第三十一章：Tomcat应用部署

**一、初识Tomcat；**

**二、Tomcat与jdk的关系；**

**三、Tomcat内部工作原理；**

**四、Tomcat配置文件详解；**

**五、案例：搭建Tomcat案例；**

**一、初识tomcat；**

**概述：**Tomcat是由Apache软件基金会下属的Jakarta项目开发的一个Servlet容器，按照Sun Microsystems提供的技术规范，实现了对Servlet和JavaServer Page（JSP）的支持，并提供了作为Web服务器的一些特有功能，如Tomcat管理和控制平台、安全域管理和Tomcat阀等。由于Tomcat本身也内含了一个HTTP服务器，它也可以被视作一个单独的Web服务器。

**便于理解：Apache与Tomcat的比较**

**相同点：**

1.两者都是Apache组织开发的

2.两者都有HTTP服务的功能

3.两者都是免费的

**不同点：**

1.Apache是专门用了提供HTTP服务的，以及相关配置的（例如虚拟主机、URL转发等等），而Tomcat是Apache组织在符合Java EE的JSP、Servlet标准下开发的一个JSP服务器.

2.Apache是一个Web服务器环境程序，启用他可以作为Web服务器使用，不过只支持静态网页。如果要在Apache环境下运行JSP的话就需要一个解释器来执行JSP网页，而这个JSP解释器就是Tomcat。

3.Apache侧重于HTTPServer ，Tomcat侧重于Servlet引擎，如果以Standalone方式运行，功能上与Apache等效，支持JSP，但对静态网页不太理想；

4.Apache是Web服务器，Tomcat是应用（Java）服务器，它只是一个Servlet（JSP也翻译成Servlet）容器，可以认为是Apache的扩展，但是可以独立于Apache运行。

**Tomcat同类产品：**

1.Resin 服务器：

Resin是Caucho公司的产品，是一个非常流行的支持Servlet和JSP的服务器，速度非常快。Resin本身包含了一个支持HTML的Web服务器，这使它不仅可以显示动态内容，而且显示静态内容的能力也毫不逊色，因此许多网站都是使用Resin服务器构建。

2.Jetty 服务器：

Jetty是一个纯粹的基于Java的网页服务器和Java Servlet容器。尽管网页服务器通常用来为人们呈现文档，但是Jetty通常在较大的软件框架中用于计算机与计算机之间的通信。Jetty作为Eclipse基金会的一部分，是一个自由和开源项目。

3.JBoss服务器：

JBoss是一个种遵从JavaEE规范的、开放源代码的、纯Java的EJB服务器，对于J2EE有很好的支持。JBoss采用JML API实现软件模块的集成与管理，其核心服务又是提供EJB服务器，不包含Servlet和JSP的Web容器，不过它可以和Tomcat完美结合。

4.WebSphere 服务器：

WebSphere是IBM公司的产品，可进一步细分为 WebSphere Performance Pack、Cache Manager 和WebSphere Application Server等系列，其中WebSphere Application Server 是基于Java 的应用环境，可以运行于 Sun Solaris、Windows NT 等多种操作系统平台，用于建立、部署和管理Internet和Intranet Web应用程序。

5.WebLogic 服务器：

WebLogic 是ORACLE公司的产品，可进一步细分为 WebLogic Server、WebLogic Enterprise 和 WebLogic Portal 等系列，其中 WebLogic Server 的功能特别强大。WebLogic 支持企业级的、多层次的和完全分布式的Web应用，并且服务器的配置简单、界面友好。对于那些正在寻求能够提供Java平台所拥有的一切应用服务器的用户来说，WebLogic是一个十分理想的选择。

**二、Tomcat与jdk的关系；**

**图解：**



**解析：**

JDK是开发人员开发使用，jre是用来运行java程序，就是java程序的运行环境，jvm是整个java实现跨平台的最核心的部分，所有的java程序会首先被编译为.class的类文件，这种类文件可以在虚拟机上执行；

JDK中包含JRE，在JDK的安装目录下有一个名为jre的目录，里面有两个文件夹bin和lib，在这里可以认为bin里的就是jvm，lib中则是jvm工作所需要的类库，而jvm和 lib和起来就称为jre。jdk是java语言编写的，用来给java程序调用的，jvm因为与平台进行交互，是用c/c++和汇编语言编写的；

jdk是java运行和编译的，tomcat是web项目的运行容器，java没有jdk没办法编译运行，java运行必须依赖jdk的环境，所以安装tomcat前必须要安装jdk环境；

**下载位置：**

tomcat下载地址：https://tomcat.apache.org/download-90.cgi

jdk下载地址：

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

注解：

1. Java SE（Java Platform，Standard Edition）：Java SE 是做电脑上运行的软件；

2. Java EE（Java Platform，Enterprise Edition）：Java EE 是用来做网站的（我们常见的JSP技术）；

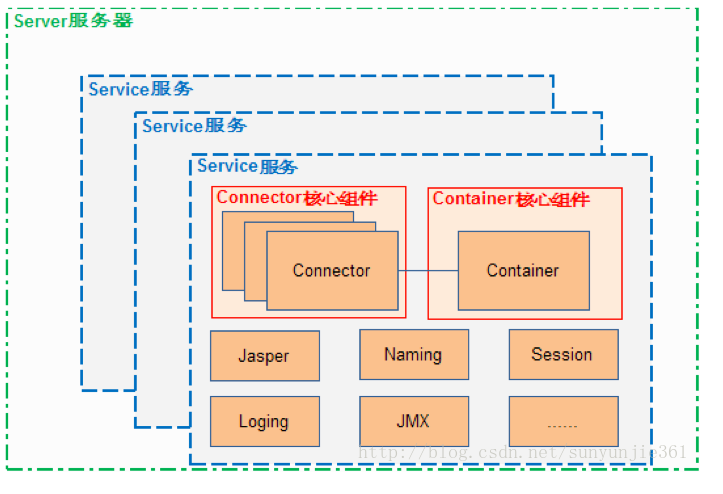
3. Java ME（Java Platform，Micro Edition）：Java ME 是做手机软件的；

**对应关系：**



**三、Tomcat内部工作原理；**

**tomcat整体架构：**

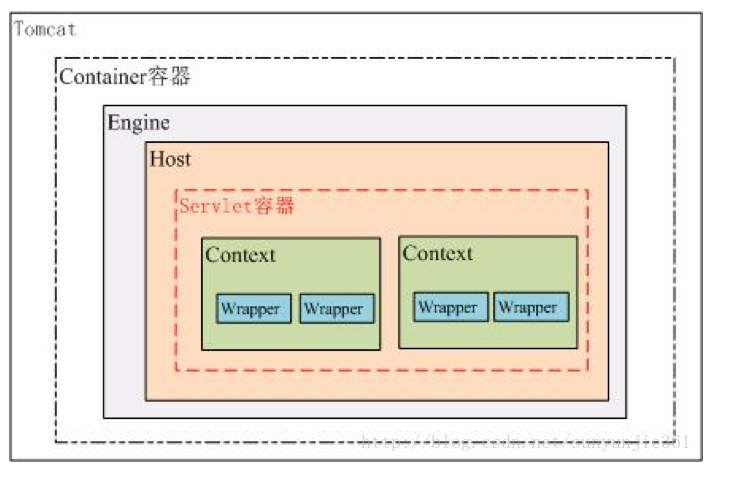


**组件关联：**

1、Tomcat 的心脏是两个组件：Connector 和 Container，Connector 组件是可以被替换，这样显得更加灵活，一个 Container 可以选择对应多个 Connector。多个 Connector 和一个 Container 就形成了一个 Service，有了 Service 就可以对外提供服务了，但是 Service 还要一个生存的环境，必须要有人能够给她生命、掌握其生死大权，那就非 Server 莫属了。所以整个 Tomcat 的生命周期由 Server 控制。

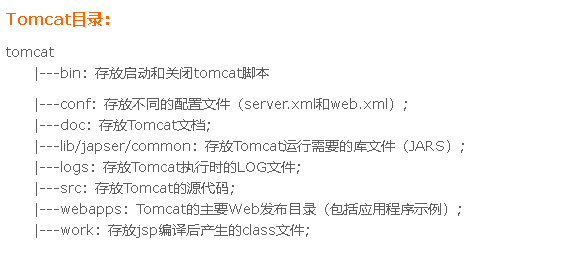
2、我们将 Tomcat 中 Connector、Container 作为一个整体比作一对情侣的话，Connector 主要负责对外交流，可以比作为 Boy，Container 主要处理 Connector 接受的请求，主要是处理内部事务，可以比作为 Girl。那么这个 Service 就是连接这对男女的结婚证了。是Service 将它们连接在一起，共同组成一个家庭。当然要组成一个家庭还要很多其它的元素说白了，Service 只是在 Connector 和 Container 外面多包一层，把它们组装在一起，向外面提供服务，一个 Service 可以设置多个 Connector，但是只能有一个 Container 容器。

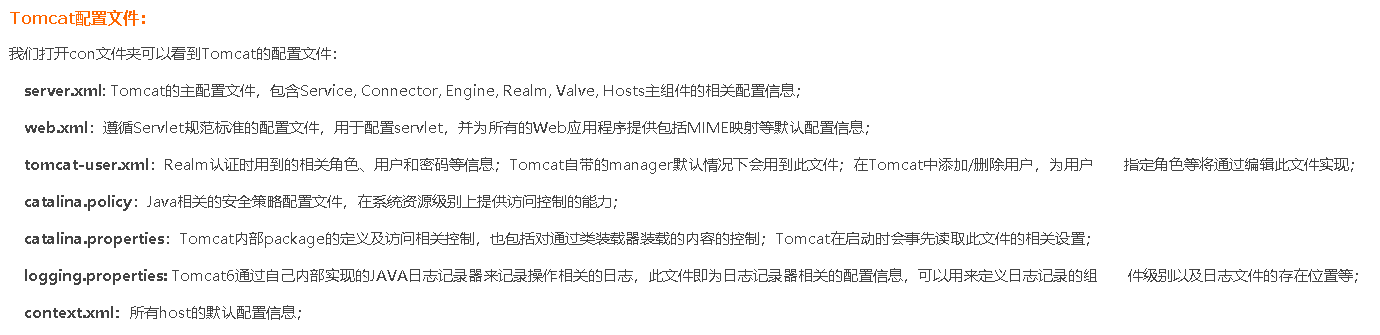
**Container组件详解：**



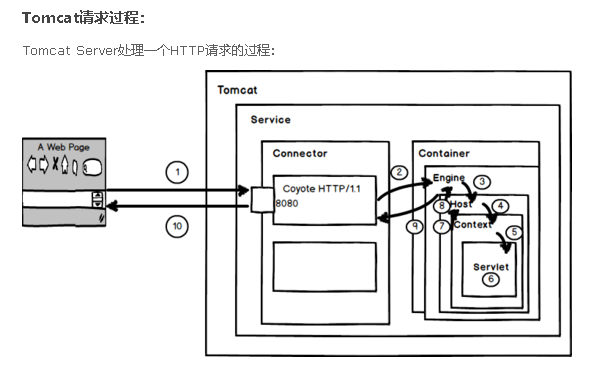
* Engine：负责处理所有请求，处理后，将结果返回给service，而connector是作为service与engine的中间媒介出现的。 作为交流者；一个engine下可以配置一个默认主机，每个虚拟主机都有一个域名。当engine获得一个请求时，它把该请求匹配到虚拟主机host上，然后把请求交给该主机来处理。 Engine有一个默认主机，当请求无法匹配到任何一个虚拟主机时，将交给默认host来处理。Engine以线程的方式启动Host。
* Host：代表一个虚拟主机，每个虚拟主机和某个网络域名（Domain Name）相匹配。 每个虚拟主机下都可以部署一个或多个web应用，每个web应用对应于一个context，有一个context path。 当Host获得一个请求时，将把该请求匹配到某个Context上；
* Wrapper： 代表一个 Servlet，它负责管理一个 Servlet，包括的 Servlet 的装载、初始化、执行以及资源回收。Wrapper 是最底层的容器，它没有子容器了；

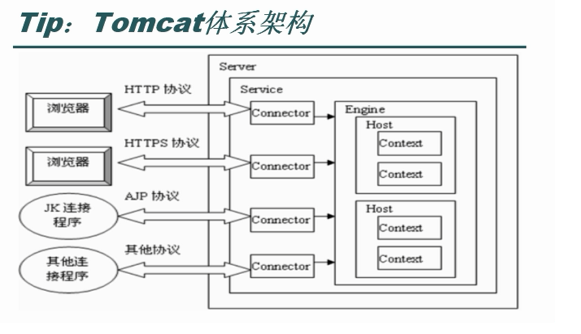
**四、Tomcat配置文件详解；**







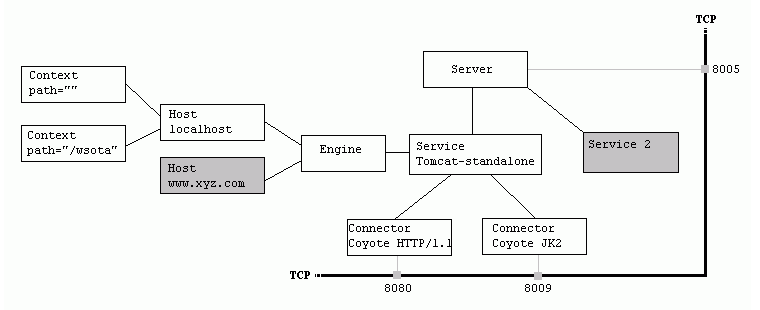




tomcat中支持两种协议的连接器：HTTP/1.1与AJP/1.3

HTTP/1.1协议负责建立HTTP连接，web应用通过浏览器访问tomcat服务器用的就是这个连接器，默认监听的是8080端口；

AJP/1.3协议负责和其他HTTP服务器建立连接，监听的是8009端口，比如tomcat和apache或者iis集成时需要用到这个连接器。

扩展：tomcat架构以及详细配置如下；

https://www.cnblogs.com/hggen/p/6264475.html

https://blog.csdn.net/skp127/article/details/52026150

**五、案例：搭建Tomcat案例；**

**案例环境：**

|  |  |  |  |
| --- | --- | --- | --- |
| 系统类型 | IP地址 | 主机名 | 所需软件 |
| Centos 7.4 1708 64bit | 192.168.100.101 | www.linuxfan.cn | apache-tomcat-9.0.10.tar.gz、 jdk-8u171-linux-x64.tar.gz |

**案例步骤：**

* 安装jdk，部署java环境；
* 安装tomact软件程序；
* 客户端访问测试初始页面；
* Tomcat服务配置目录讲解，配置文件讲解，主配置文件讲解；
* 安装mariadb数据库，提供数据库服务；
* 配置Tomcat的manager管理界面；
* 客户端访问测试manager管理界面；
* 项目一：云海管理平台，使用tomcat的Manager管理系统上线；
* 配置数据库，导入项目的数据库文件并且进行授权；
* 客户端访问测试项目一：
* 项目二：超市订单管理系统，手动上线，创建项目中的数据库或者导入数据库文件，并且进行授权；
* 超市订单管理系统项目上线；
* 客户端访问测试项目二；
* 自定义jsp页面，并测试访问；
* 部署tomcat的虚拟主机，客户端访问测试；
* **安装jdk，部署java环境；**

[root@www ~]# ls

apache-tomcat-9.0.10.tar.gz jdk-8u171-linux-x64.tar.gz

[root@www~]# rpm -qa |grep java

[root@www ~]# tar zxvf jdk-8u171-linux-x64.tar.gz

[root@www ~]# mv jdk1.8.0\_171/ /usr/local/java

[root@www ~]# ls /usr/local/java

bin db javafx-src.zip lib man release THIRDPARTYLICENSEREADME-JAVAFX.txt

COPYRIGHT include jre LICENSE README.html src.zip THIRDPARTYLICENSEREADME.txt

[root@www ~]# cat <<END >>/etc/profile

export JAVA\_HOME=/usr/local/java

export PATH=$PATH:/usr/local/java/bin

END

[root@www~]# source /etc/profile

[root@www ~]# java -version

java version "1.8.0\_171"

Java(TM) SE Runtime Environment (build 1.8.0\_171-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.171-b11, mixed mode)

* **安装tomact软件程序；**

[root@www ~]# tar zxvf apache-tomcat-9.0.10.tar.gz

[root@www ~]# mv apache-tomcat-9.0.10 /usr/local/tomcat

[root@www ~]# ls /usr/local/tomcat

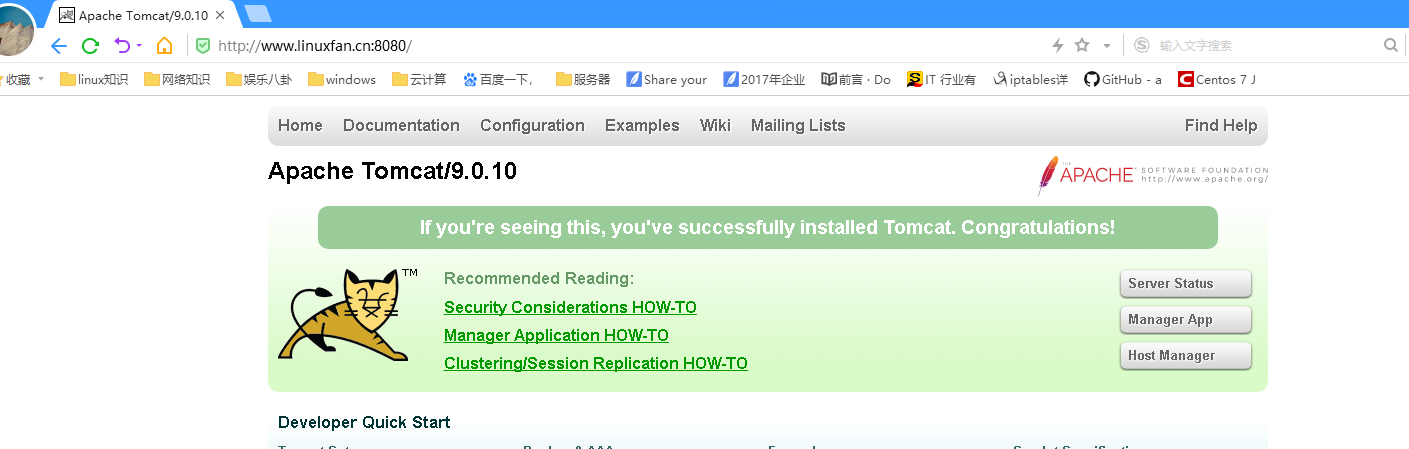
bin conf lib LICENSE logs NOTICE RELEASE-NOTES RUNNING.txt temp webapps work

[root@www ~]# /usr/local/tomcat/bin/startup.sh ##启动apache-tomcat

[root@www ~]# netstat -utpln |grep 8080

tcp 0 0 0.0.0.0:8080 0.0.0.0:\* LISTEN 14758/java

* **客户端访问测试初始页面；**



* **Tomcat服务配置目录讲解，配置文件讲解，主配置文件讲解；**
* **安装mariadb数据库，提供数据库服务；**

[root@www ~]# yum -y install mariadb-server mysql

[root@www ~]# systemctl start mariadb

[root@www ~]# systemctl enable mariadb

[root@www ~]# mysqladmin -uroot password 123123

[root@www ~]# mysql -uroot -p123123

MariaDB [(none)]> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| test |

+--------------------+

4 rows in set (0.00 sec)

MariaDB [(none)]> exit

* **配置Tomcat的manager管理界面；**

[root@www ~]# vi /usr/local/tomcat/conf/tomcat-users.xml ##添加管理页面的用户认证，帐户名为tomcat，密码为tomcat；

44 <role rolename="manager-gui"/>

45 <user username="tomcat" password="tomcat" roles="manager-gui"/>

46 </tomcat-users>

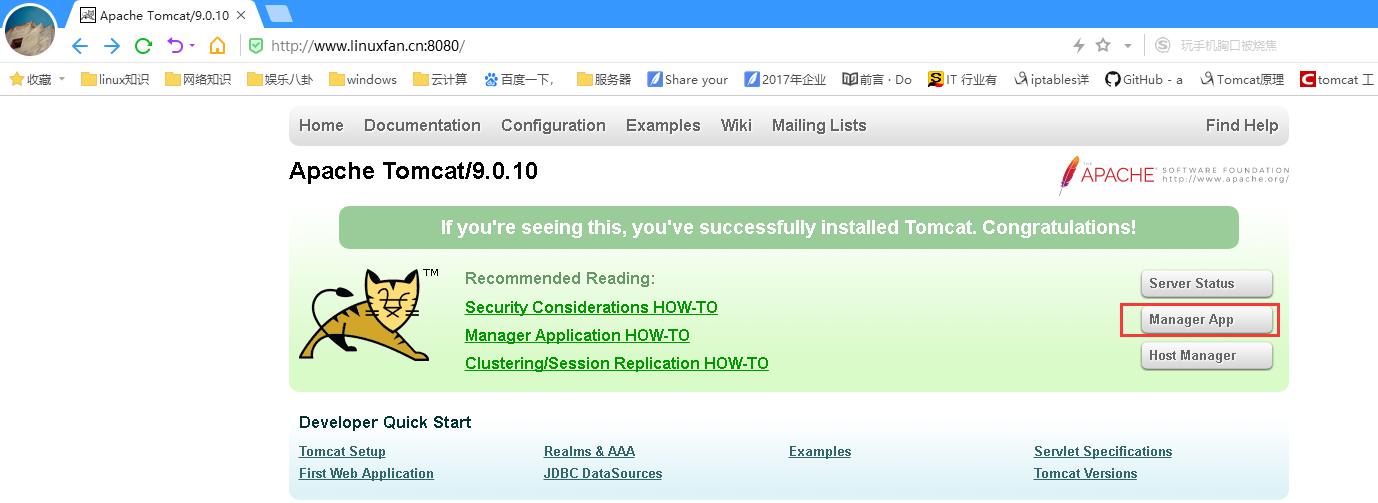
[root@www ~]# vi /usr/local/tomcat/webapps/manager/META-INF/context.xml ##默认tomcat的管理页面只能通过tomcat服务器上的浏览器才可以访问，设置其可以在其他主机访问；

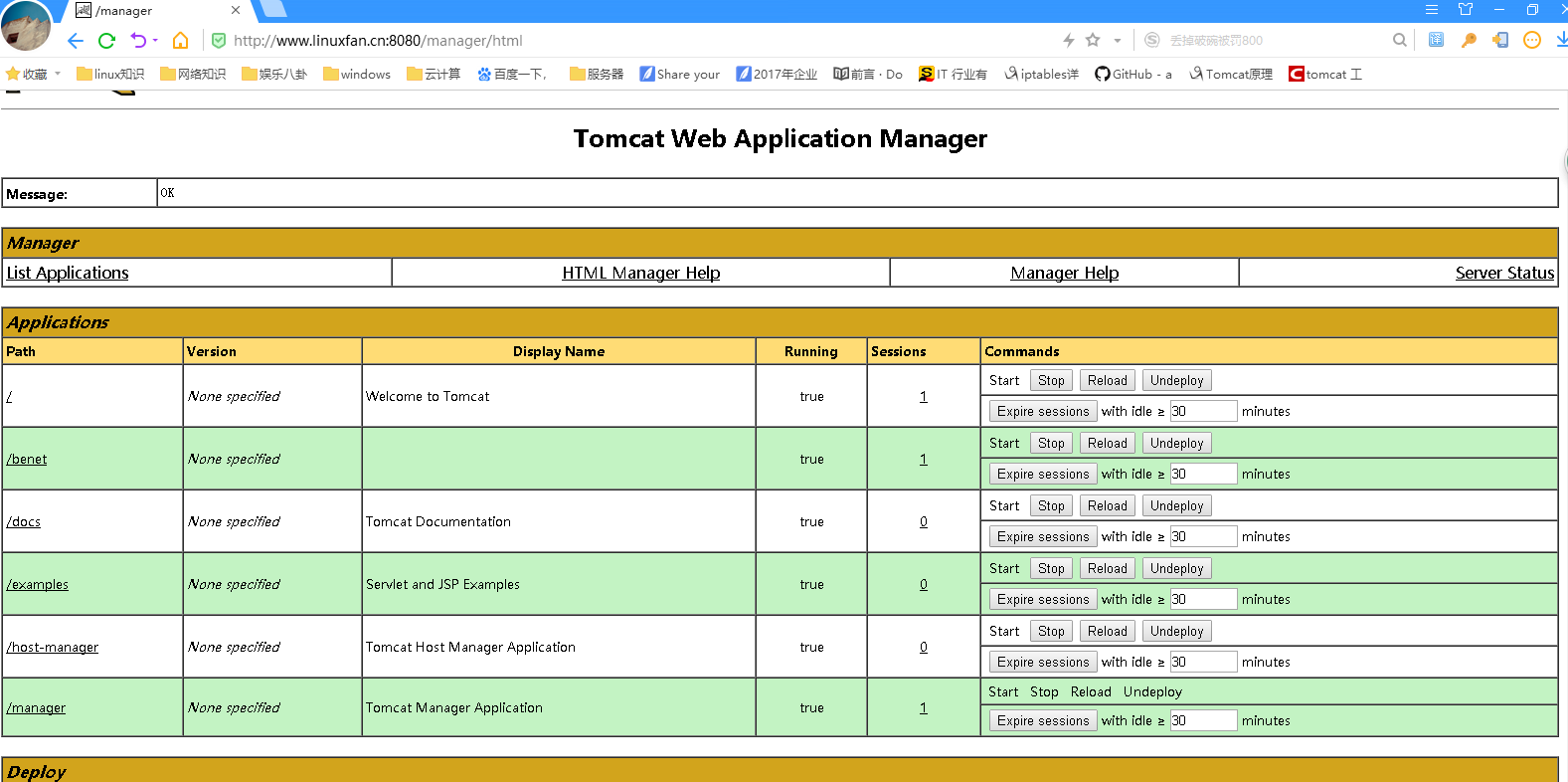
20 allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1|\d+\.\d+\.\d+\.\d+" />

[root@www ~]# /usr/local/tomcat/bin/shutdown.sh

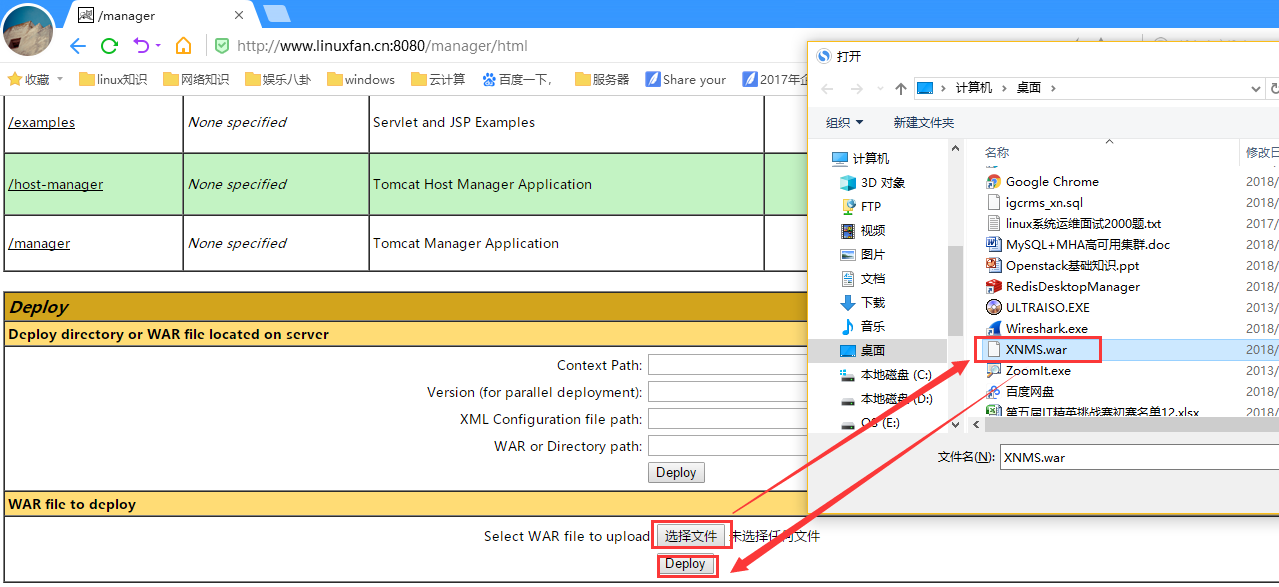
[root@www ~]# /usr/local/tomcat/bin/startup.sh

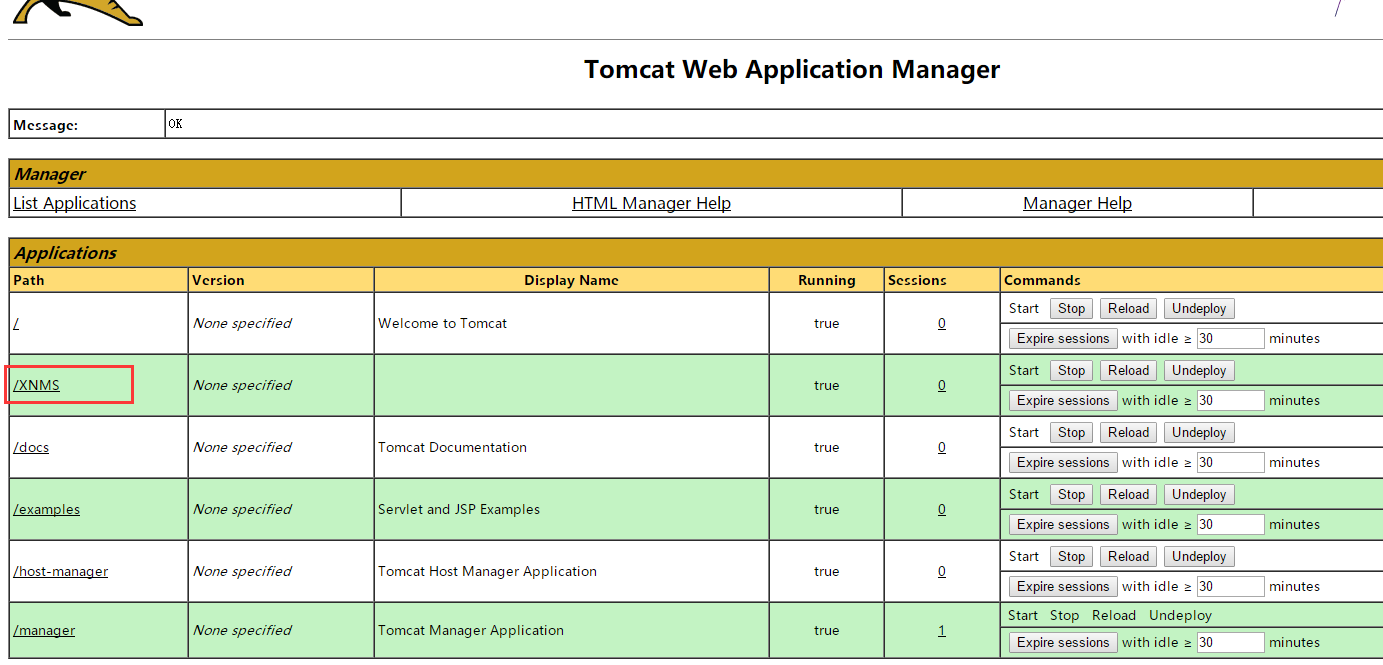
* **客户端访问测试manager管理界面；（客户端测试启动停止项目、测试服务端状态查看）：**





* **项目一：云海管理平台，使用tomcat的Manager管理系统上线；**

****



[root@www ~]# ls /usr/local/tomcat/webapps/

docs examples host-manager manager ROOT XNMS XNMS.war

[root@www ~]# vi /usr/local/tomcat/webapps/XNMS/WEB-INF/classes/jdbc.properties

6 url=jdbc:mysql://192.168.100.101:3306/igcrms\_xn?useUnicode=true&characterEncoding=utf-8&amp;allo wMultiQueries=true

7 username=linuxfan

8 password=123123

* **配置数据库，导入项目的数据库文件并且进行授权；**

[root@www ~]# ls igcrms\_xn.sql

igcrms\_xn.sql

[root@www ~]# mysql -uroot -p123123<igcrms\_xn.sql

[root@www ~]# mysql -uroot -p123123

MariaDB [(none)]> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| igcrms\_xn |

| mysql |

| performance\_schema |

| test |

+--------------------+

MariaDB [(none)]> grant all on igcrms\_xn.\* to 'linuxfan'@'192.168.100.101' identified by "123123";

MariaDB [(none)]> flush privileges;

MariaDB [(none)]> use igcrms\_xn

MariaDB [igcrms\_xn]> show tables;

...

| sys\_user\_info |

+-----------------------------+

MariaDB [igcrms\_xn]> update sys\_user\_info set password=password("123123") where name="irms";

MariaDB [igcrms\_xn]> select name,password from sys\_user\_info;

+--------+-------------------------------------------+

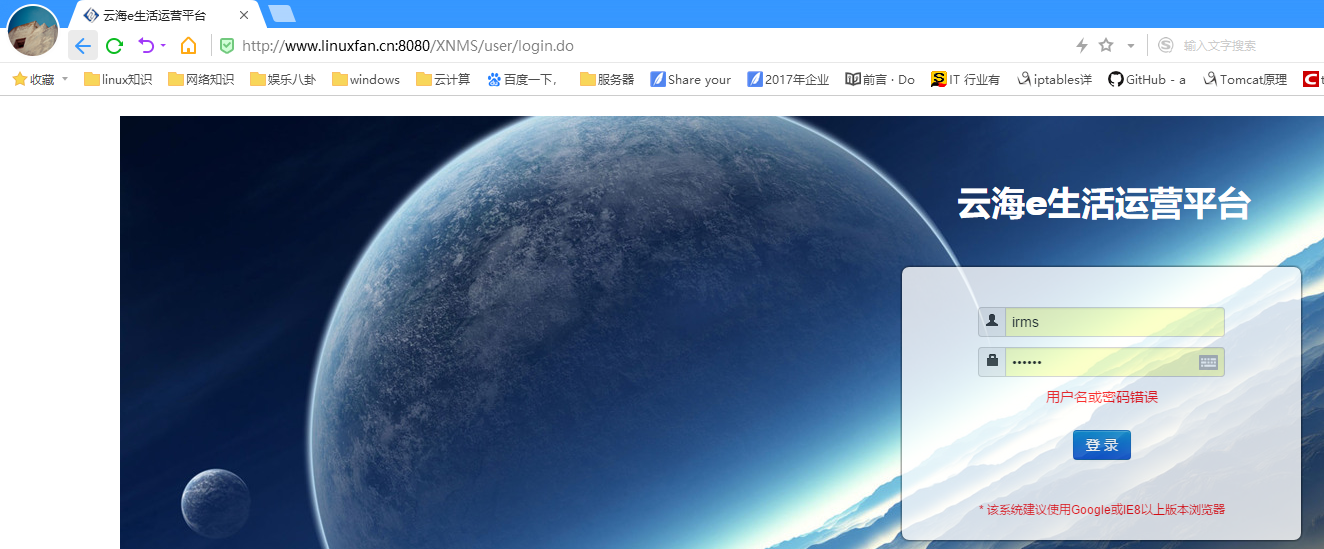
| name | password |

+--------+-------------------------------------------+

| irms | \*E56A114692FE0DE073F9A1DD68A00EEB9703F3F1 |

MariaDB [igcrms\_xn]> exit

* **客户端访问测试项目一：**



* **项目二：超市订单管理系统，手动上线，创建项目中的数据库或者导入数据库文件，并且进行授权；**

[root@www ~]# ls smbms\_db.sql

smbms\_db.sql

[root@www ~]# mysql -uroot -p123123<smbms\_db.sql

[root@www ~]# mysql -uroot -p123123

MariaDB [(none)]> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| igcrms\_xn |

| smbms |

| performance\_schema |

| test |

+--------------------+

MariaDB [(none)]> grant all on smbms.\* to 'linuxfan'@'192.168.100.101' identified by "123123";

Query OK, 0 rows affected (0.00 sec)

##为项目中所指定的数据库进行授权

mysql> flush privileges;

Query OK, 0 rows affected (0.01 sec)

mysql> use smbms;

Database changed

mysql> show tables;

+-----------------+

| Tables\_in\_smbms |

+-----------------+

| smbms\_address |

| smbms\_bill |

| smbms\_provider |

| smbms\_role |

| smbms\_user |

+-----------------+

5 rows in set (0.00 sec)

mysql> select \* from smbms\_user; ##查看项目中的管理员账户

...

| 1 | admin | ????? | 1234567 | 1 |

...

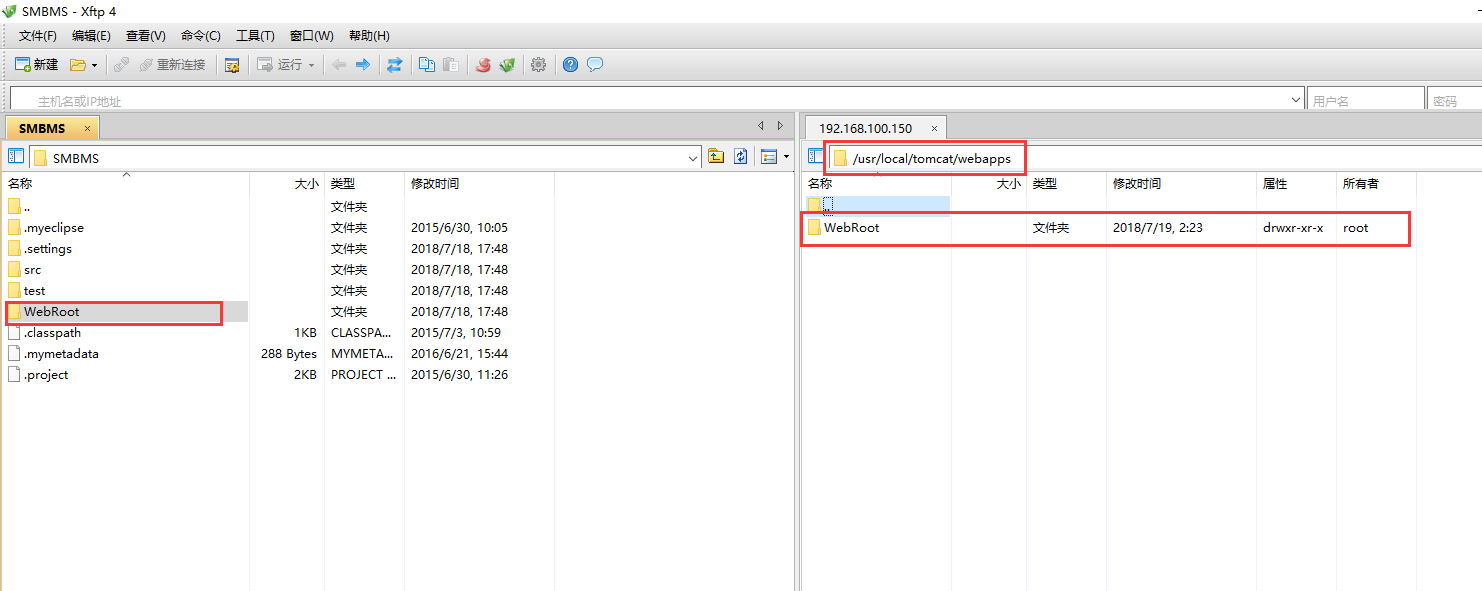
mysql> exit

* **超市订单管理系统项目上线；**

[root@www ~]# rm -rf /usr/local/tomcat/webapps/\* ##注意上传项目时的路径；

[root@www ~]# ls /usr/local/tomcat/webapps/

空



[root@www ~]# ls /usr/local/tomcat/webapps/

WebRoot

[root@www ~]# vi /usr/local/tomcat/webapps/WebRoot/WEB-INF/classes/database.properties

driver=com.mysql.jdbc.Driver

#在和mysql传递数据的过程中，使用unicode编码格式，并且字符集设置为utf-8

url=jdbc:mysql://192.168.100.101:3306/smbms?useUnicode=true&characterEncoding=utf-8

user=linuxfan

password=123123

* **客户端访问测试项目二；**

**登录后如若访问不了，可以尝试重启tomcat；**





* **自定义jsp页面，并测试访问；**

[root@www ~]# mkdir -p /web/webapp/

[root@www ~]# vi /web/webapp/index.jsp

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<html>

<head>

<title>JSP TEST PAGE1 </title>

</head>

<body>

<% out.println("Welcome to test site;http://www.linuxfan.cn");%>

</body>

</html>

:wq

[root@www ~]# vi /usr/local/tomcat/conf/server.xml

148 <Host name="www.linuxfan.cn" appBase="webapps"

149 unpackWARs="true" autoDeploy="true">

150 <Context docBase="/web/webapp" path="" reloadable="false"></Context>

注解：

unpackWARs="true" ##设置自动识别war包

autoDeploy="true" ##开启自动部署

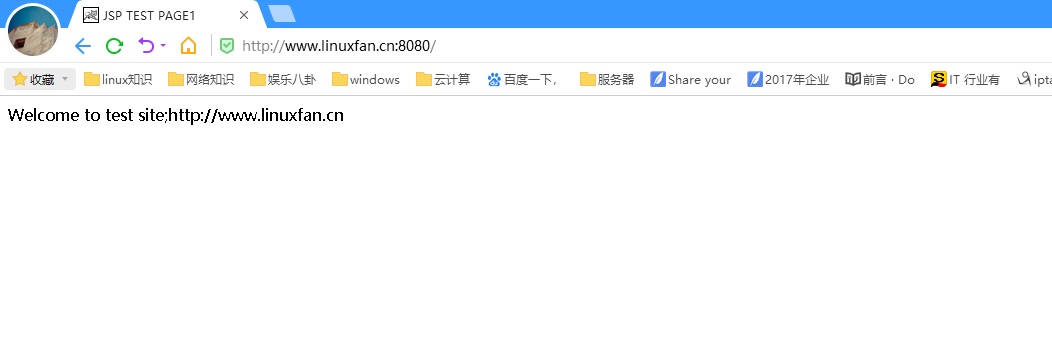
Context docBase="/web/webapp" ##设置项目存放位置

path="" ## tomcat把web应用程序映射为root URI路径reloadable="true" ## tomcat服务器在运行状态下会监视在WEB-INF/classes和WEB-INF/lib目录下class文件的改动，如果监测到代码中有class文件被更新的，服务器会自动重新加载Web应用，但是开启会加重服务器负荷，建议设置为false选项为关闭，有可能会出现内存溢出，tomcat服务不正常等异常，建议还是false掉，更新完代码脚本重启tomcat才是王道。；

[root@www ~]# /usr/local/tomcat/bin/shutdown.sh

[root@www ~]# /usr/local/tomcat/bin/startup.sh

客户端访问测试：



* **部署tomcat的虚拟主机，客户端访问测试；**

[root@www ~]# mkdir -p /web2/webapp/

[root@www ~]# cp /web/webapp/index.jsp /web2/webapp/

[root@www ~]# sed -i 's/linuxfan/linuxren/g' /web2/webapp/index.jsp

[root@www ~]# grep linuxfan /web/webapp/index.jsp

<% out.println("Welcome to test site;http://www.linuxfan.cn");%>

[root@www ~]# grep linuxren /web2/webapp/index.jsp

<% out.println("Welcome to test site;http://www.linuxren.cn");%>

[root@www ~]# vi /usr/local/tomcat/conf/server.xml

148 <Host name="www.linuxfan.cn" appBase="webapps"

149 unpackWARs="true" autoDeploy="true">

165 </Host>

166 <Host name="www.linuxren.cn" appBase="webapps"

167 unpackWARs="true" autoDeploy="true">

168 <Context docBase="/web2/webapp" path="" reloadable="false"></Context>

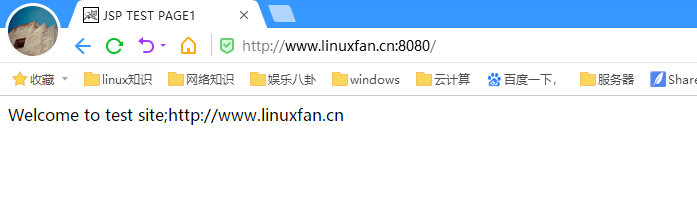
169 </Host>

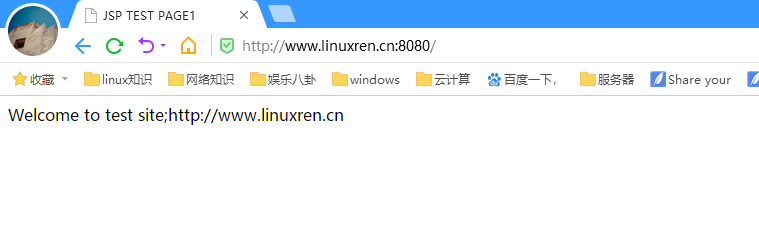
:wq

[root@www ~]# /usr/local/tomcat/bin/shutdown.sh

[root@www ~]# /usr/local/tomcat/bin/startup.sh

客户端访问测试：





注意：虽然更改了原有host，但是不影响/usr/local/tomcat/webapps下项目的访问；