**通向八段的道路**

**第十六天**

lifetragedy

目录

[一、 IBM Websphere与IBM HttpServer 2](#_Toc332973641)

[二、 基本概念 2](#_Toc332973642)

[三、安装IBM HttpServer 2](#_Toc332973643)

[四、使用WAS控制台自动生成与IHS集成时的plugins 8](#_Toc332973644)

[五、集成IHS与WAS 13](#_Toc332973645)

[5.1 配置httpd.conf 13](#_Toc332973646)

[5.2 开始整合 14](#_Toc332973647)

[第一步： 14](#_Toc332973648)

[第二步： 14](#_Toc332973649)

[第三步： 16](#_Toc332973650)

# IBM Websphere与IBM HttpServer

IBM WebSphere非常强大,几乎可以和任何主流Web服务器集成。其原理和Weblogic与Apache以及tomcat与Apache的集成原理一致，也是以plug-in(插件)的方式来做应用的。

不过我们今天要介绍的是IBM自带的HttpServer，又称IHS。

IHS其实就是一个Apache，IBM拿了Apache过来进行了改造，使得它可以和它的WAS系列产品紧密结合并且可以通过WAS的Admin Console对它进行可视化管理。

在今后的教程中我们会涉及到IBM WebSphere的集群及WebSphere Application Server Network Deployment也是使用这个IHS来做各个不同集群节点间的HTTP请求派发的。

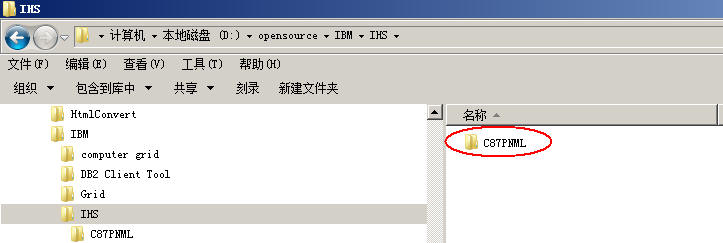
# 基本概念

IBM 的HttpServer其实就是一个Apache，除了插件生成这一块不需要像Weblogic或Tomcat那样需要去它们的官网下载相关的.so文件，其它设置都和Apache无疑，因此大家可以参考我的教程的“**第一天之Apache整合Tomcat**”与“**第三天之apache性能调优**”中的内容去调优和配置这个Apache。

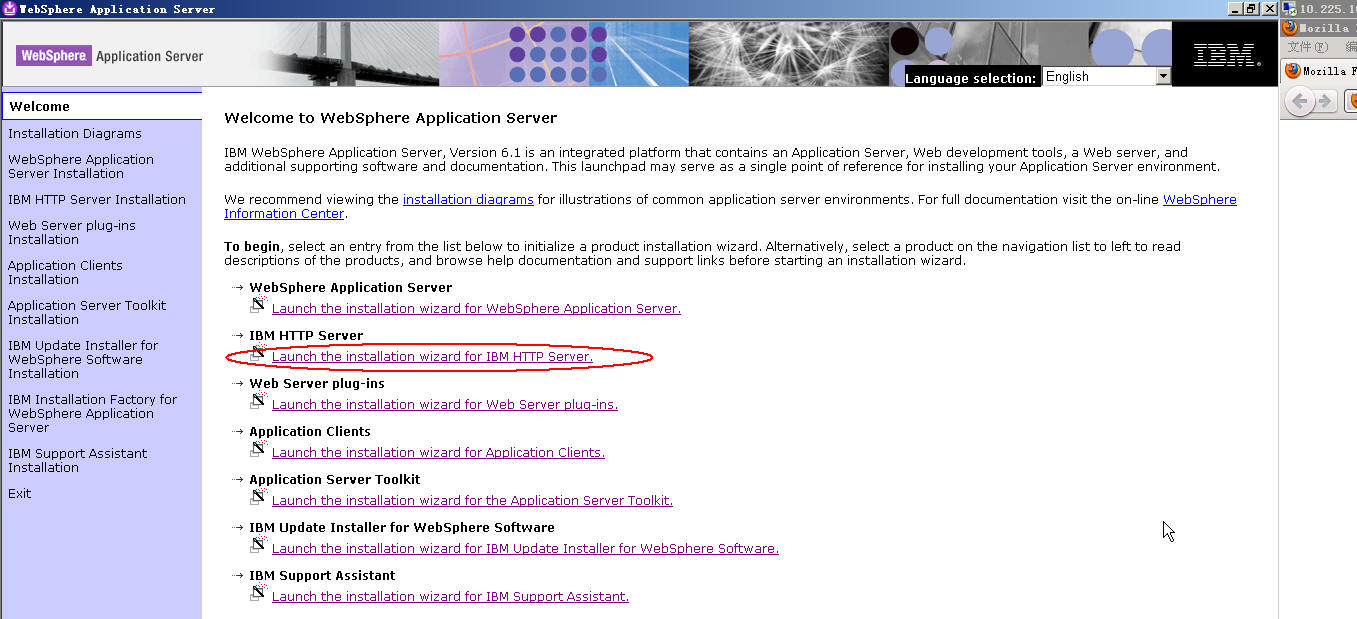
IBM的HttpServer与WAS作集成时，只要你装了IBM HttpServer那么它的plug-in是可以通过WAS的Admin Console动态**Generate**出来的。我们下面就来介绍这两者是如何集成的。

# 三、安装IBM HttpServer

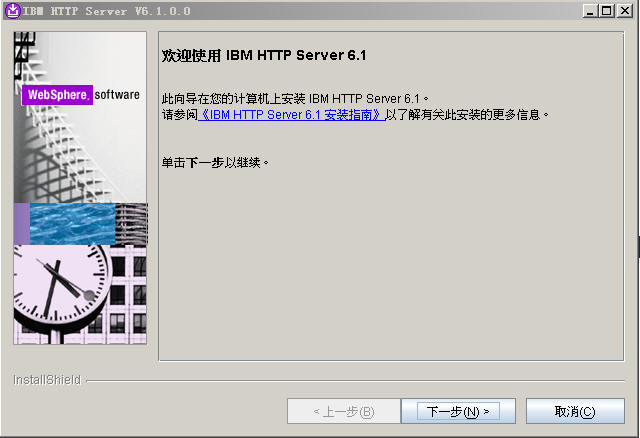
我们这边用的是IBM HttpServer，它在官网上下载后的文件名为” **C87PNML.zip**”，我们把它解压到一个指定目录中去。



找到launchpad.exe文件，运行它。

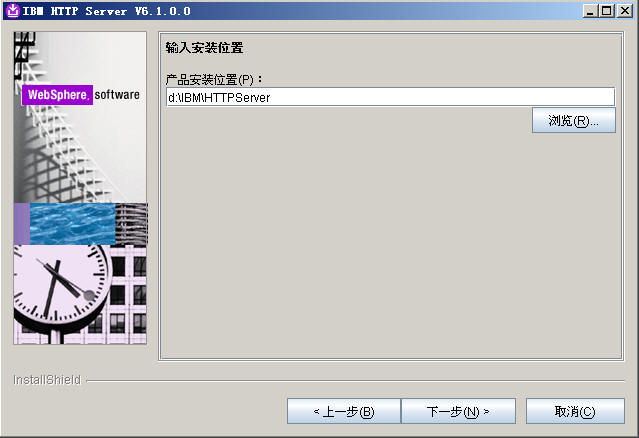


选择”**Launch the installation wizard for IBM HTTP Server**”项。



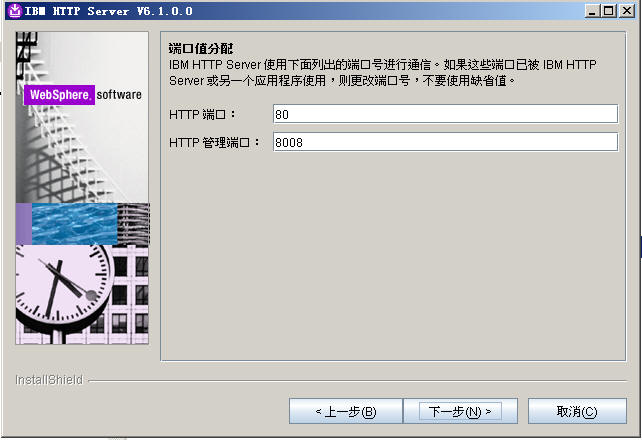
**[下一步]**

指定相关的安装目录。



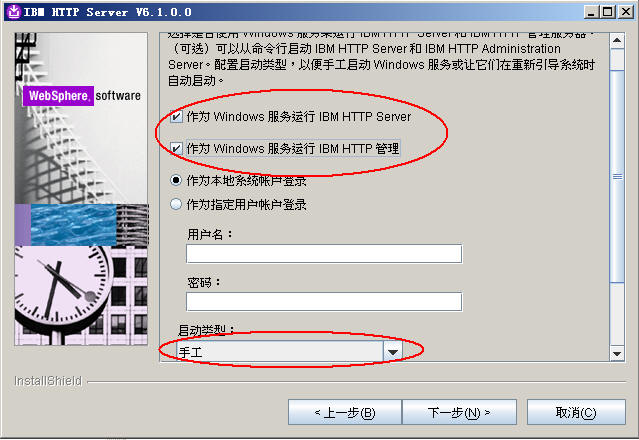
**[下一步]**

可以使用默认的端口，保持这个设置不动直接下一步



**[下一步]**

你可以把它作为windows的“服务项”，也可以不作为winodws的“服务项”，这个安装和apache是一样的。

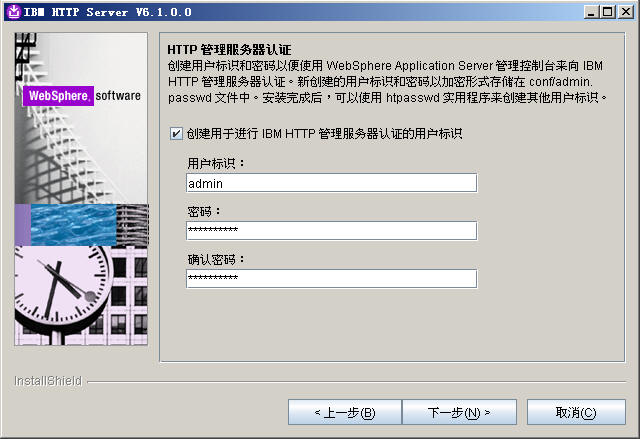


**[下一步]**

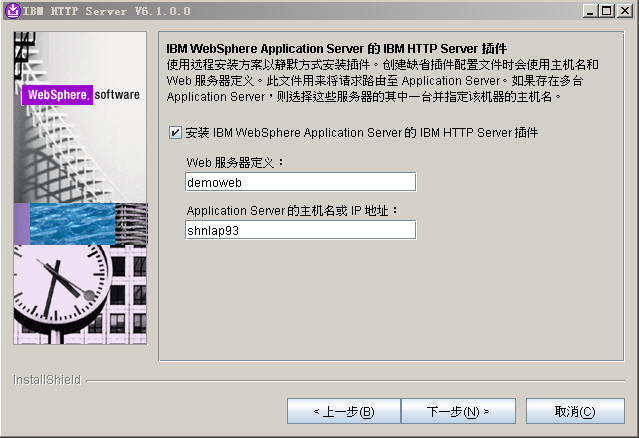
我们这边的用户名和密码保持和我们的WAS的控制台一致即：

**用户名** admin

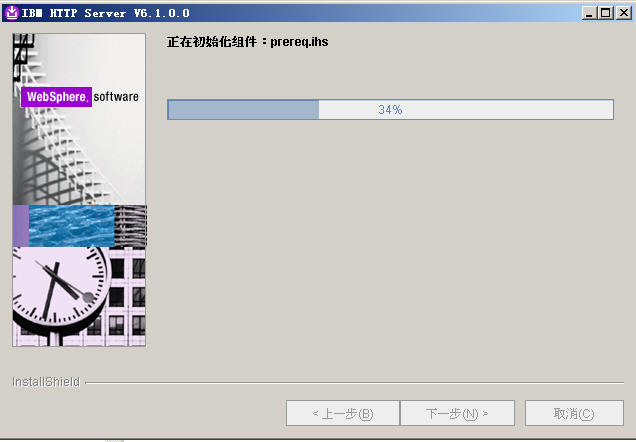
**密 码** password\_1



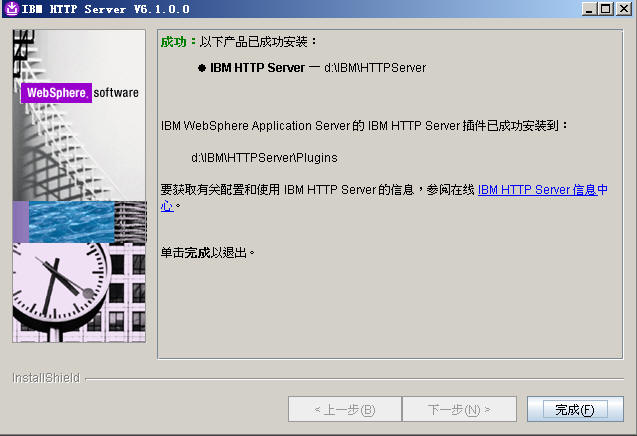
**[下一步]**



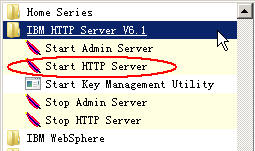
**[下一步]**->**[下一步]**后开始安装



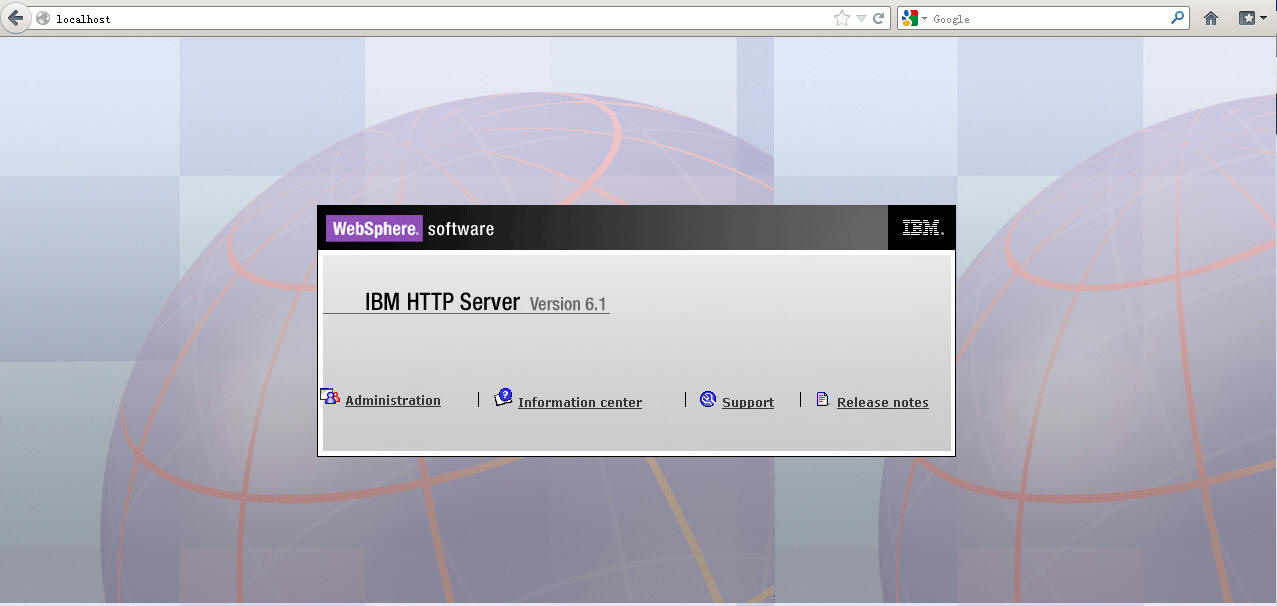
安装完后点击**[完成]**按钮



这是IBM HTTPServer安装完后在系统中生成的相关的菜单项，我们可以直接点这个Start Http Server。

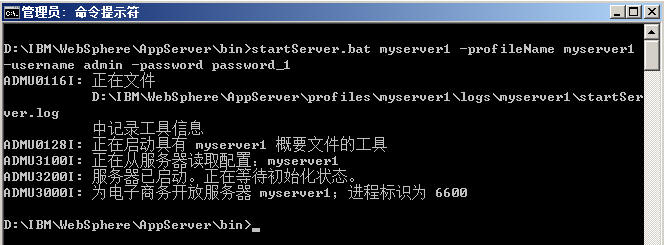


运行后在IE中输入<http://localhost>或者是<http://主机名> ，看到以下网页即代表我们的HTTPServer安装成功了。



# 四、使用WAS控制台自动生成与IHS集成时的plugins

我们先把我们的WAS启动起来。



通过<http://localhost:9060/ibm/console>登录WAS的admin console，点左边的**服务器**->**服务器类型**->**Web服务器**。



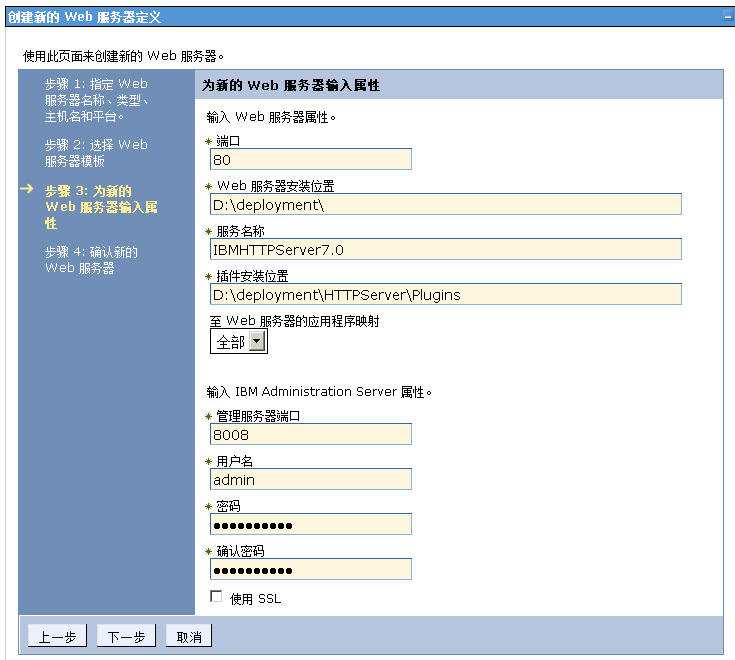
我们点击这个[**新建**]按钮，这个新建按钮将生成我们的plugins以及相关的集成配置文件。

在下面这个界面处填写我们的web server的配置名与主机名，并且选择我们的“**插件类型**”，因为我们用的是IHS+WAS，因此我们使用类型：**IBM HTTP Server**。



**[下一步]**

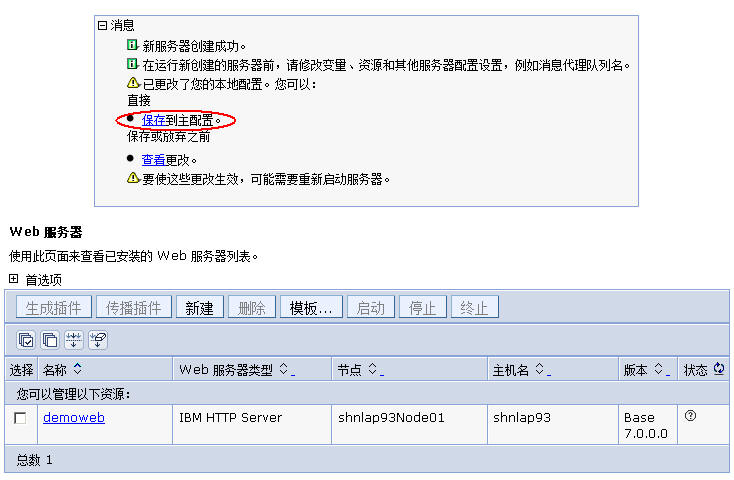
我们把即将要生成的IHS集成WAS的plugin及其配置文件生成到一个其它的目录，我们这边使用的是**d:\deployment\HTTPServer**目录用于存放我们从WAS中生成的IHS的plugins文件，因此WAS会在此目录下生成一个Plugins的目录。



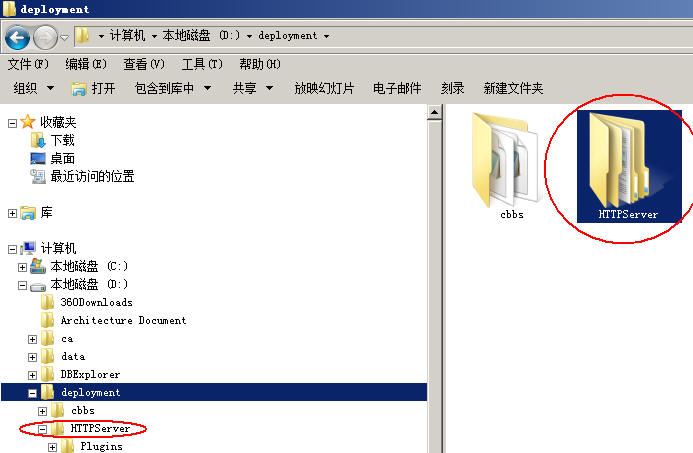
**[下一步]**



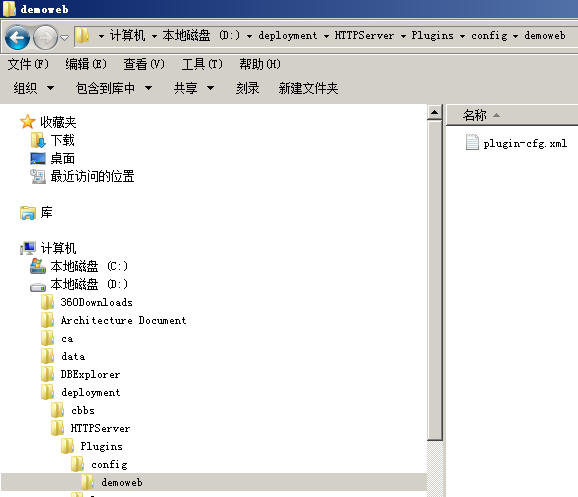
点击**[完成]**按钮，此时它会在我们的**d:\deployment\HTTPServer**输出相关的IHS的plugin与相关文件了，此步骤有点慢，稍等一会后，当出现下面这样的界面时



点击“**保存**”完成配置。然后我们去**d:\deployment\**下查看我们生成了什么东西。



我们可以看到在我们的**d:\deployment\HTTPServer**目录下生成了一个Plugins目录，其内容如下。



好了，现在我们开始要真正集成我们的IHS与WAS了。

# 五、集成IHS与WAS

## 5.1 配置httpd.conf

IBM HttpServer就是一个Apache，因此它的主配置文件也是httpd.conf文件。

我们先来到我们的httpserver的安装位置，如：D:\IBM\HTTPServer

我们用纯文本编辑工具打开“**D:\IBM\HTTPServer\conf\httpd.conf**”这个文件。我们来看最后一行：

|  |
| --- |
| LoadModule was\_ap20\_module "d:\IBM\HTTPServer\Plugins\bin\mod\_was\_ap20\_http.dll"  WebSpherePluginConfig "d:\IBM\HTTPServer\Plugins\config\demoweb\plugin-cfg.xml" |

如果你仔细研读过之前的Tomcat与Apache的整合或者是Weblogic与Apache的整合，这两句你一看就懂了。

我们在这两行下增加一个Virtual Host：

|  |
| --- |
| <VirtualHost \*:80>  ServerName shnlap93  ServerAlias shnlap93  ServerAdmin admin@shnlap93  DocumentRoot d:/www  DirectoryIndex index.htm index.html index.jsp  <Directory "D:/www/cbbs">  Options None  AllowOverride None  Order allow,deny  Allow from all  </Directory>  </VirtualHost> |

我们设置了一个virtual host，并使得这个host不具有目录浏览只能够解析静态html文件内容。

但好像我们发觉总觉得还是缺了点什么内容？还记得我们在做tomcat与apache结合时用的：

**JKMount \*.jsp**

或者是

Weblogic与Apache结合时用的：

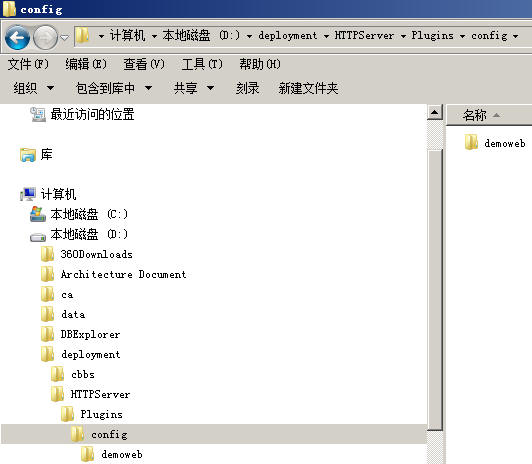
**MatchExpression /cbbs/WEB-INF**

这样的东西吗？

别急，下面来了。

## 5.2 开始整合

把我们在第四小节中生成的plugins的内容，即**D:\deployment\HTTPServer\Plugins\config**目录下的内容：



就是这个**demoweb**目录拷贝到我们的IBM HTTPServer的安装目录下去，如果有提示覆盖请选择“是，全部”。

如：

我们的HTTPServer是装在D:\IBM\HTTPServer\

那么我们就把：

**D:\deployment\HTTPServer\Plugins\config\**下的内容全部覆盖式拷贝到

**D:\IBM\HTTPServer\Plugins\config**目录内。

然后我们用纯文本编辑器打开**D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-cfg.xml**文件。

### 第一步：

把所有的d:\deployment\HTTPServer的字样换成我们的真实的HTTPServer安装的目录，因为我们的HTTPServer可能与我们的WAS是安装在两台不同的机器上的。

**如果你在生成plugins时使用的IHS的路径和我们的HTTPServer所在的路径是一致的，那么你大可跳过这一步**。

如我们这边就需要：

把所有的**d:\ deployment\HTTPServe替换成D:\IBM\HTTPServer**。

### 第二步：

在D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-cfg.xml文件内找到下面这一行：

|  |
| --- |
| <UriGroup Name="default\_host\_myserver1\_shnlap93Node01\_Cluster\_URIs"> |

往下看，我们可以看到如下这些行：

|  |
| --- |
| <UriGroup Name="default\_host\_myserver1\_shnlap93Node01\_Cluster\_URIs">  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/ivt/\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/snoop/\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/hello"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/hitcount"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="\*.jsp"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="\*.jsv"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="\*.jsw"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/j\_security\_check"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/ibm\_security\_logout"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/servlet/\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/\*"/>  </UriGroup> |

这些语句就相当于我们的：

**JKMount \*.jsp**

或者是：

**MatchExpression /cbbs/WEB-INF**

我们把先把下面这段注释掉，因为我们想让所有的/cbbs下动态的内容给我们的WebSphere去解释，所有的静态html内容给我们的IHS解析，而不是像它默认这样，所有的动态静态全交给了WebSphere去解析，这不是我们想要的。因此我们把这一行注释掉。

|  |
| --- |
| <!--  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/\*"/>  --> |

然后我们在这一行下再增加如下几行：

|  |
| --- |
| <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/WEB-INF/\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/\*.action"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/servlet/\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/\*.jsp"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/\*fckeditor/editor/filemanager/connectors/\*.\*"/>  <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid" Name="/cbbs/fckeditor/editor/filemanager/connectors/\*"/> |

### 第三步：

保存我们修改的httpd.conf文件，保存我们修改的plugins文件，重启我们的IBM HTTPServer。

如果一切无误，那么我们先来书写一个index.htm文件，并把它放在我们的d:\www\cbbs目录下，其内容如下：

|  |
| --- |
| <html>  <head>  <META HTTP-EQUIV="Refresh" CONTENT="1;URL=http://shnlap93/cbbs/index.jsp">  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>loading</title>  <body>  Loading…  </body>  </html> |

然后我们打开一个IE，输入: <http://localhost/cbbs/>，得到我们的应用的登录界面，输入相关的用户名与密码登录后，进行一些操作：



一切无误后，IBM HTTPServer与我们的IBM WebSphere结束成功。

结束今天的教程！