认为是一个字符串。  #set($foo ="gibbous")  $moon = $foo  输出结果为$moon=gibbous  3、velocity模板中不会解释为对象的实例变量。例如  $foo.getBar() 等同于$foo.Bar;  $data.getUser("jon") 等同于 $data.User("jon");  data.getRequest().getServerName()等同于   $data.Request.ServerName等同于$(data.Request.ServerName) |

### 关系和逻辑操作符

|  |
| --- |
| && == || !  #if($foo && $bar)  <strong> This AND that</strong>  #end |

### 注释

|  |
| --- |
| 单行注释：##this is single  多行注释：#\* .........\*#  文档格式:#\*\*.............\*#  xx |

### 条件语句

|  |
| --- |
| **#if(condition)**  **#elseif(condition)**  **#else**  **#end**  例如：  #set($arr=["jiayou","jiayou2","jiayou3"])  #foreach($element in $arr )  #if($velocityCount==1)  <div>jiayou</div>  #elseif($velocityCount==2)  <div>jiayou2</div>  #else  <div>jiayou3</div>  #end  #end |

### 循环

|  |
| --- |
| #foreach($element in $list)  This is $element  $velocityCount  #end  例如：  #set($list=["pine","oak","maple"])  #foreach($element in $list)  $velocityCount  This is $element.<br>  #end  $list 可以为Vector、Hashtable、Array。分配给$element 的值是一个java对象，并且可以通过变量被引用。  例如  #foreach($key in $list.keySet)  Key: $key--->value: $list.get($key) <br>  #end  Velocity 还提供了循环次数的方法，$velocityCount变量的名字是Velocity默认的名字，表示循环到第几次了。  #foreach($foo in [1,2,3,4,5])  $foo  #end  <br>  #foreach($bar in [2,1,0,-1,-2])  $bar  #end  <br>  #set($arr=[0,1])  #foreach($i in $arr)  $i  #end |

### 语句嵌套

|  |
| --- |
| #foreach($element in $list)  #foreach($element in $list)  this is $element .$velocity <br>inner<br>  #end  this is $element.$velocity <br>out<br>  #end |

### velocity中的宏

Velocity中的宏我们可以理解为函数。

①宏的定义

|  |
| --- |
| #macro(宏的名称 $参数1 $参数2 …)  语句体(即函数体)  #end |

②宏的调用

#宏的名称($参数1 $参数2 …)

说明：参数之间用空格隔开。

### #stop

停止执行模板引擎并返回，应用于debug很有帮助。

### $include和#parse

#include和#parse的作用都是引入本地文件，为了安全，被进入的文件只能在Template\_root目录下。这两个引入区别：

* #include可以引入多个文件。例如#include("one.gif","two.txt",  
  "three.html")
* # include引入的文件内容不会被velocity模板引擎解析。#parse引入的文件内容，将解析其中的velocity并交给模板，相当于把引入的文件内容copy到文件中。parse只能引入指定的单个对象。例如：#parse("layout/index.vm")

#parse是可以递归调用的。例如：

|  |
| --- |
| 调用者dofoo.vm中代码：  count down.<br>  #set($count= 8)  #parse(parsefoo.vm)  <br>all done with diao yong parsefoo.vm.  parsefoo.vm中代码：  $count  $set($count = $count - 1)  #if($count>0)<br>  #parse("parsefoo.vm") ##自己调用自己  #else  <br>递归调用结束  #end |

注：使用#parse来嵌套另一个vm页面时，变量共享问题，如：

a.vm 中嵌套了b.vm

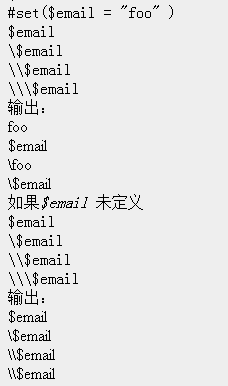
a.vm中使用了变量$param;

b.vm里可直接使用$param.

但其中一个定义了，则使用它自己定义的那个同名变量。

### 转义字符

当变量被定义的时候，两个\\代表一个\，如果未被定义，则按照原样输出。



### 内置对象

velocity内置了一些对象，在vm中可以直接调用。例如$request、$response、$session，使用$msg内的消息工具访问struts的国际化资源，达到简便实现国际化方法。

### 数组访问

数组要改成list等其他类容器方式来包装。

# SpringMVC集成Velocity