# 2.单机环境部署

# 环境搭建

## ****1.    准备工作****

准备必要软件：JDK、svn、maven、tomcat

### 1.1.JDK准备

jdk准备有几种方法：

#### 1.1.1.下载

1.使用语句find / -name java 看在哪个位置有jdk安装包，如果没有则使用下面方法

2.从已有的服务器上远程scp

 scp -r -P 52119 /usr/local/java/jdk1.6.0\_45 root@本地服务器:/usr/local/java

3.使用wget语句从网上下载

wget  jdk下载网址

下载时首先注意先创建/usr/local/java文件目录，在文件目录下下载，这样下载的文件就直接放入创建的文件夹中，或者直接使用语句

wget  -P /usr/local/java 下载地址

#### 1.1.2安装

进入/usr/local/java文件目录下执行./ jdk-6u45-linux-x64.bin

#### 1.1.3.配置

环境变量配置

执行 vi  /etc/profile  在文件中添加JAVA\_HOME 、PATH、CLASSPATH路径如下：

export JAVA\_HOME=/usr/local/java/jdk1.6.0\_45

export PATH=$/JAVA\_HOME/bin:@PATH

export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

按ESC退出编辑，输入:wq保存后退出

输入source  /etc/profile 是配置文件生效

#### 1.1.4验证

输入java –version 和javac –version查看是否配置成功，两个一直则配置成功，这个有时会出现版本不一致情况（我出现过），具体解决方方法见：<http://blog.csdn.net/config_man/article/details/5717177>

### 1.2.maven准备

这里也不跟jdk一样那么细介绍了，这个也是下载、安装、配置、验证几个步骤，我这里简单介绍一下

#### 1.2.1.下载

下载可以先用find  / -name mvn搜索，如果没有则用远程scp或者wget，下载的时候提前创建文件目录/usr/local/maven

#### 1.2.2.安装

将下载的压缩文件解压即可tar zxvf apache-maven-3.0.5-bin.tar.gz

注意：jdk1.6与apache-maven-3.3.3版本不适合，你会发现配置之后mvn始终不能编译

#### 1.2.3.配置

vi /etc/profile   添加M2\_HOME和PATH路径,具体如下

export M2\_HOME=/usr/local/maven/apache-maven-3.0.5

export PATH=$PATH:$M2\_HOME/bin

#### 1.2.4.验证

输入mvn –v

在这一步我这边在服务器172.16.1.76上配置一个jdk1.7，maven3.3.3的，每天来使用mvn –v编译都会出问题，必须每天source  /etc/profile是配置文件生效一下就ok，具体我也不知道为什么。

### 1.3.svn准备

svn准备这个我经常出问题，至今没有完整弄好过一次，唯一一次就是在192.168.1.76上弄成功，当时下载出现一些列问题，最后在网上搜到一种方法 yum -y install svn成功了，之后自己试了很多机器都没有成功，都会显示错误，而且错误总是显示Could not retrieve mirrorlist <http://mirrorlist.centos.org/?release=6&arch=x86_64&repo=os> error was

14: PYCURL ERROR 6 - "Couldn't resolve host 'mirrorlist.centos.org'"

Error: Cannot find a valid baseurl for repo: base，大概意思是不能找到这个地址（我自己理解），不能解决主机，之后找到一种方法，适用语句echo "nameserver 8.8.8.8">>/etc/resolv.conf （好管用），这个命令直接DNS服务器写入Linux的DNS客户端解析文件resolv.conf里，之后在使用 yum –y install svn 安装不会出现问题了，当然这个也得看服务器的网络情况，有些好事可能有点慢，要耐心一点。

### 1.4.Tomcat下载

直接使用远程scp传文件或者wget语句下载

## ****2.svn下载代码****

由于刚进来没有svn权限，所以我每次下载代码都是通过上Jenkins下载代码（注意代码版本）

若有权限，则可以使用：svn co 代码路径

注意先创建一个文件夹存放代码，使用svn co下载代码需在当前文件夹下

## ****3.Maven编译打成war包****

1.maven是一个异常强大的构建工具，能够帮我们自动化构建过程，从清理，编译，测试到生成报告，再到打包部署，这里maven有单个配置文件pom.xml和两个setting.xml

pom.xml：主要描述项目：包括配置文件。开发者需要遵循规则，权限管理系统，组织和licenses，项目的url，项目的依赖性以及其他所有跟项目相关的因素

setting.xml存在于两个地方

1.安装的地方：maven安装目录下的conf里，这个称为全局配置

2.用户目录：用户目录下的.m2文件下面（这个需要自己添加进去），这个称为用户配置

2.使用语句mvn clean install –Pbeta –Dmaven.test.skip=true –U

解释语句：clean 作用是清除上一次构建

          install作用是将包安装至本地仓库，以让其他项目依赖

          -Pbeta作用是按照beta的分组进行打包，根据pom文件中profile选择分组进行打包

          -Dmaven.test.skip作用是跳过测试

          -U 强制更新snapshot类型的查件或依赖库

mvn编译过程中有可能失败，这要看配置文件里面是否有问题，主要是依赖文件，也有可能是版本问题，这在之前遇到过

## ****3.部署发布****

3.1.将mvn编译打包的target文件下的war包放入tomcat下面的webapps文件夹下，然后再bin目录下启动tomcat（./startup.sh），在启动之前可以观察日志，可以检查启动过程的问题（新开一个窗口cd /usr/local/tomcat\_core/logs路径下，tail -f catalina.out观察日志输出）

第一次启动的时候我看那就出问题了，主要是地址被占用，这个时候就要知道tomcat配置如何修改，修改路径在/usr/local/tomcat\_core/conf/的server.xml文件，主要修改端口，改成不一样就是了（端口可以随意改，只要不与其它在运行tomcat服务端口一致就行），之后查看正在进行的tomcat进程号，如果使我们起的服务起来了，先结束进程（kill），然后再启动tomcat观察日志，服务成功发布。至此环境搭建过程已结束。

## 附：

### tomcat端口配置解析

(1)<connector port="8080"protocol="HTTP/1.1"connectionTimeout="20000"

redirectPort="8443"/>

此连接器是负责建立HTTP连接。再通过浏览器访问Tomcat服务器的web应用时使用的就是这个端口，如果此端口被占用可将其修改一个未被占用的端口,例如 将上面代码中的port="8080"修改为port="8081"，即可将Tomcat的默认端口设置为8081。在修改端口时，应避免与公用端口冲突。建议采用默认的8080端口，不要修改，除非8080端口被其他程序所占用。

（2） <Server port="8005" shutdown="SHUTDOWN">

8005端口是用来关闭Tomcat服务的端口，如果此端口被占用,可在配置文件中将8005改成其他端口

（3）<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

此连接器负责和其他的HTTP服务器建立连接，再把Tomcat与其他的HTTP服务集成时需要使用此连接器。如果此端口被占用,可在配置文件中将8009改成其他端口

（4）<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"

maxThreads="150" scheme="https" secure="true" clientAuth="false"

keystoreFile="D:\Tomcat6.0\conf\tomcat.key" keystorePass="123456"

sslProtocol="TLS" />

此连接器的作用是启动HTTPS服务，一般的情况此段内容是被注释掉的，只有开启HTTPS服务时才会放开使用，如果此端口被占用则修改8443端口为其他没被占用的端口，并且需要将（1）和（3）中的也要修改

3.  修改成功后，为了使新设置的端口生效，还需要重新启动Tomcat服务器。

### 发布服务配置

war包解压后生成文件夹以core为例，在tomcat的webapps路径下进入yooli-web/WEN-INF/classes这个为服务器配置

（1）coreMessageTemplate.properties core发送消息的模板配置文件

（2）db.properties db数据库的配置文件

（3）dubbo-consumer.properties **服务的消费者**

（4）dubbo-provider.properties  **服务的提供者**

（5）log4j.properties     **日志文件的输出路径配置**

（6）mqclient.properties  **mq的配置**

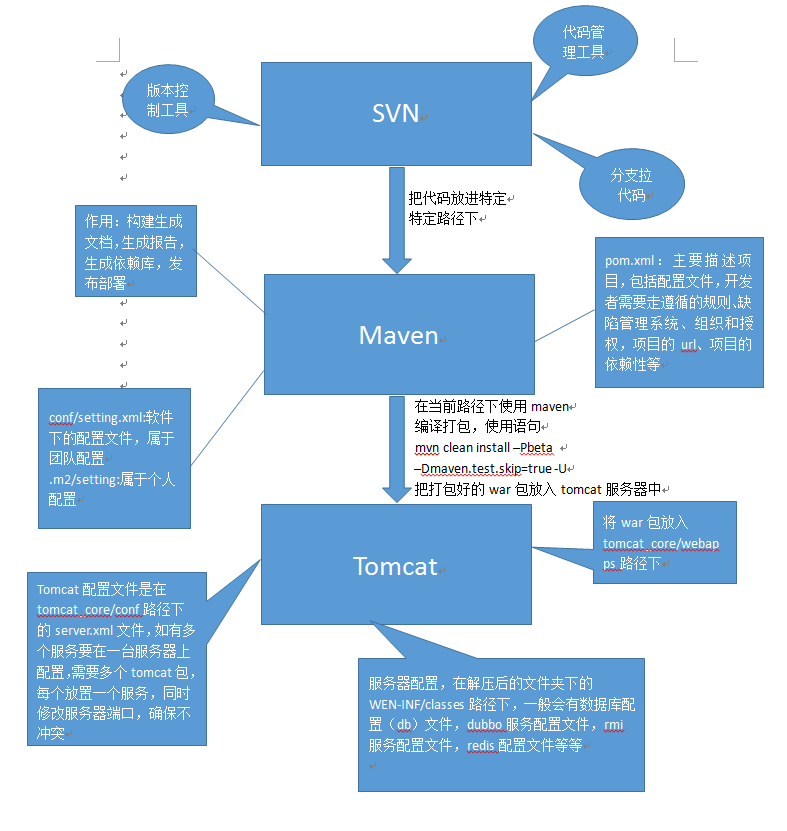
（7）mybatis-config.xml  **mybatis的配置文件**

（8）redis.properties     **redis的配置文件**

（9）wx.properties

任何修改服务器配置，修改完之后都需要重启tomcat发布服务。

### 整体流程图



### pom.xml文件解析

<project> </project>  根元素，包含约束信息

<modelVersion> </modelVersion>指定当前pom文件版本

<parent>

       <group        Id></groupId>主项目标识，

坐标    <artifactId></artifactId>模块标识，项目名+模块名

         <version></version>一般由三个数字组成：第一个0标识大版本号，第二个标识分支版本号，第三个标识小版本号

        snapshot 快照

        alpha    内部测试

        beta     公测

        release   稳定

        GA      正式发布

</parent>

<packaging></packaging> 设置打包方式，默认是jar包，也可以是war包，zip....

<name></name>项目描述名，产生项目文档才使用

<url></url>项目地址

<description></description>项目描述

<licenses></licenses>许可证

<organization></organization>组织

<dependencies>依赖列表

<dependency>依赖项

里面包含坐标信息（groupId、artifactesId、version）、type、scope（依赖范围）、optional(true or false 设置依赖是否可选)

<exclusions>

<exclusion>     排除依赖传递列表

</exclusion>

</exclusions>

</dependency>

</depencies>

<dependencymangement></dependencymangement>依赖管理

<bulid>

<plugins>插件系列表

<plugin>插件项

坐标.....

</plugin>

</plugins>

</bulid>