**Velocity教程**

参考：<http://blog.csdn.net/qq_25237663/article/details/52262532>

Velocity是一个基于Java的模板引擎，通过特定的语法，Velocity可以获取在java语言中定义的对象，从而实现界面和java代码的真正分离，这意味着可以使用velocity替代jsp的开发模式了(实际上笔者所在的公司已经这么做了)。这使得前端开发人员可以和 Java 程序开发人员同步开发一个遵循 MVC 架构的 web 站点，在实际应用中，velocity还可以应用于很多其他的场景.

1. Velocity的介绍

Velocity是一个基于Java的模板引擎，其提供了一个Context容器，在java代码里面我们可以往容器中存值，然后在vm文件中使用特定的语法获取，这是velocity基本的用法，其与jsp、freemarker并称为三大视图展现技术，相对于jsp而言，velocity对前后端的分离更加彻底：在vm文件中不允许出现java代码，而jsp文件中却可以.（me：即那些<%%>语句）

作为一个模块引擎，除了作为前后端分离的MVC展现层，Velocity还有一些其他用途，比如源代码生成、自动email和转换xml等，具体的用法可以参考[这篇文章](http://www.ibm.com/developerworks/cn/java/j-lo-velocity1/" \t "_blank).

2. Velocty的基本用法

在这里我们以一个HelloVelocity作为Velocity的入门实例.首先在[官 网](http://velocity.apache.org/download.cgi?cm_mc_uid=24596835221614568365819&cm_mc_sid_50200000=1470570755" \t "_blank)下载velocity的最新发布包，新建普通java项目，引入其中的velocity-1.7.jar和lib文件夹下的所有jar包即可. 然后分为如下两步：

2.1 初始化Velocity引擎

编写HelloVelocity.java文件如下：

public static void main(String[] args) {

// 初始化模板引擎

VelocityEngine ve = new VelocityEngine();

ve.setProperty(RuntimeConstants.RESOURCE\_LOADER, "classpath");

ve.setProperty("classpath.resource.loader.class", ClasspathResourceLoader.class.getName());

ve.init();

// 获取模板文件

Template t = ve.getTemplate("hellovelocity.vm");

// 设置变量

VelocityContext ctx = new VelocityContext();

ctx.put("name", "Velocity");

List list = new ArrayList();

list.add("1");

list.add("2");

ctx.put("list", list);

// 输出

StringWriter sw = new StringWriter();

t.merge(ctx,sw);

System.out.println(sw.toString());

}

首先，我们在代码中初始化了VelocityEngine这个模板引擎，对其设置参数进行初始化，指定使用ClasspathResourceLoader来加载vm文件。然后我们就可以往VelocityContext这个Velocity容器中存放对象了，在vm文件中我们可以取出这些变量，从而进行模板输出.

2.2 编写hellovelocity.vm文件

其中，vm文件放在classpath目录下即可，类加载器会进行加载   
hellovelocity.vm文件如下：

#set($greet = 'hello')

$greet $name

#foreach($i in $list)

$i

#end

控制台输出如下：

hello Velocity

1

2

2.3 Velocity的基本语法

本文中只简单的介绍几个Velocity的基本语法，具体可以参考[这篇文章](http://blog.csdn.net/nengyu/article/details/6671904" \t "_blank)

3.1 变量

在Velocity中也有变量的概念，使用$符声明变量，可以声明变量也可以对变量进行赋值(变量是弱类型的)。另外还可以使用$取出在VelocityContext容器中存放的值

#set(${!name} = "velocity")

#set(${!foo} = $bar)

#set($foo =“hello”)

#set($foo.name = $bar.name)

#set($foo.name = $bar.getName($arg))

#set($foo = 123)

#set($foo = [“foo”,$bar])

需要注意，上面代码中 $!{}的写法，使用$vari获取变量时，如果变量不存在，Velocity引擎会将其原样输出，通过使用\$!{}的形式可以将不存在的变量变成空白输出.

3.2 循环

在Velocity中可以使用循环语法遍历集合，语法结构如下：

#foreach($item in $list)

$item

$velocityCount

#end

其中，$item代表遍历的每一项，velocityCount是Velocity提供的用来记录当前循环次数的计数器，默认从1开始计数，可以在velocity.properties文件中修改其初始值

3.3 条件控制语法

在Velocity中可以使用条件语法对流程进行控制

#if(condition)

...dosonmething...

#elseif(condition)

...dosomething...

#else

...dosomething...

#end

3.4 宏

在Velocity中也有宏的概念，可以将其作为函数来理解，使用#macro声明宏

## 声明宏

#macro(sayHello $name)

hello $name

#end

## 使用宏

#sayHello("NICK")

3.5 parse和include指令

在Velocity中可以通过parse或者include指令引入外部vm文件，但是二者存在区别：include指令会将外部文件原样输出，而parse指令会先对其进行解析再输出(即对外部文件中的vm语法解析)

#parse("header.vm")

#include("footer.vm")

* 1
* 2

4. 在web项目中使用Velocity

velocity只是一个模板引擎，在web项目中使用Velocity还得添加一个HTTP框架来处理请求和转发，apache提供了[velocity-tools](http://velocity.apache.org/tools/devel/" \t "_blank)，其提供了VelocityViewServlet，也可继承VelocityViewServlet,从而实现自己的HTTP框架   
一般都是继承VelocityViewServlet，重写handleRequest方法，在其中存入公共的参数.

通过继承或直接使用VelocityViewServlet，可以在管理的vm文件中获得request、session与application对象，也可以直接获取在这几个域对象中保存的值，获取的顺序与EL表达式获取的顺序类似:   
${request} –> ${session} –> ${application}   
比如${testArr}，**获取testArr属性，velocity会在velocity的context中寻找。没找到在request域中找，没找到在session中找.**

下面将通过实例的方式讲解如何在web项目中使用Velocity   
首先引入velocity-tools及其依赖的[相关jar包](http://velocity.apache.org/download.cgi" \t "_blank)，然后分为如下4步：

4.1 继承VelocityViewServlet

通过继承VelocityViewServlet重写handleRequest方法，可以自定义转发规则

public class MyVelocityViewServlet extends VelocityViewServlet {

@Override

protected Template handleRequest(HttpServletRequest request, HttpServletResponse response, Context ctx) {

// 往Context容器存放变量

ctx.put("fullName","lixiaolin");

// 也可以往request域中存值

request.setAttribute("anotherName","xlli");

// forward到指定模板

return getTemplate("test.vm");

}

}

4.2 配置web.xml

对自定义的VelocityViewServlet配置就像配置普通的Servlet一样，如下:

<servlet>

<servlet-name>MyVelocityServlet</servlet-name>

<servlet-class>com.lxl.velocity.MyVelocityViewServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>MyVelocityServlet</servlet-name>

<url-pattern>/servlet/myVelocityServlet</url-pattern>

</servlet-mapping>

4.3 编写vm文件

vm文件是作为jsp的替代来展示给用户，在vm文件中可以获得在Context域或request等域中存放的值。默认情况下，会在资源根路径下搜索vm文件，所以直接将vm放在根路径下即可(也可以通过配置velocity.properties指定加载路径)   
如下：

#set($greet = "hello")

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Document</title>

</head>

<body>

<p>$!{greet} $!{fullName}</p>

<p>my another name is $!{anotherName}</p>

</body>

</html>

4.4 配置velocity.properties

通过配置velocity.properties文件，可以自定义vm文件加载方式，指定编码等。当然，也可以不配置velocity.properties，使用缺省的值即可.

## 设置模板文件加载器，webapp从应用根目录加载

resource.loader = webapp

webapp.resource.loader.class = org.apache.velocity.tools.view.WebappResourceLoader

## 模板路径，根目录下的vm文件夹

webapp.resource.loader.path = /vm

## 设置编码

input.encoding = UTF-8

output.encoding = UTF-8

最后，在浏览器中访问http://localhost:8080/VelocityApp/servlet/myVelocityServlet即可

5. 使用VelocityLayoutServlet

在web站点开发的过程中，经常会碰到几个页面的布局大致相同，比如引用相同的头部和尾部、左侧边栏相同等，在使用jsp开发时我们可以将头部等公共文件抽离出来，然后在实际页面中引入。Velocity也提供了类似的功能，并且该功能更加强大.

apache提供了VelocityLayoutServlet来实现页面布局，它是VelocityViewServlet的子类，通过使用VelocityLayoutServlet可以简化velocity下页面布局开发，可以使当forward到一个vm页面时，把该页面作为一个已有页面布局的一部分整体显示出来，比如访问资料页面，能够自动把头、尾部显示出来

velocity-tools包中已经包含了这个类，其使用分为如下几步：

5.1 配置velocity.properties

在/WEB-INF/路径下配置velocity.properties文件，指定模板布局文件的位置

input.encoding=UTF-8

output.encoding=UTF-8

## 定义加载器

resource.loader=webapp

webapp.resource.loader.cache=false

## 布局文件夹位置

tools.view.servlet.layout.directory = /templates/layout

## 定义默认布局文件

tools.view.servlet.layout.default.template = layout.vm

## 错误模板文件

tools.view.servlet.error.template = err.vm

5.2 布局母版vm文件

布局layout.vm文件是所有要展示的vm文件的母版，如下所示：

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>${page\_title}</title>

#if($!{CSS})

#foreach($\_css in ${CSS})

<link type="text/css" rel="stylesheet" href="${ContextPath}/$\_css">

#end

#end

</head>

<body>

<div class="header">

#parse("/templates/layout/header.vm")

</div>

<div class="container">

<div class="sub">

#parse($sub)

</div>

<div class="main">

$screen\_content

</div>

</div>

#if($!JS)

#foreach($\_js in $JS)

<script type="text/javascript" src="${CntextPath}/${\_js}">

#end

#end

</body>

</html>

其中，有个特殊的变量 screen\_content,这是Velocity内置的变量，代表将要转发的页面

5.3 编写转发的vm文件

#set($layout = "/templates/layout/layout.vm")

#set($CSS = ["scripts/css/index.css"])

#set($JS = ["scripts/js/jquery-1.11.3.js"])

#set($page\_title = "主页")

#set($sub = "/templates/sub.vm")

<div id="main-show">

this is main-show

</div>

5.4 继承VelocityLayoutServlet

public class MyLayoutServlet extends VelocityLayoutServlet {

@Override

protected void doRequest(HttpServletRequest request, HttpServletResponse response) throws IOException {

// 设置通用的变量

request.setAttribute("Request", request);

request.setAttribute("ContextPath", request.getContextPath());

request.setAttribute("BasePath", request.getScheme() + "://" + request.getServerName() + ":" + request.getServerPort() + request.getContextPath());

long runtime = System.currentTimeMillis();

super.doRequest(request, response);

if (request.getAttribute("close\_comment") == null) {

Date cur\_time = Calendar.getInstance(request.getLocale()).getTime();

PrintWriter pw = response.getWriter();

pw.print("\r\n<!-- Generated by VelocityApp Server(");

pw.print(cur\_time);

pw.print(") Cost ");

pw.print(cur\_time.getTime() - runtime);

pw.print(" ms -->");

pw.flush();

pw.close();

}

}

}

6. 附录及参考文献

参考文献   
\* [使用 Velocity 模板引擎快速生成代码](http://www.ibm.com/developerworks/cn/java/j-lo-velocity1/)   
\* [Velocity教程](http://blog.csdn.net/nengyu/article/details/6671904)

本文中的完整代码可在[github](https://github.com/jslixiaolin/GADemo)上下载.   
你可以通过jslinxiaoli@foxmail.com联系我.   
欢迎在[github](https://github.com/jslixiaolin)或者[知乎](http://www.zhihu.com/people/li-xiao-lin-4)上关注我 ^\_^.   
也可以访问个人网站: [https://jslixiaolin.github.io](https://jslixiaolin.github.io/)