**通向八段的道路**

**第十七天**

lifetragedy

目录

[一、 IBM Websphere集群 4](#_Toc333445136)

[二、 基本概念 4](#_Toc333445137)

[三、 安装WASND 5](#_Toc333445138)

[3.1 WASND软件的安装 5](#_Toc333445139)

[3.2 创建DMGR（管理单元） 9](#_Toc333445140)

[3.3 创建AppSrv01与02（集群节点） 17](#_Toc333445141)

[四、 使普通的应用程序服务器与Dmgr联合 25](#_Toc333445142)

[4.1 联合AppSrv01节点入Dmgr 25](#_Toc333445143)

[4.2 联合AppSrv02节点入Dmgr 27](#_Toc333445144)

[4.3 重新用集群的方式启动Dmgr，AppSrv01, AppSrv02 28](#_Toc333445145)

[4.3.1 先依次关闭AppSrv02, AppSrv01, Dmgr 28](#_Toc333445146)

[关闭AppSrv02 28](#_Toc333445147)

[关闭AppSrv01 29](#_Toc333445148)

[关闭Dmgr 29](#_Toc333445149)

[4.3.2 依次启动集群各节点Dmgr, AppSrv01, AppSrv02 30](#_Toc333445150)

[先启动Dmgr 30](#_Toc333445151)

[再启动AppSrv01 30](#_Toc333445152)

[再启动AppSrv02 30](#_Toc333445153)

[4.4 利用WAS自带的admin console组建集群 31](#_Toc333445154)

[五、 在WASND中布署JDBC 36](#_Toc333445155)

[5.1 创建JDBC Provider 37](#_Toc333445156)

[5.2 数据源 37](#_Toc333445157)

[六、 在WebSphere集群环境中布署企业级应用 41](#_Toc333445158)

[七、 整合IBM HTTPServer与WASND 48](#_Toc333445159)

[第一步： 52](#_Toc333445160)

[第二步： 52](#_Toc333445161)

[第三步： 54](#_Toc333445162)

[第四步： 54](#_Toc333445163)

# IBM Websphere集群

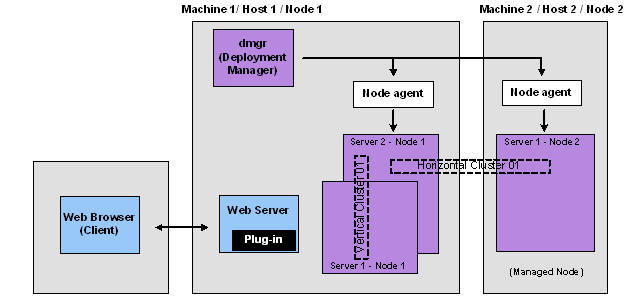
IBM WebSphere的发行版分为单机版和Network Deployment版，我们把Network Deployment称为ND即可作集群的版本。

WAS的集群和Weblogic的集群一样，它也可以作垂直和横向两种集群。

而WASND的集群是在所有的集群中最强大的，因为它可以：

通过各个node组成一个个cell，又可以把这一个个cell组成一个个新的cluster，而cluter与cluster还可以通过共属一个manager node来组成新的一个集群，以此可以通过WASND来组集超大规模集群。

同时，IBM的新一代产品如：网格计算（回头我们说完了WASND就来说这个），云计算一些企业级虚拟应用都是建立在WASND的基础之上的。



# 基本概念

使用WASND作集群的步骤很多，下面的教程有几十页之多，都是步骤，只要记住下面几个步骤就能抓住关键点了。

1. 先装WASND，废话，但你也必须要有WASND而不是一般的WAS，一般的WAS作不了集群。
2. 建立一个Deployment Manager节点，我们称它为DMGR。
3. 建立 “应用单元”。
4. 将“应用单元”与DMGR进行“联合”，这边的官方文档上写的是”Federate”，我们可以把它理解成“把应用单元”加入DMGR的管理之下。
5. 用startManager启动dmgr。
6. 用startNode启动各个node。
7. 进入dmgr的admin console可以启动和停止集群下的各个成员。

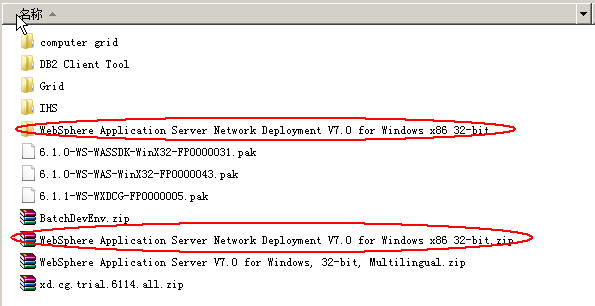
下面我们就来造一个集群，在本案例内我们装1个dmgr，两个node，然后把两个node加入该dmgr，然后使用IBM HttpServer(IHS)来做HTTP请求派发（**请见下面的表格-大小写严格区分**）

|  |  |  |  |
| --- | --- | --- | --- |
| 逻辑名称 | 单元功能 | profileName | server |
| Dmgr | 集群管理单元 | Dmgr | dmgr |
| Server1 | 集群节点1 | AppSrv01 | server1 |
| Server2 | 集群节点1 | AppSrv02 | server1 |

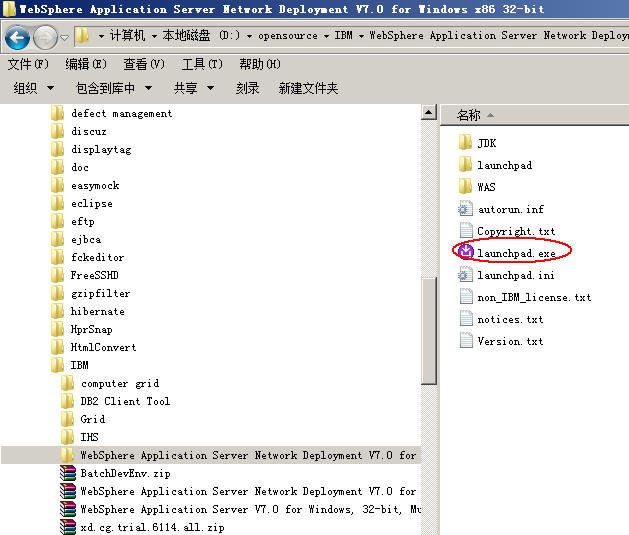
为了测试方便面，我们三个单元**全部启用“安全性管理”**，用户名与密码均为“**admin/password\_1**”。

# 安装WASND

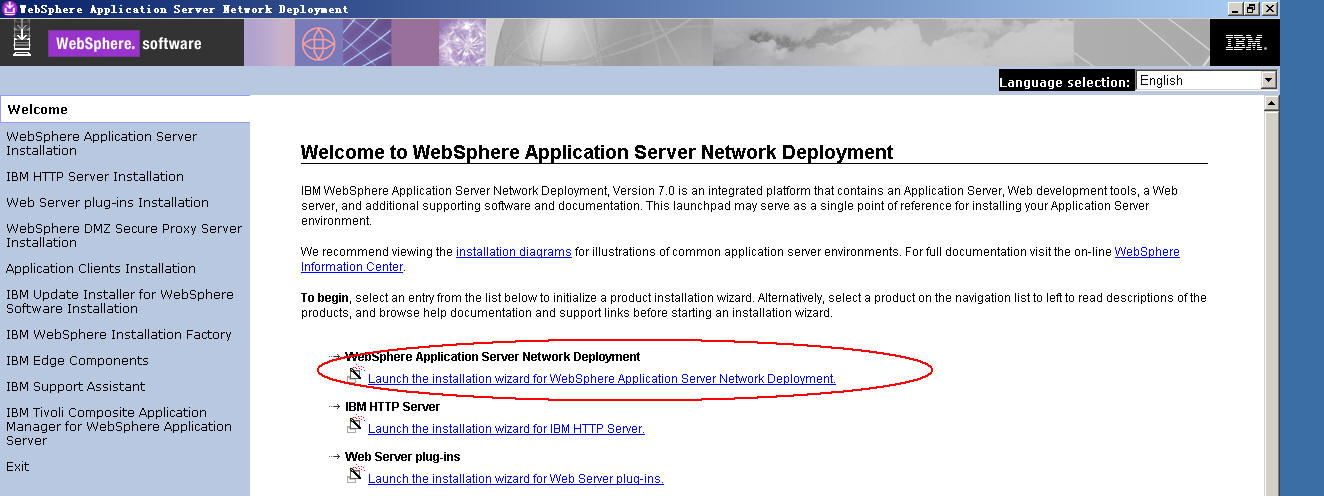
## 3.1 WASND软件的安装



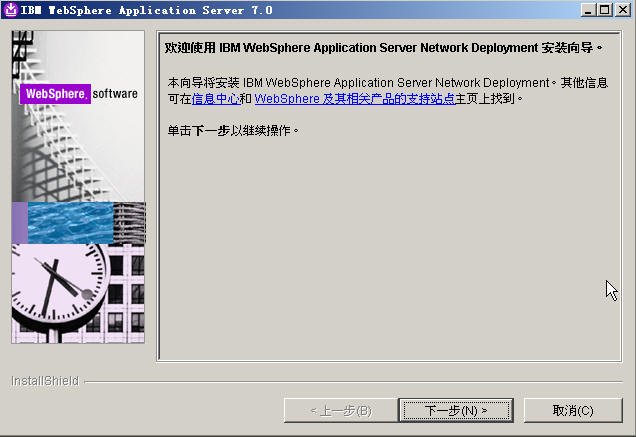
我们在此使用的是WASND7.0。



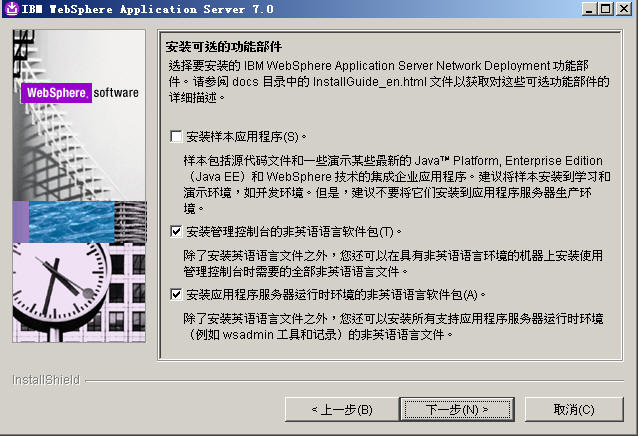
解压后进入相关目录双击launch.exe来启动WASND的安装界面。



开始安装。

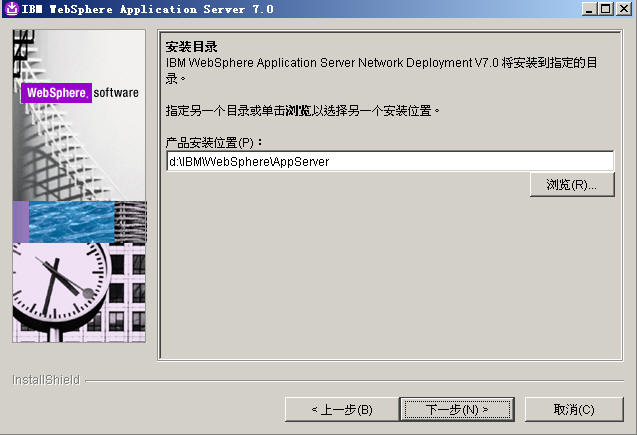


下一步



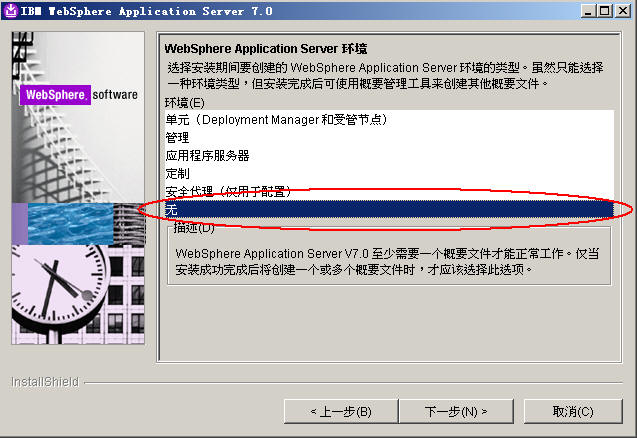
下一步

填入相应的安装目录（简单明了，不要装在什么c:\program files这样的目录名下），不要有空格，不要有特殊字符，JAVA很忌讳这个的。



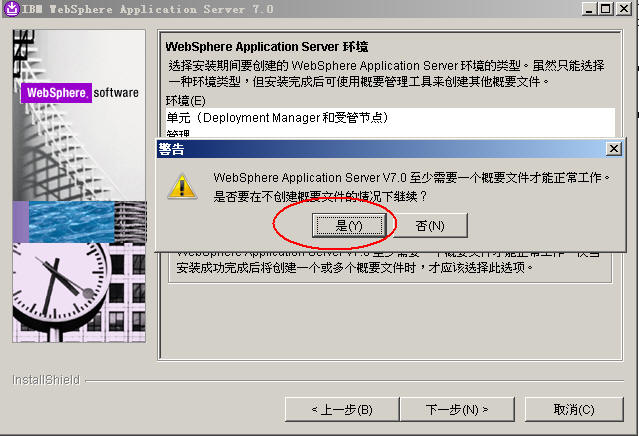
下一步

我们不要让安装程序替我们做任何节点的配置，我们将在安装完成后自己来手动配置，因此这边选“**无**”。

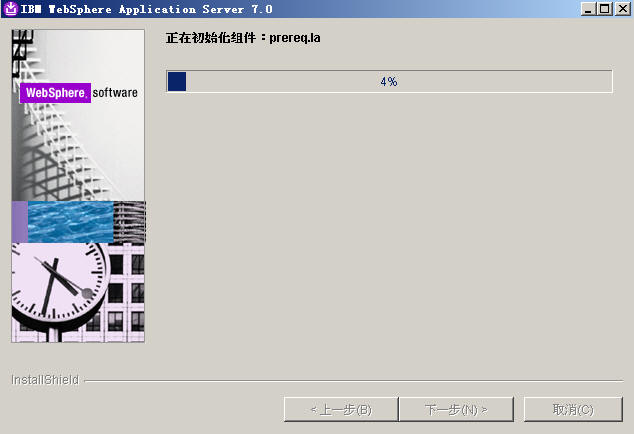


下一步

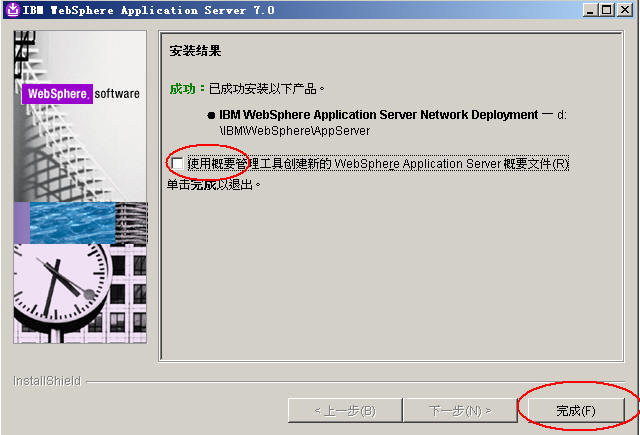
在弹出对话框中选[是]



一直下一步，直到安装开始。



装完后，把“使用概要管理工具创建新的WebSphere Application概要文件”选项前的勾去掉，并点击[完成]按钮。



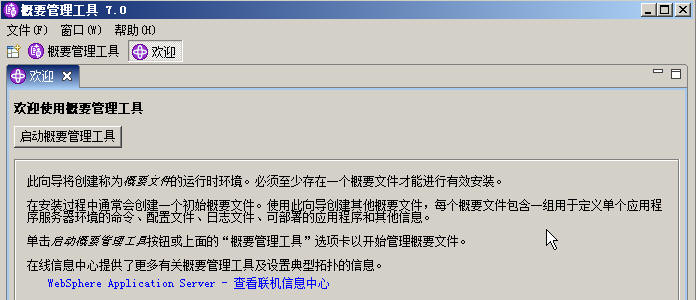
装完WASND后我们在系统菜单中会得到如下的菜单项



我们选“概要管理工具”来创建我们的dmgr吧。

## 3.2 创建DMGR（管理单元）

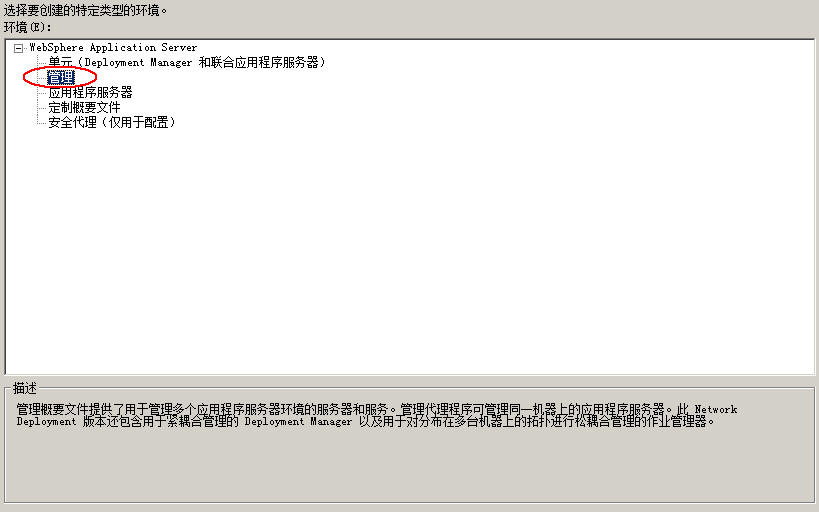
启动概要管理工具



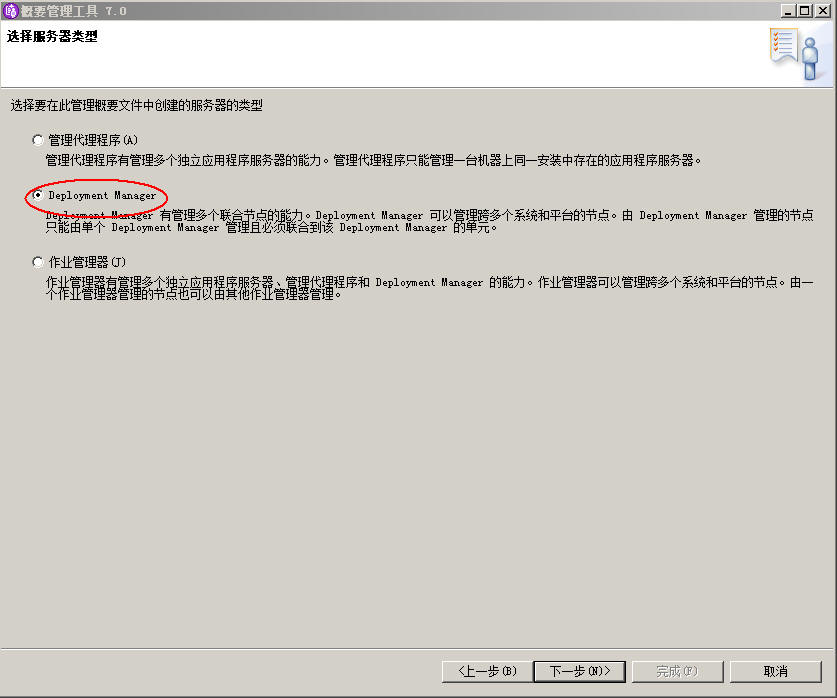
点[启动概要管理工具]



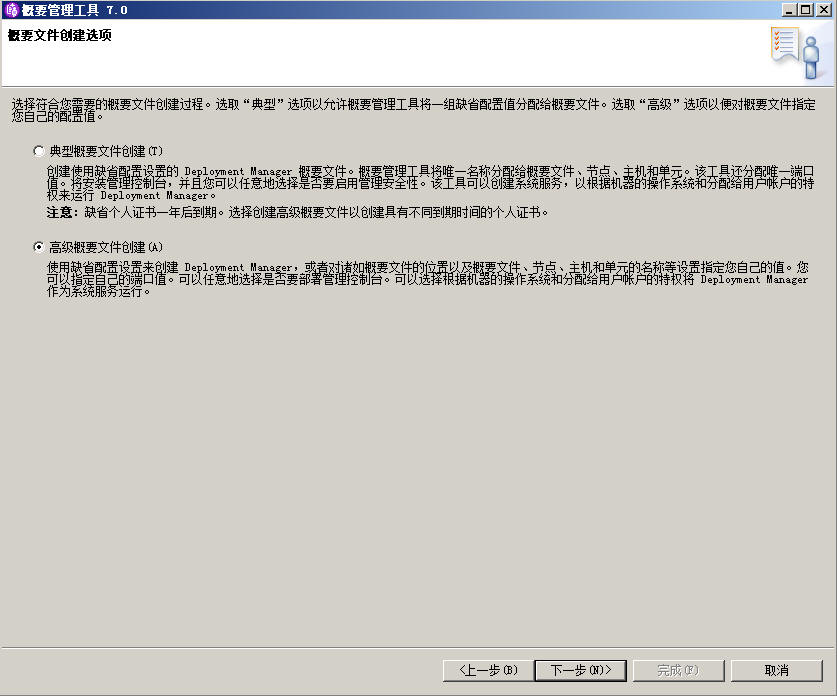
点[创建]按钮



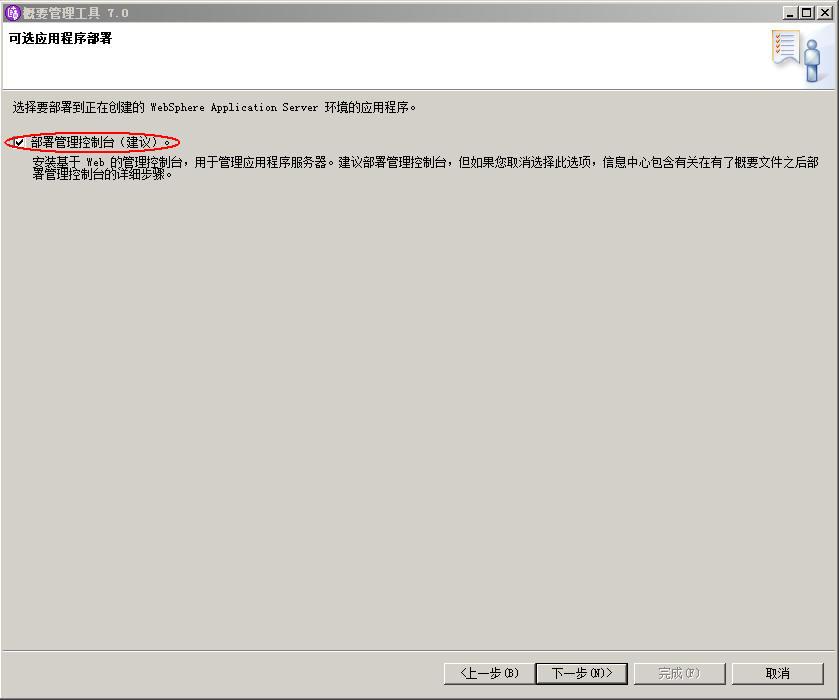
选择“管理”点[下一步]



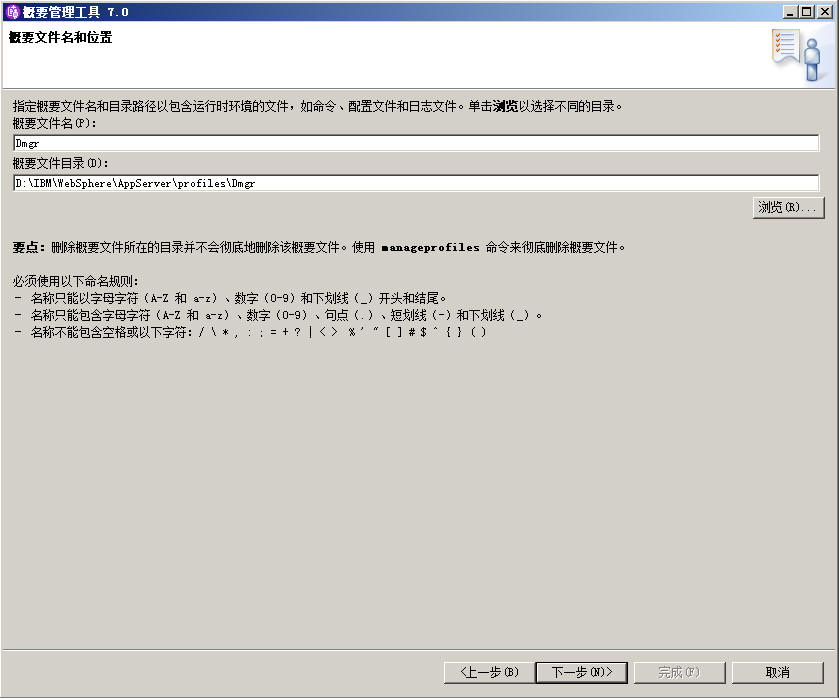
选择“Deployment Manager”选[下一步]



我们这边选择“高级概要文件创建”。



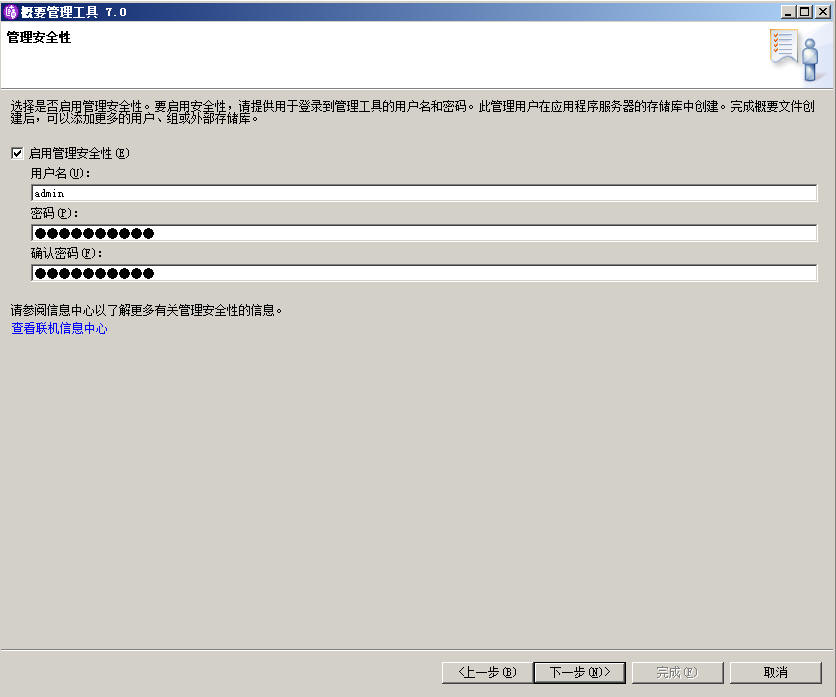
[下一步]



我们创建一个概要名即profileName为:Dmgr的概要文件，该概要文件包含一个Deployment Manager Server， server名为dmgr。

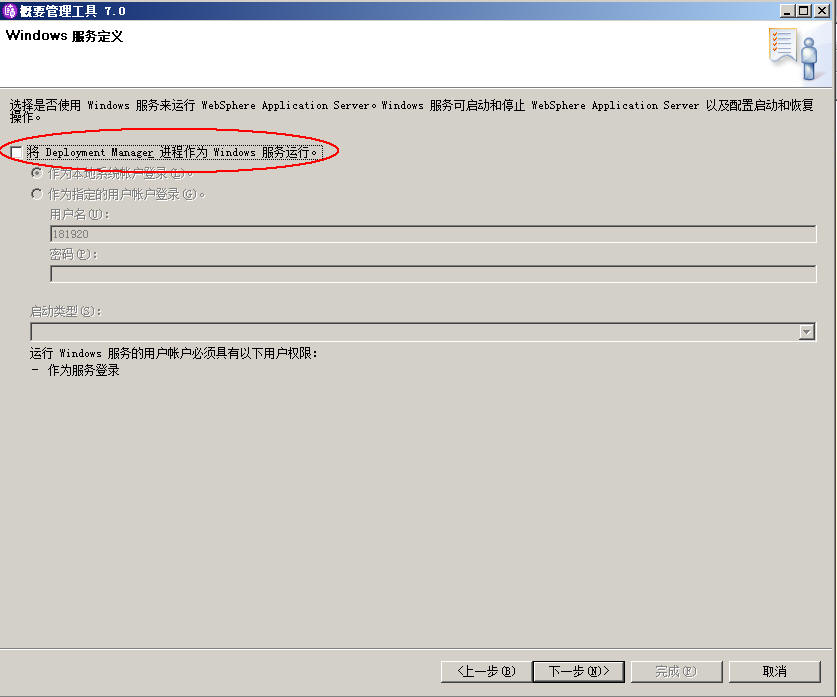
下一步，然后输入我们的admin console的用户名与密码，我们使用：

admin/password\_1。



下一步

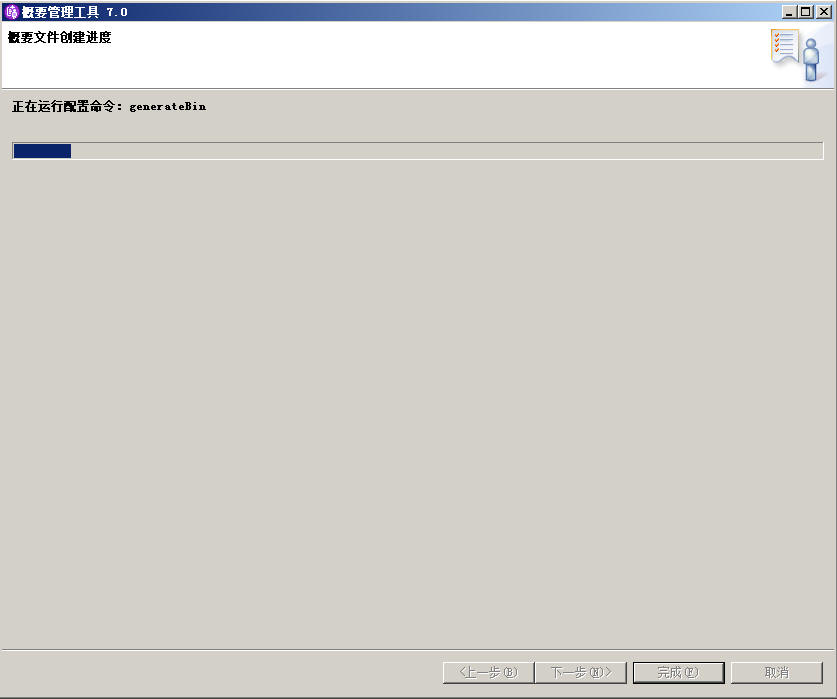
我们不希望这些创建后的单元随着我们系统的启动而启动，我们将在随后的教程中使用命令行来启动这些单元节点。



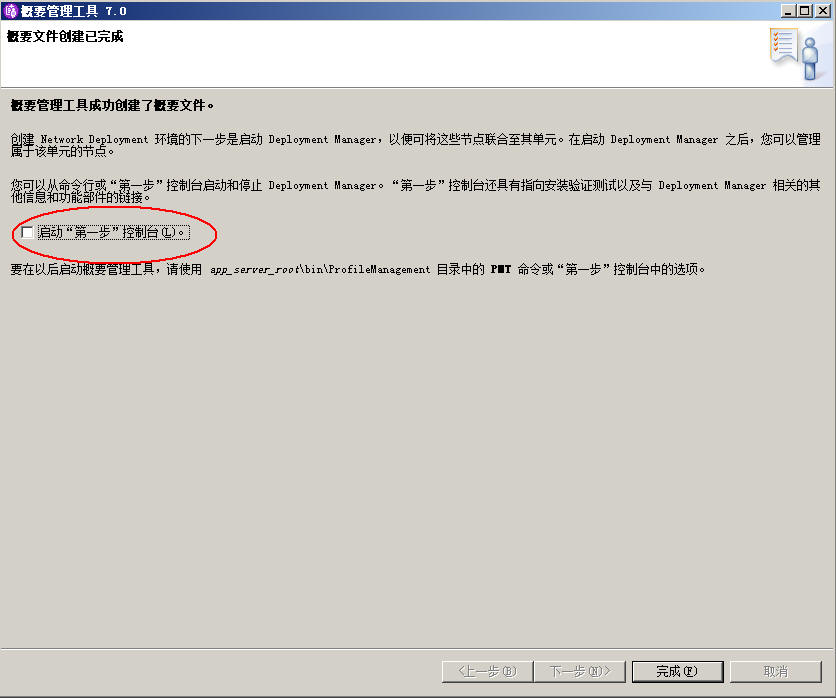
下一步



我们最后review一下我们的dmgr的配置，下一步开始[创建]这个管理单元的节点吧



创建完成成后不要勾选“启动第一步控制台”这个选项点[完成]按钮即可。



回到“根据管理工具”主界面，我们可以看到我们刚才的步骤已经创建了一个管理单元节点了。

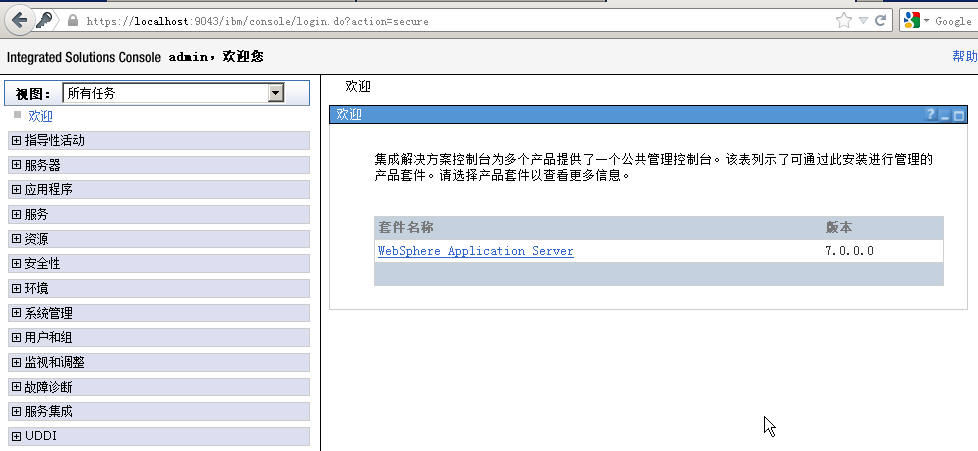


使用以下命令启动我们的管理单元



看到“进程标识为4980”的字样后代表我们的管理单元已经启动成功了。

我们在IE地址栏中输入[http://localhost:9060/ibm/console 后使用admin/password\_1](http://localhost:9060/ibm/console%20后使用admin/password_1)登录后看到管理控制台后代表我们的管理单元已经创建成功了。

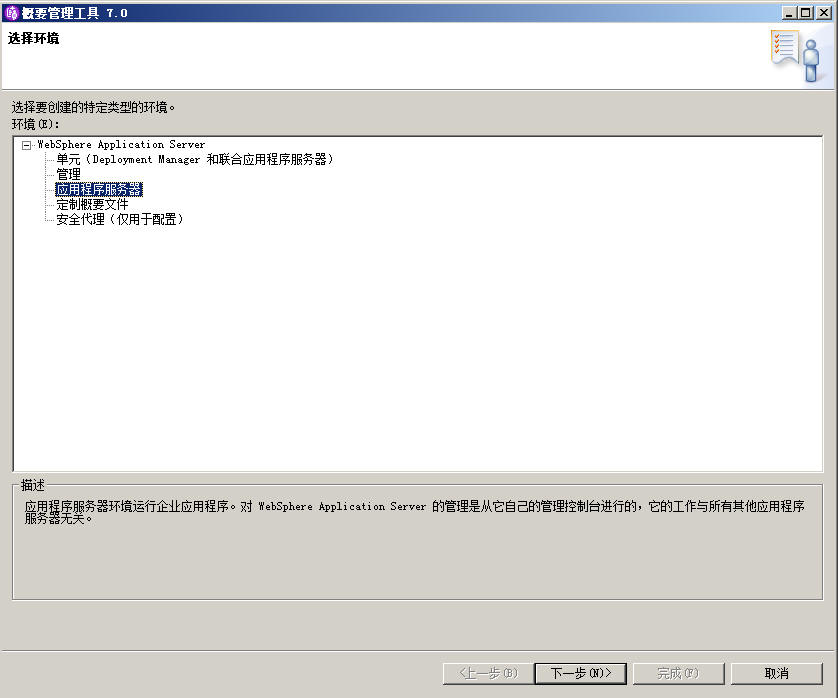


接下来我们就要开始创建我们的集群节点：AppSrv01与AppSrv02了。

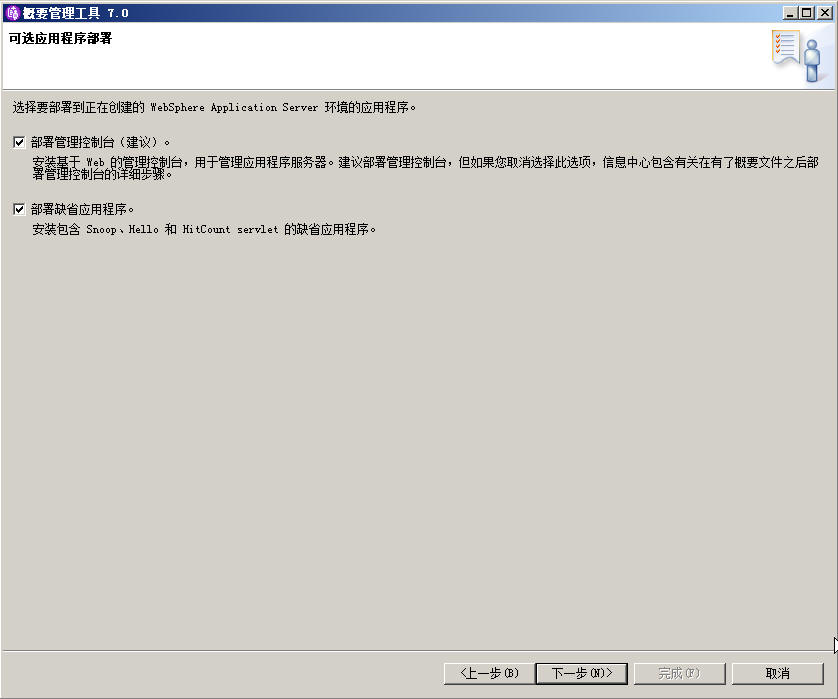
## 3.3 创建AppSrv01与02（集群节点）

依旧使用这个“管理概要工具”，点[创建]按钮。

这次我们要选“应用程序服务器”这个模板来创建我们的集群节点了。

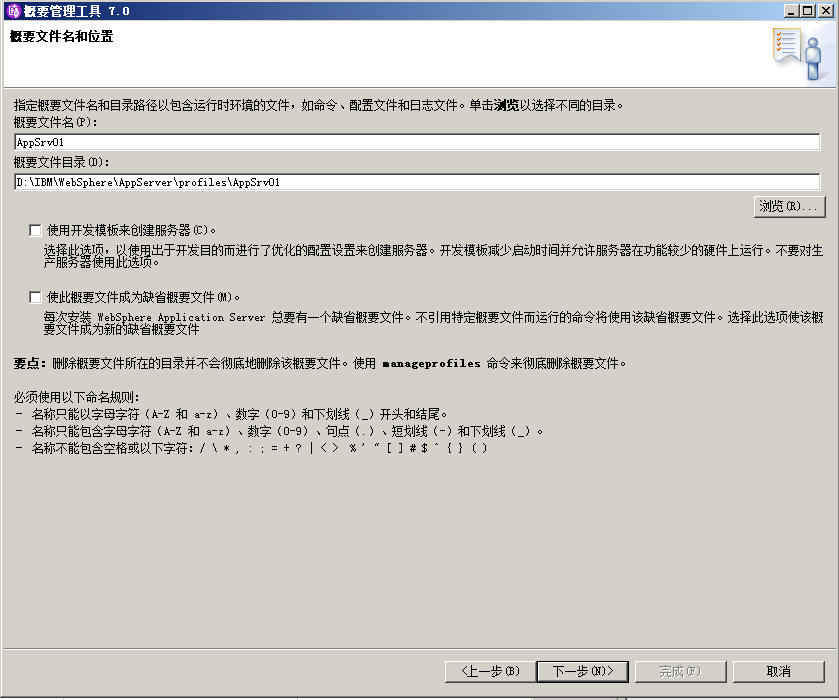


下一步



下一步

在此我们创建我们的第一个集群节点即：profileName为AppSrv01，server名为server1。



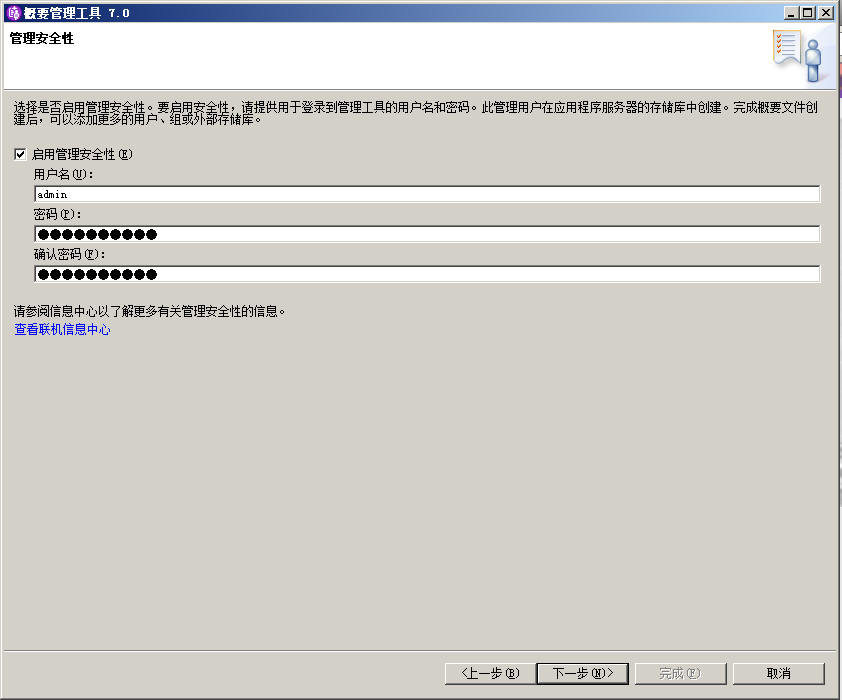
下一步

填入server名，我们这边填server1

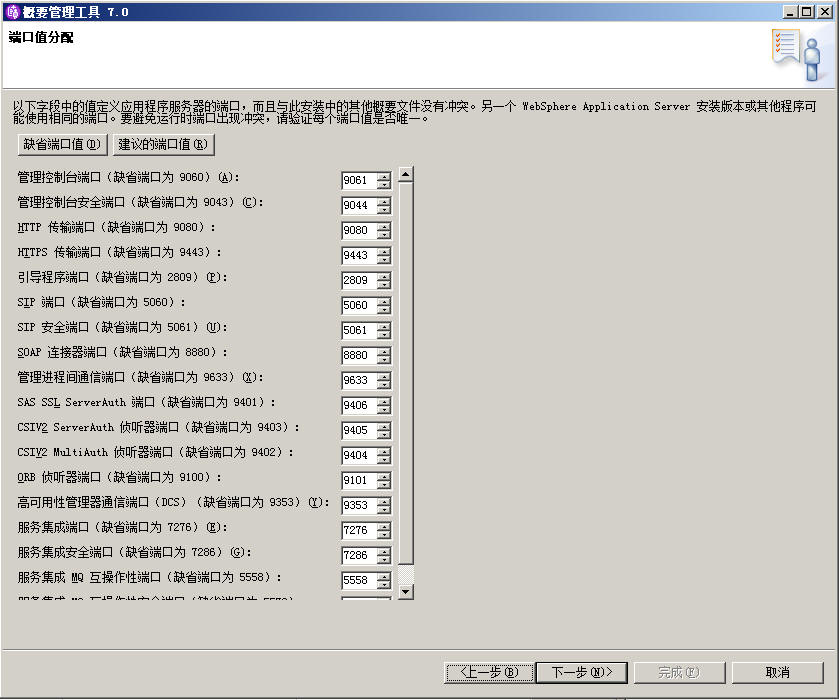


下一步

此处的用户名与密码还是用admin/password\_1。

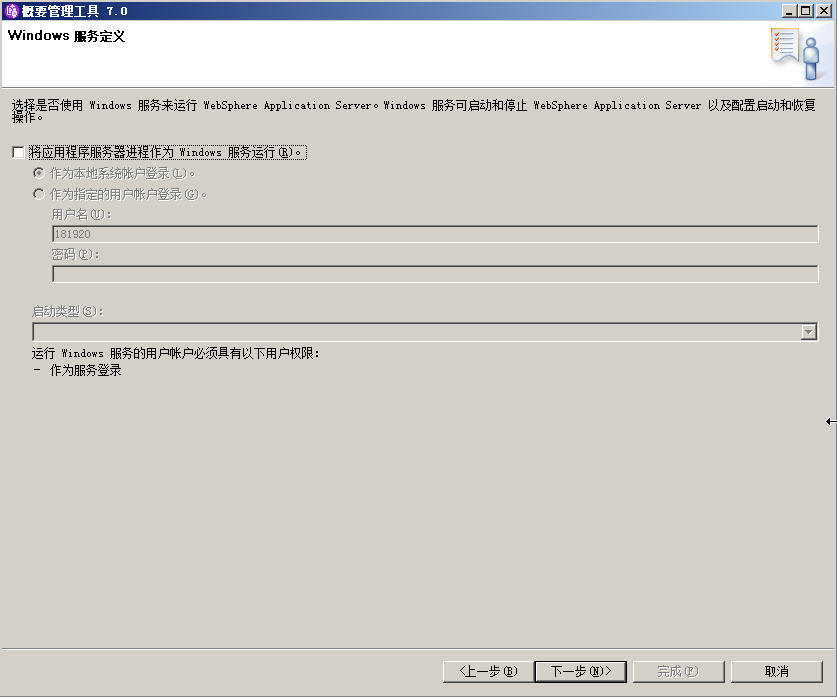


下一步



我们review一下我们的配置后下一步

和管理单元的创建一样，我们不把AppSrv01设为随着操作系统的启动而启动，我们将手工启动AppSrv01与AppSrv02。



下一步



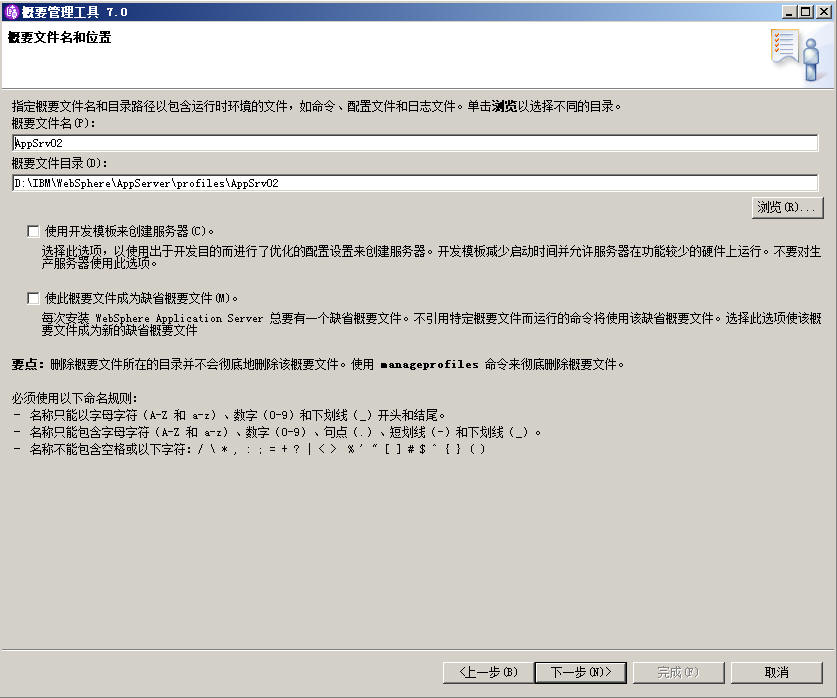
最后确认一下，我们开始创建我们这个节点。

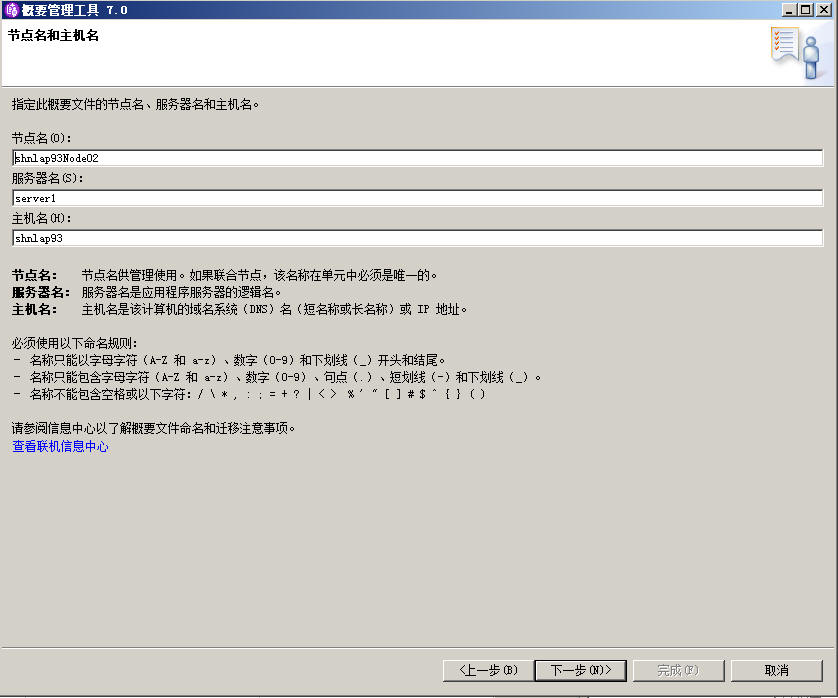
创建完了AppSrv01后我们依照AppSrv01的步骤再创建一个：

profileName为AppSrv02；

Server Name:为server1（**这边也是server1**哦，看仔细了）；

的“应用程序服务器”

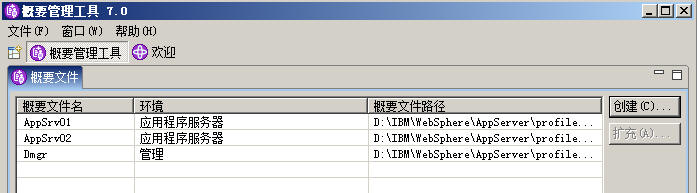




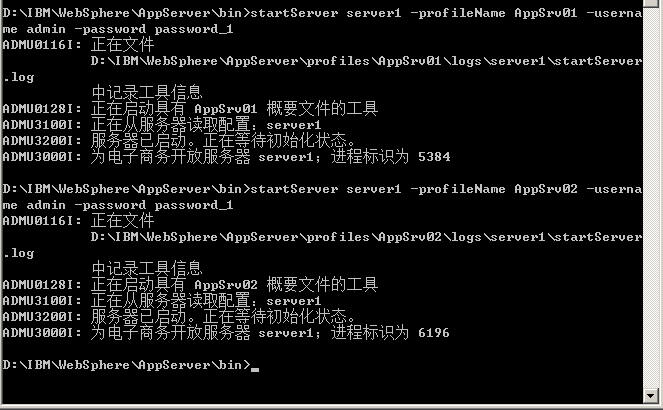


全部节点创建完成后对照着我们回过头来，再来对照着上面的那个表格，来看下面这个列表。

|  |  |  |  |
| --- | --- | --- | --- |
| 逻辑名称 | 单元功能 | profileName | server |
| Dmgr | 集群管理单元 | Dmgr | dmgr |
| Server1 | 集群节点1 | AppSrv01 | server1 |
| Server2 | 集群节点1 | AppSrv02 | server1 |



我们依次启动AppSrv01与AppSrv02来测试我们刚才的创建是否成功。



两个server的节点起来后我们在IE地址栏分别用：

<http://localhost:9061/ibm/console>与<http://localhost:9062/ibm/console>进行登录，如果都能登录成功，代表刚才我们的创建一切无误。

接下去我们就要开始我们的“集群”了，即“联合集点”的工作了。

# 使普通的应用程序服务器与Dmgr联合

## 4.1 联合AppSrv01节点入Dmgr

进入： d:\IBM\WebSphere\AppServer\profiles\AppSrv01\bin目录

键入：addNode.bat localhost 8879 –username admin –password password\_1进行节点联合

**此处的localhost 8879即为你的Dmgr所在的ip与dmgr的soap连接端口**。

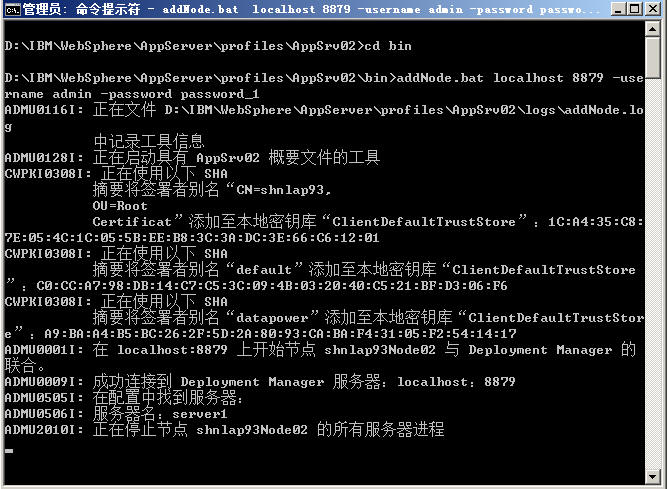
**注意：**

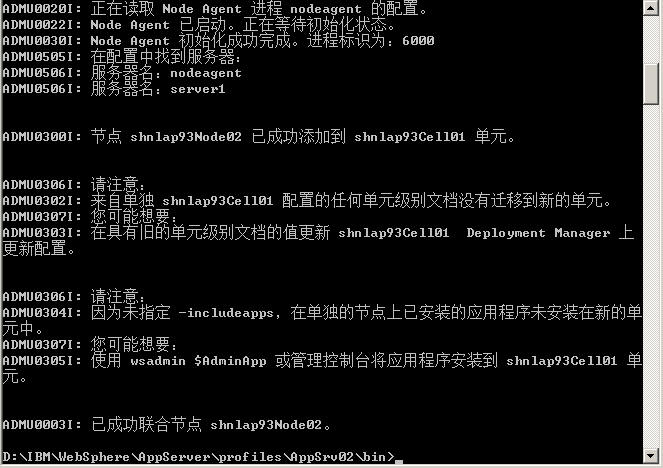
**WAS内的soap连接端口默认从8879开始计数，所以Dmgr的soap端口是8879，AppSrv01的soap端口就是8880，AppSrv02的soap端口就为8881，依次类推。**





## 4.2 联合AppSrv02节点入Dmgr



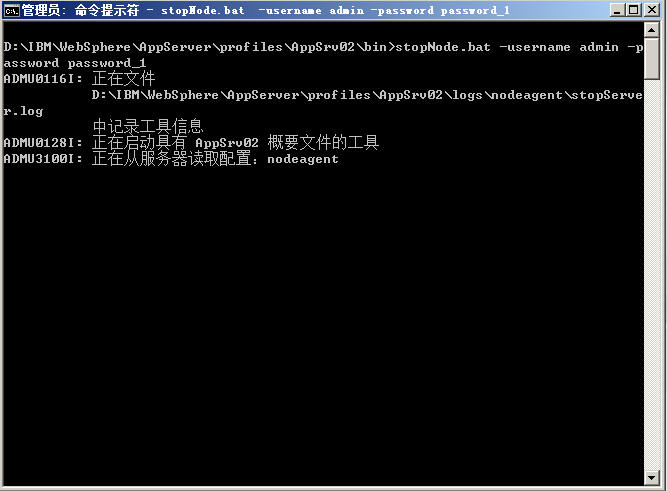


***当有任意一个“应用程序服务器”被联合入Dmgr后，该Dmgr即不再以server的形式运行了，而是以manager的形式来运行了，因此再关闭和启动该Dmgr时不能用startServer命令了，而是要用startManager, stopManager这对命令来控制该Dmgr的启动与关闭了。***

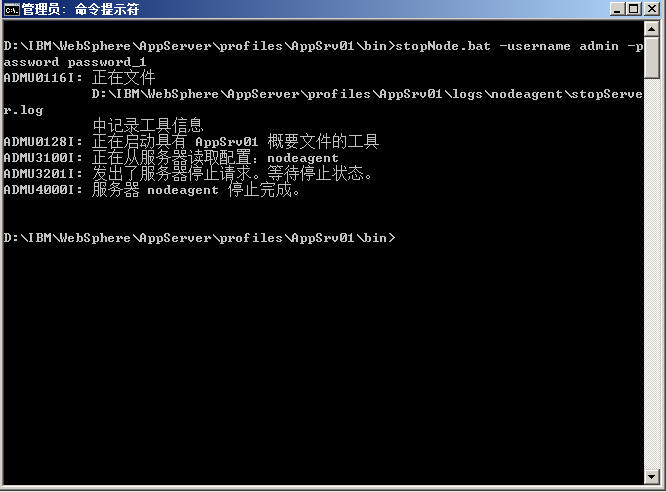
## 4.3 重新用集群的方式启动Dmgr，AppSrv01, AppSrv02

### 4.3.1 先依次关闭AppSrv02, AppSrv01, Dmgr

### 关闭AppSrv02



### 关闭AppSrv01



### 关闭Dmgr

***因为已经有任意一个“应用程序服务器”被联合入Dmgr后，该Dmgr即不再以server的形式运行了，而是以manager的形式来运行了，因此再关闭和启动该Dmgr时不能用startServer命令了，而是要用startManager, stopManager这对命令来控制该Dmgr的启动与关闭了。***

**所以我们使用stopManager而不再是stopServer命令了。**



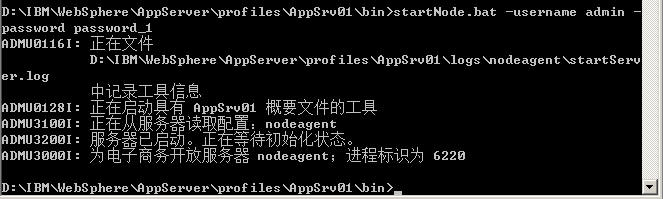
### 4.3.2 依次启动集群各节点Dmgr, AppSrv01, AppSrv02

### 先启动Dmgr

使用：D:\IBM\WebSphere\AppServer\profiles\Dmgr\bin\startManager –username admin –password password\_1来启动我们的Dmgr。

### 再启动AppSrv01

AppSrv01与AppSrv02的启动没有先后顺序，可以任意。



### 再启动AppSrv02



## 4.4 利用WAS自带的admin console组建集群

Dmgr, AppSrv01, AppSrv02都以集群的方式启动起来后我们还需要使用我们的admin console来把这些个节点真正组成我们想要的集群。

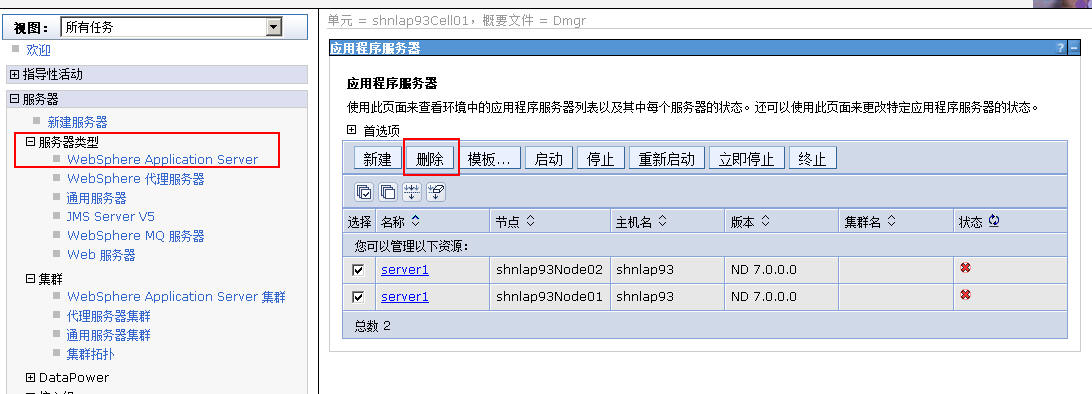
一旦集群组成后，我们就可以在admin console中start, stop我们的集群了。

**注：**

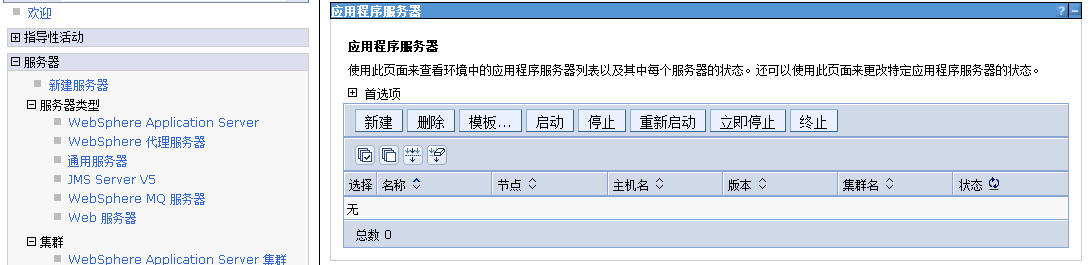
**要在admin console中使用GUI的方式start stop我们的集群必须先以startMangaer的方试启动我们的Dmgr，再以startNode的形式把各个节点启动起来后才能利用admin console中的GUI方式来控制集群。**

我们登录dmgr的admin console即使用：<http://localhost:9060/ibm/console>登录后

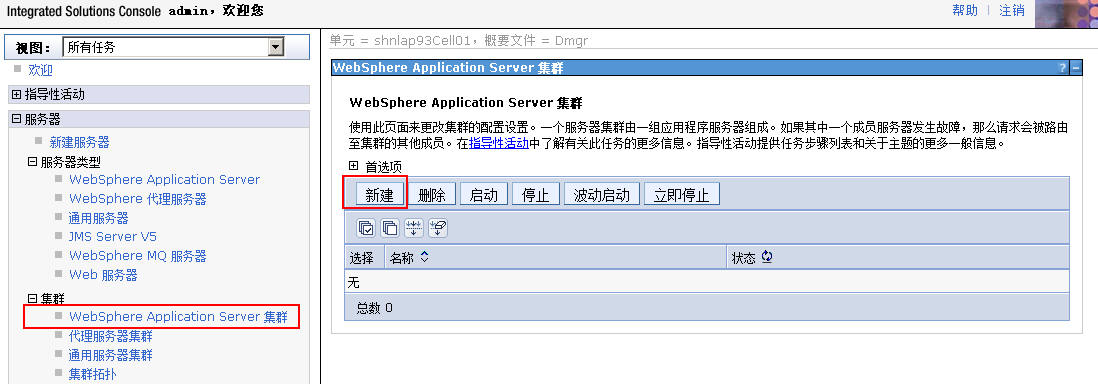
我们选择左边的服务器类型->WebSphere Application Server，在右边如下图我们可以看到现在列出2个server，全部选中后，点[删除]按钮（**对，你没看错，我们要的是集群不要单独运行的server）**。



全删空后的效果如下图



此时，我们选择左边菜单的“WebSphere Application Server集群”，然后在右边的操作面板中点[新建]按钮。



我们把我们这个集群名叫“mycluster”吧，按照下面的截图填选



下一步

此处，我们把AppSrv01这个profile所带的server1加入这个cluster中去。



下一步

**这步要注意了，很重要**

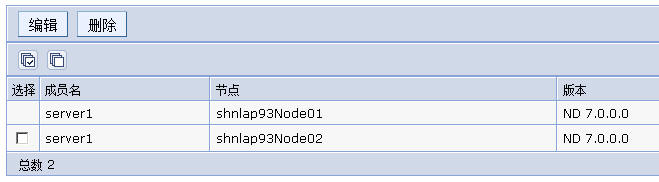
刚才我们将位于shnlap93Node01即AppSrv01中的server1加入了mycluster，现在我们要将AppSrv02中的server1也加入mycluster中。

记住在选择节点出选择“shnlap93Node02”，成员名还是添server1（不是server2哦）这边有点搞。



**然后不是点[下一步]，而是点那个[添加成员]这个按钮。**

点了这个[添加成员]按钮后我们会发觉在下方的成员列表中多了一项了，如下图所示。

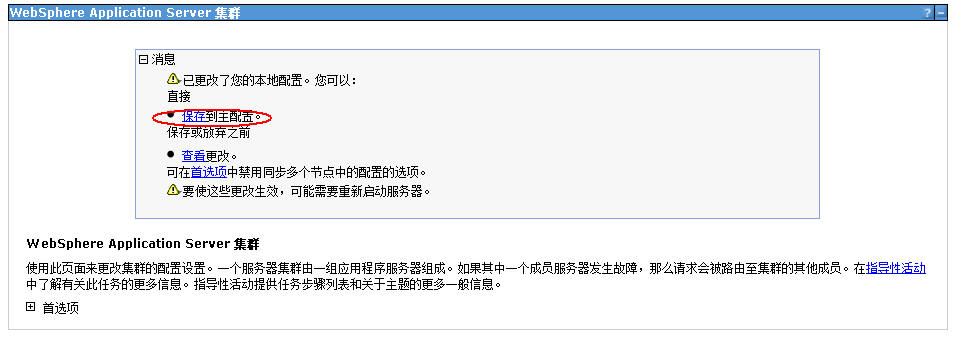


此时，再点[下一步]



我们确认信息后点[完成]，开始创建“mycluster”集群。

**创建完后千万不要忘记点“保存”这个连接啊，要不然前功尽弃。**



点完“保存”这个连接后系统将显示如下这样的消息输出界面



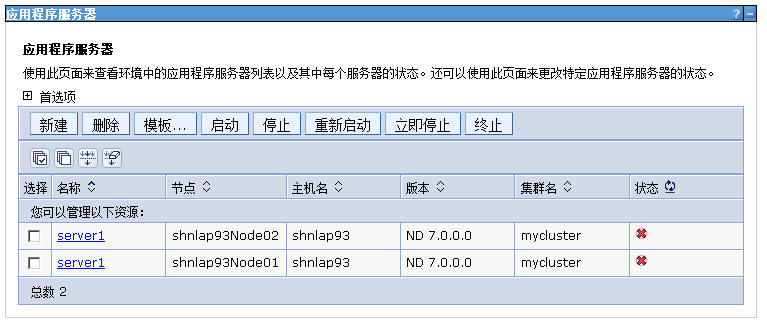
等到所有的消息输出完成后，再点[确定]按钮。

集群mycluster创建完成后将显示如下，此时，它的状态为不可用。



我们点这个“mycluster”，点进去，看到下面这样的界面。

可以看到这个集群包括两个节点以及节点的相关信息，并且这两个节点虽然都已经startNode了，但实质上相关的服务没有启动起来。



我们先不急着启动我们的集群，我们先来布署我们的JDBC吧。

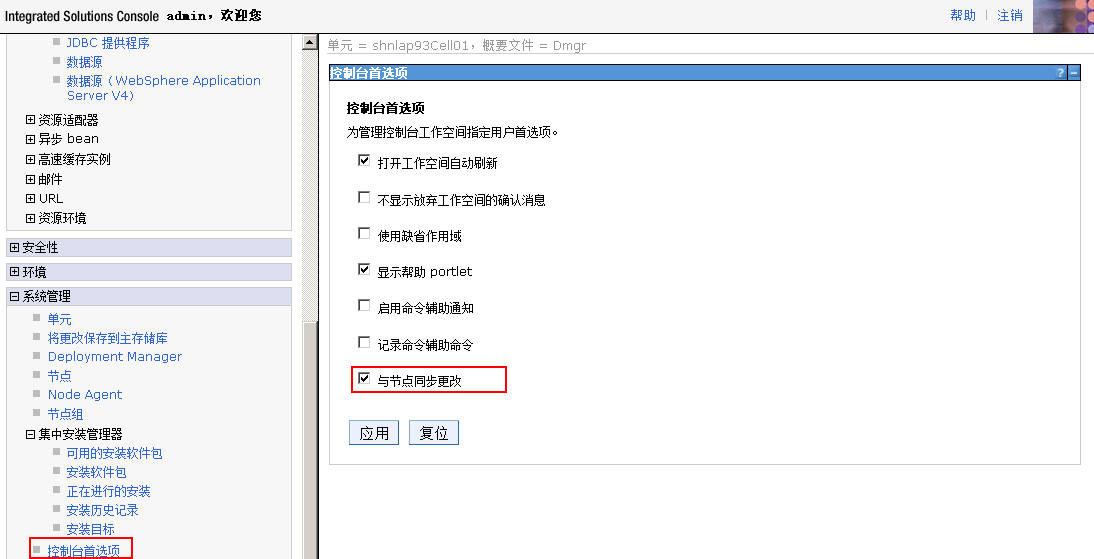
# 在WASND中布署JDBC

在创建JDBC前我们先作一个小操作。

因为我们现在是集群了，在一个单元的任何资源发布都会被同步到另一个相同的集群中的另一个节点，但是WASND默认是不会进行这个“节点间的同步”的，需要我们人工设置一下，按照下图：

系统管理->控制台首选项中，勾选这个“与节点同步更改”的选项，并点击[应用]。

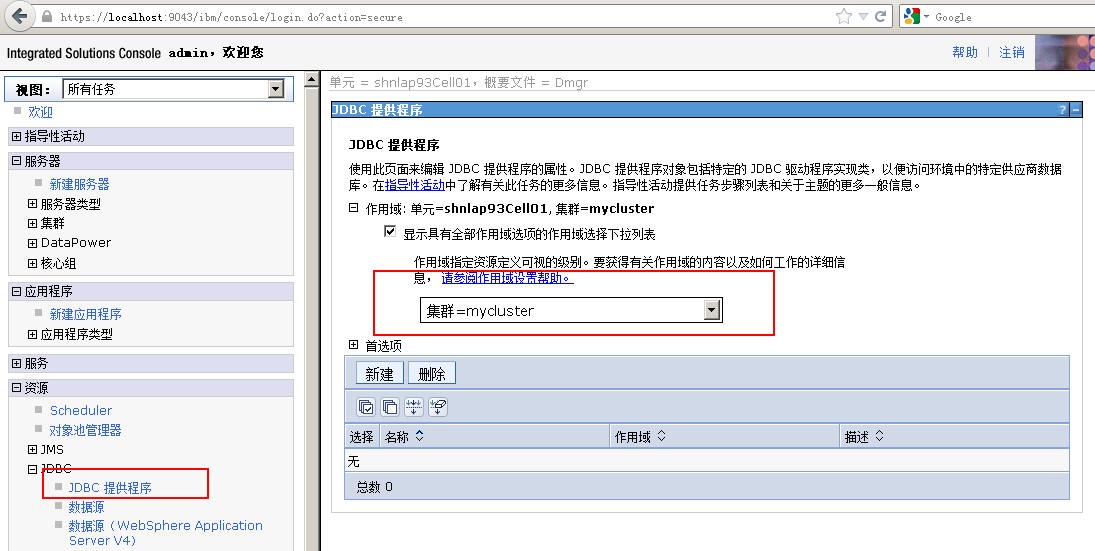
以后，无论什么资源发布，企业级应用程序发布，节点间的应用就都会自动“同步”了。





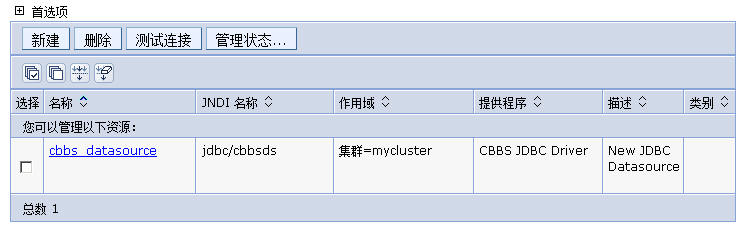
## 5.1 创建JDBC Provider

这个和我们在**（第十五天）IBM Websphere的安装与优化**中的**5.2**小节中的介绍一样，没有什么特殊的，只是在创建JDBC Provider时的作用域要选“mycluster”，而不能再是普通的Node了。



## 5.2 数据源

创建完我们的JDBC Provider后，我们还要创建我们的数据源，对吧？



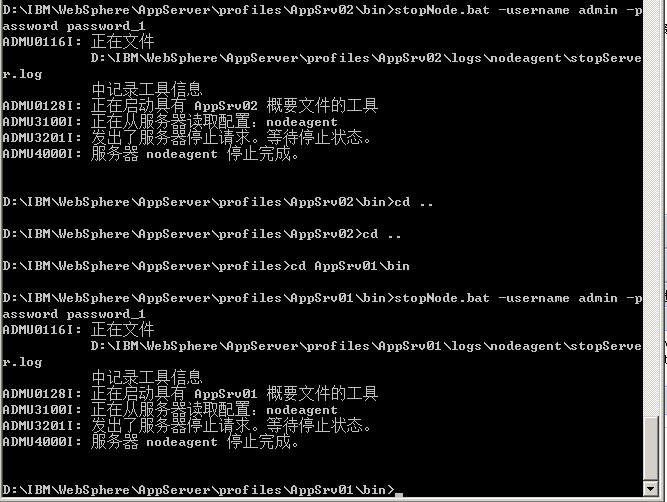
别忘了给该数据源指定一个JAAS-J2C认证数据-见**（第十五天）IBM Websphere的安装与优化**中的数据源的创建部分。

好，此时请不要急着点测试。

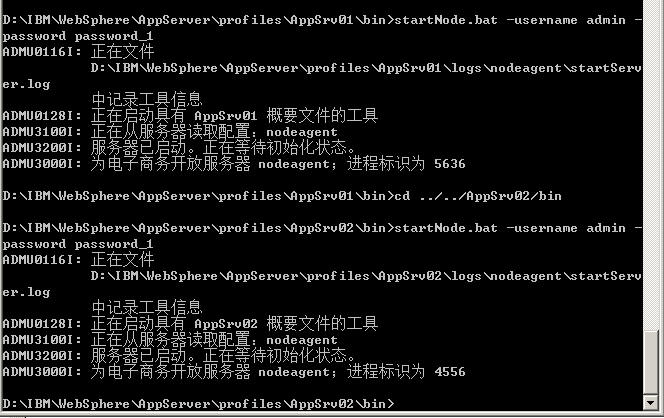
**WASND7中有个BUG，即有时发布完一个资源，并且同步到各个节点后并未马上生效，而是需要物理上重启各个“同步的节点”才会生效。**

因此，我们打开一个命令行：

**第一步：依次stop AppSrv01与AppSrv02**



**第二步：依次start AppSrv01与AppSrv02**



回到我们的admin console界面中。

1. 使用左边的菜单项中的“WebSphere Application Server集群”。
2. 勾选”mycluster”这个集群
3. 点[启动]按钮



我们可以不断的通过“状态”旁的刷新按钮来监控这个集群启动状态

**正在启动状态**



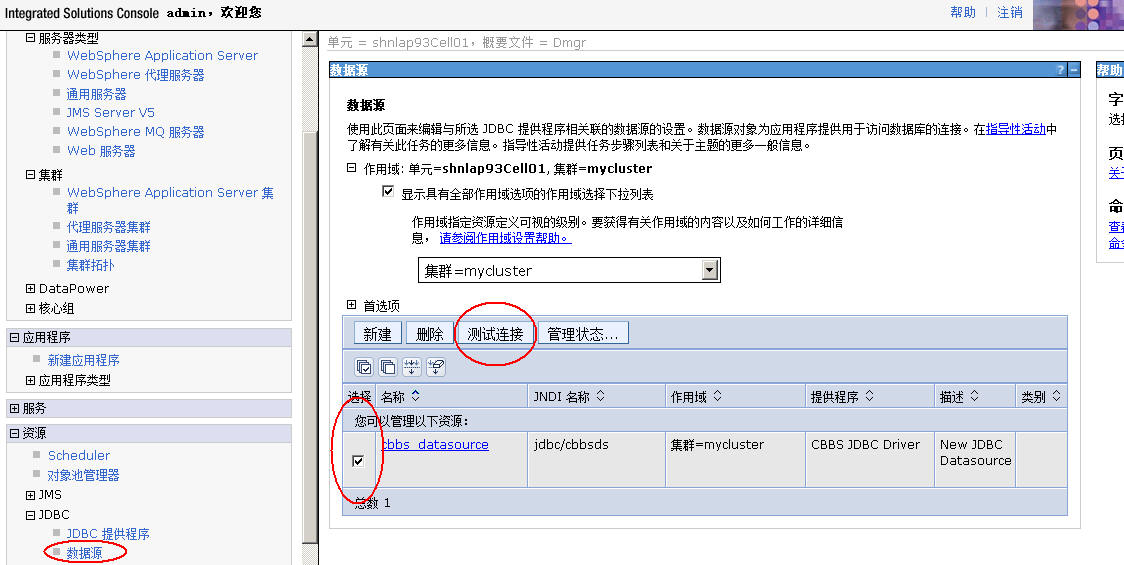
**启动成功状态**



启动成功后该集群mycluster中的各个节点的状态都应该为“绿色箭头”。



回到数据源主窗口，此时我们可以测试这个数据源的连接是否正常了。



如果数据源测试成功因该显示如下信息



# 在WebSphere集群环境中布署企业级应用

我们将使用在**（第十五天）IBM Websphere的安装与优化**中的第5.3小节的通过Ant脚本打包出来的EAR包来进行布署。

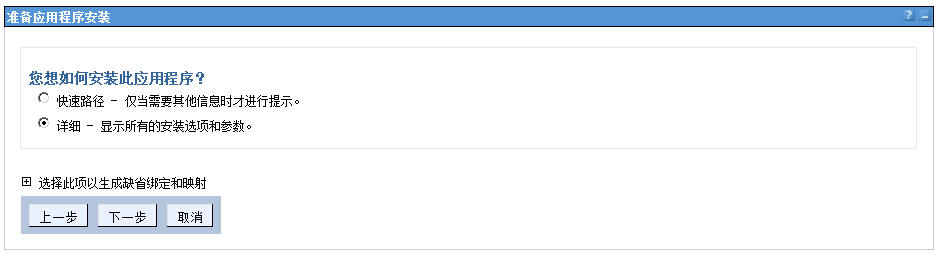
**应用程序->新建应用程序**



下一步



下一步

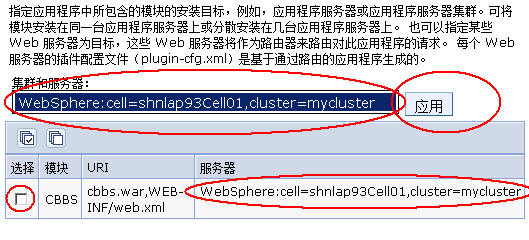


这边的作用域一定要按照如下图所选啊。

同时要确保“服务器”（即Target）这栏中显示的内容即有包含cell又有包含cluster。

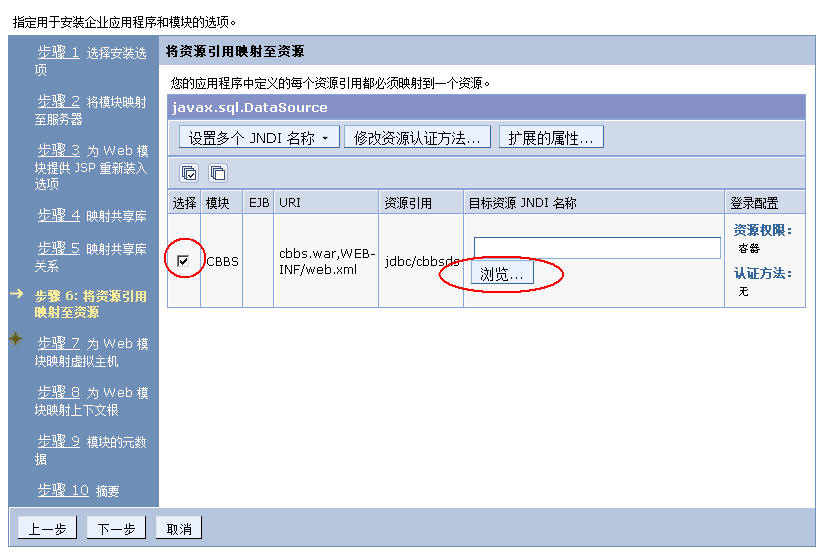
如果只包含cell那么：

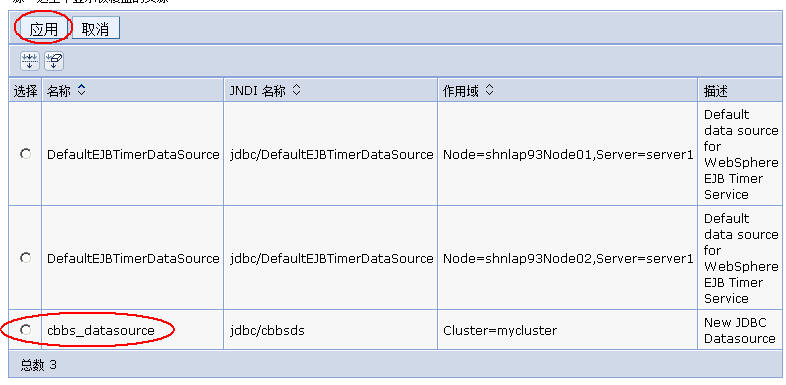
1. 请勾选这个“选择”前的checkbox
2. 然后在上部的“集群和服务器”中选择正确的即有cell又有cluster的下拉项，然后点应用，使得服务器（即Target）所显示的内容即有cell又有cluster



下一步

绑定工程中的数据源与WASND中刚才设置的数据源。







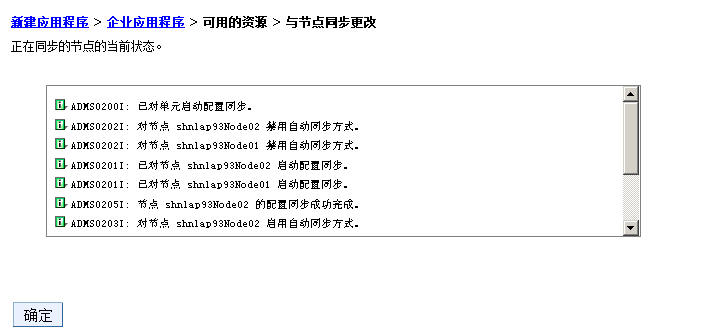
下一步



下一步下一步一直到出现最后确认信息的对话框，对[完成]按钮



别忘了点“保存”啊，再提醒一下



消息输出完成后点[确定]

一个企业级应用即ear包刚被布署到集群环境中，它的状态是不可用的，请按照如下图操作，使得该应用能够被启用。

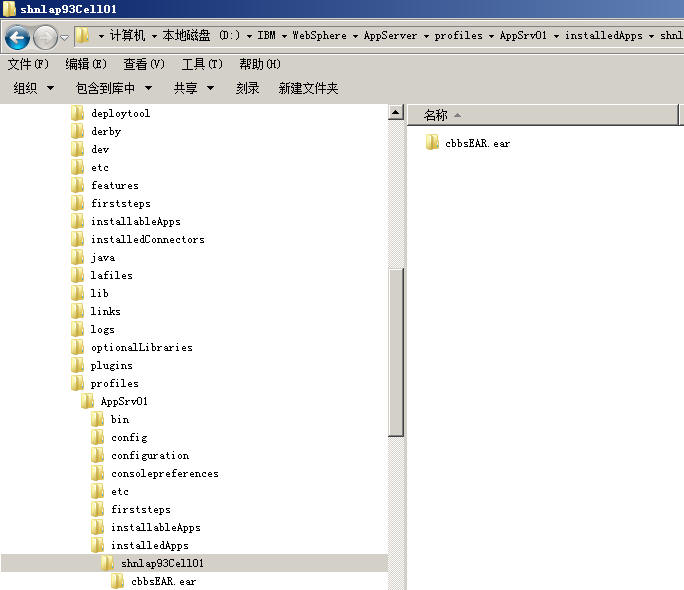


一个企业级应用启动成功后的显示界面



此时我们来看：

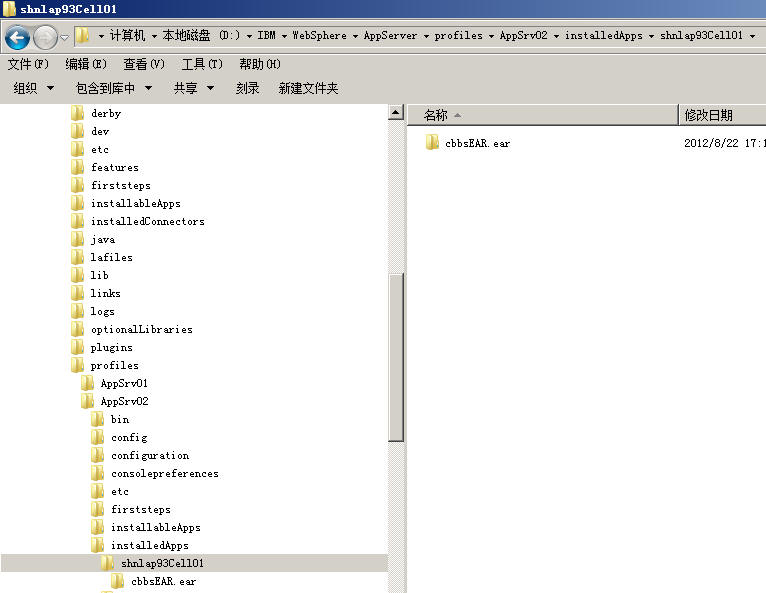
D:\IBM\WebSphere\AppServer\profiles\AppSrv01\installedApps\shnlap93Cell01目录下



看到没有，自动已经布署了这个ear。

那么集群间的节点是同步的，好！

我们再来看D:\IBM\WebSphere\AppServer\profiles\AppSrv02\installedApps\shnlap93Cell01目录下



看到没有，也已经被自动布署了这个ear文件了。

**集群上发布应用，节点间自动同步！！YEAH！！**

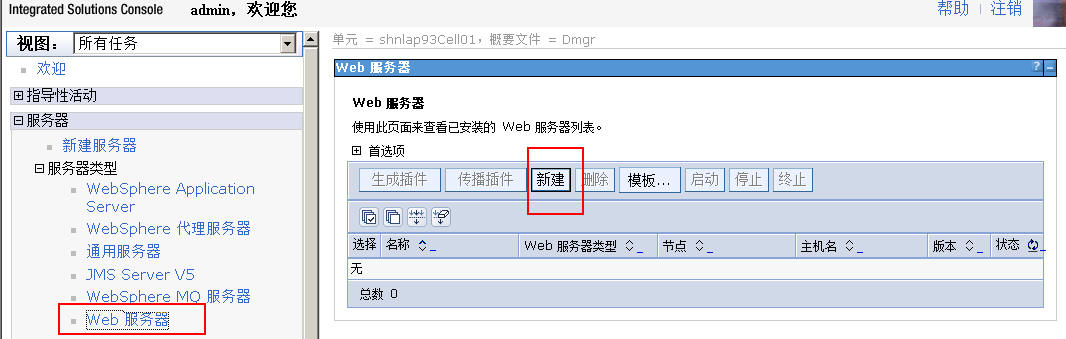
我们可以试着使用一个IE，一个FireFox分别使用：

<http://localhost:9080/cbbs>与<http://localhost:9081/cbbs> 这两个地址分别来试着登录我们的应用，如果两个地址的应用都能够被成功登录，那就说明我们的应用已经在集群环境中发布成功了。

# 整合IBM HTTPServer与WASND

我们应用已经在集群环境中发布成功了，下面我们来使用我们的IHS来集成我们的WASND吧，必尽在实际应用中我们对外只有一个网址即：<http://localhost/cbbs>，至于它背后有几个App Server对用户来说应该是“透明”的。

通过服务器->服务器类型->web服务器，我们在右边的操作面板中点[新建]



我们把我们这个web server命名为demoweb。

把它安装在节点：shnlap93Node01上吧（没关系，随便装哪个节点都可以，它会同步另一个节点的配置的）。



下一步



下一步下一步直到完成，别忘了点“保存”。





然后我们得到下面这个Web Server，请启动这个Web Server，怎么启动还记么？系统菜单栏中去启动它吧！！！

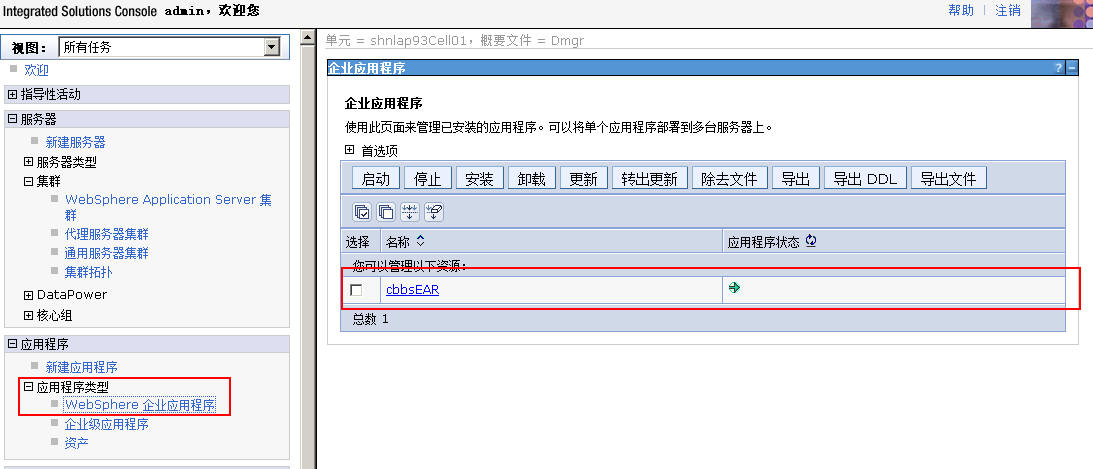


这是我们刚才建立的一个Web Server。



**确保我们的企业级应用程序布署的作用域即Target正确。**

应用程序->应用程序类型->WebSphere企业应用程序



用鼠标单击我们刚才布署的这个应用



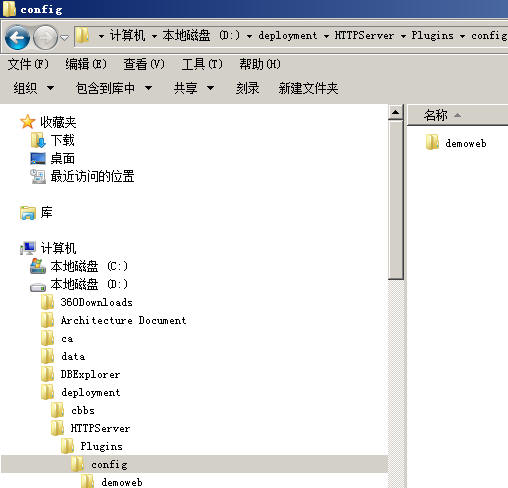
进入这个“管理模块”



确保我们的作用域中含有集群，在此我们这个集群为mycluster，要不然，你后一步生成的IHS的plugin的作用域只能够用来连接单个WebSphere节点与IHS，而不是整个集群+IHS的配置。

然后我们选中这个demoweb点[生成插件]，点一下这个[传播插件]

我们可以看到在D:\deployment\HTTPServer\Plugins\config目录下多了一个目录叫demoweb目录。



把这个**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\_mycluster\_URIs"> |

往下看，我们可以看到如下这些行：

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

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/\*"/> |

这个plugin.xml很像“**（第十六天）IBM Websphere与IBM HttpServer的集成**”中的那个plugin文件，对吧？

区别在于这边的几行语句，这个就是IBM HttpServer与WASND结合时的plugin描述语句。

|  |
| --- |
| <ServerCluster CloneSeparatorChange="false" GetDWLMTable="false" IgnoreAffinityRequests="true"  LoadBalance="Round Robin" Name="mycluster" PostBufferSize="64" PostSizeLimit="-1"  RemoveSpecialHeaders="true" RetryInterval="60">  <Server CloneID="1756mq3p8" ConnectTimeout="5" ExtendedHandshake="false" LoadBalanceWeight="2"  MaxConnections="-1" Name="shnlap93Node01\_server1" ServerIOTimeout="60"  WaitForContinue="false">  **<Transport Hostname="shnlap93" Port="9080" Protocol="http"/>**  **<Transport Hostname="shnlap93" Port="9443" Protocol="https">**  <Property Name="keyring" Value="D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-key.kdb"/>  <Property Name="stashfile" Value="D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-key.sth"/>  </Transport>  </Server>  <Server CloneID="1756mq6cs" ConnectTimeout="5" ExtendedHandshake="false" LoadBalanceWeight="2"  MaxConnections="-1" Name="shnlap93Node02\_server1" ServerIOTimeout="60"  WaitForContinue="false">  **<Transport Hostname="shnlap93" Port="9081" Protocol="http"/>**  **<Transport Hostname="shnlap93" Port="9444" Protocol="https">**  <Property Name="keyring" Value="D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-key.kdb"/>  <Property Name="stashfile" Value="D:\IBM\HTTPServer\Plugins\config\demoweb\plugin-key.sth"/>  </Transport>  </Server>  <PrimaryServers>  **<Server Name="shnlap93Node01\_server1"/>**  **<Server Name="shnlap93Node02\_server1"/>**  </PrimaryServers>  </ServerCluster> |

## 第三步：

打开D:\IBM\HTTPServer\conf目录下的httpd.conf文件，最后几行确保是以下内容：

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

## 第四步：

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：

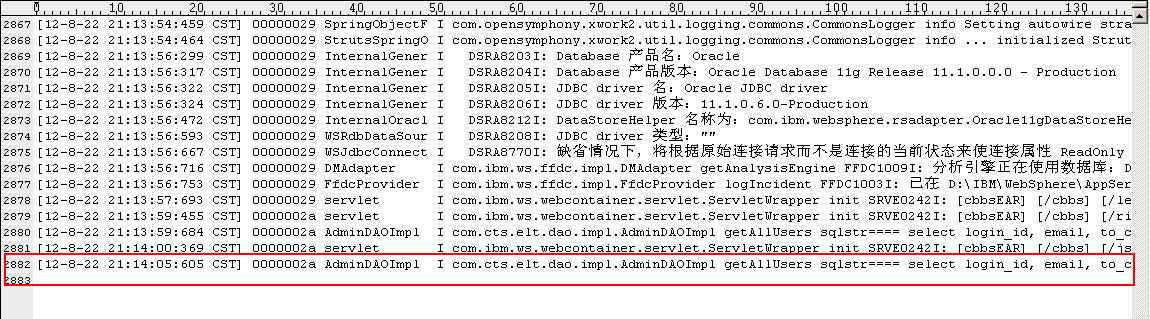


然后我们再打开一个firefox，输入: <http://localhost/cbbs/>，得到我们的应用的登录界面，输入另一个用户名与密码登录后，进行一些操作，一切无误：



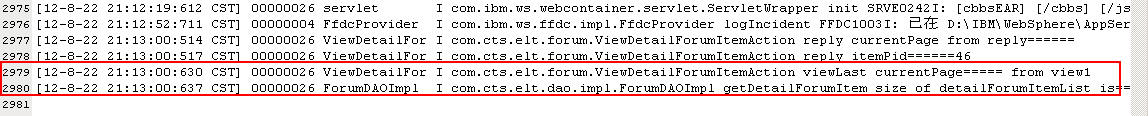
我们看：

D:\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1目录下的SystemOut.log文件



再看：

D:\IBM\WebSphere\AppServer\profiles\AppSrv02\logs\server1目录下的SystemOut.log文件



这说明我们我们通过两个不同的浏览器使用两个用户在访问集群环境时：

一个用户名被分流到的AppSrv01中去；

另一个用户在登录时则被分流到了AppSrv02中去；

WebSphere的集群+IBM HTTPServer集成成功，结束今天的教程。