# 1.   环境准备

## 1.1. 服务端（[Windows](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)）

### 1.1.1.  [maven](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)环境准备

首先下载maven包，解压后放在任意位置，并配置系统环境变量

环境验证：在cmd命令行窗口中输入mvn –version

[](http://www.51testing.com/batch.download.php?aid=61732)

### 1.2.2.  jdk环境准备

安装jdk1.8并配置系统环境变量

### 1.2.3.  Tomcat准备

在本地部署Tomcat。

## 1.3. 服务端（[Linux](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)）

需要准备的内容类似，也是maven环境、JDK环境和tomcat

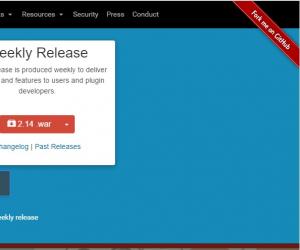
# 2.   平台搭建

## 2.1. 搭建步骤

1、登录[Jenkins](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)官网：

<https://jenkins.io/index.html>

下载最新Jenkins.war包

[](http://www.51testing.com/batch.download.php?aid=61733)

2、将war包放入Tomcat的webapps文件夹中

3、启动Tomcat

4、访问<http://localhost:9080/jenkins>

其中地址为本机地址，端口为在Tomcat中配置的端口

## 2.2.配置与插件

1、 点击系统管理——>系统设置，配置基本信息

2、 在JDK一栏中配置jdk版本及位置

3、 在MAVEN一栏中配置maven版本及位置

4、 点击系统管理——>插件管理，添加及更新插件

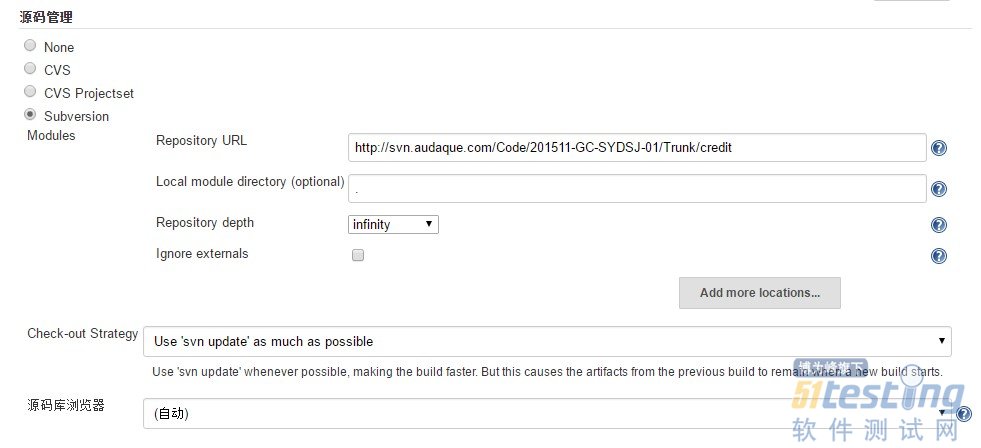
5、 在可选插件中搜索“Deploy to container Plugin”，添加

# 3.   [自动构建](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)配置

1、点击新建，选择新建一个maven项目

[](http://www.51testing.com/batch.download.php?aid=61734)

2、配置SVN源，只需改动SVN地址

[](http://www.51testing.com/batch.download.php?aid=61735)

3、配置构建参数

其中pom.xml当前工程均放在根目录下，无需更改，如位置改变，则进行相应修改即可。

下方命令为mvn命令：

clean package -Dmaven.test.skip=true -D maven.javadoc.skip=true

4、配置完成后点击下方的“保存”按钮。

# 4.   [自动部署](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)配置（方式一）

## 4.1.Tomcat准备

1.  确保要部署的Tomcat中有manager功能（[测试](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)环境开启manager，生产环境删除）

2.  进入Tomcat根目录中的conf文件夹，编辑tomcat-users.xml

3.  在最下方添加：

<role rolename="manager"/>

<role rolename="manager-gui"/>

<role rolename="admin"/>

<role rolename="admin-gui"/>

<role rolename="manager-script"/>

<role rolename="manager-jmx"/>

<role rolename="manager-status"/>

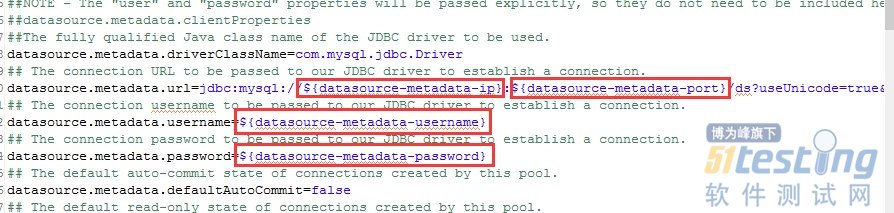
<user username="Tomcat" password="Tomcat" roles="admin-gui,admin,manager-gui,manager,manager-script,manager-jmx,manager-status"/>

## 4.2.maven准备

此处主要是采用Profilers的方式，准备多套配置文件环境，可以方便的在开发环境、测试环境A、测试环境B间切换

1.  在项目src/main下建立一个config文件夹，将需要修改的配置文件所在的文件夹放在此处

2.  将需要修改的配置文件进行参数化，以datasource/metadata.properties为例，将可变的地方参数化

[](http://www.51testing.com/batch.download.php?aid=61736)

3.  修改工程的pom.xml，添加maven-war-plugin插件引用，如pom中已有，则修改即可。其中directory是配置文件的来源，即刚才新建的参数化配置文件，targetPath是配置文件的目标，即项目发布完成后的配置文件位置。此处可以参考项目的实施文档。

<plugin>

     <groupId>org.apache.maven.plugins</groupId>

     <artifactId>maven-war-plugin</artifactId>

     <version>2.6</version>

     <configuration>

      <webResources>

       <resource>

        <directory>src/main/config/etc</directory>

        <targetPath>WEB-INF/etc</targetPath>

        <filtering>true</filtering>

       </resource>

      </webResources>

     </configuration>

    </plugin>

4.  配置Profiler，还是在pom.xml中，配置多套环境的参数值

配置结构为：（默认激活开发环境），在<properties>标签里配置参数值

<profiles>

  <profile>

   <!—测试环境-->

   <id>TestEnv</id>

   <properties>

    <exploration-datasource-ip>XXX</exploration-datasource-ip>

   </properties>

  </profile>

  <profile>

   <!--开发环境-->

   <id>DevEnv</id>

   <properties>

    <exploration-datasource-ip>XXX</exploration-datasource-ip>

   </properties>

   <!--默认激活本环境-->

   <activation>

    <activeByDefault>true</activeByDefault>

   </activation>

  </profile>

 </profiles>

## 4.3.Jenkins配置

1.  进入Jenkins平台，选择之前配置完自动构建的项目，点击“配置”

2.  修改build命令

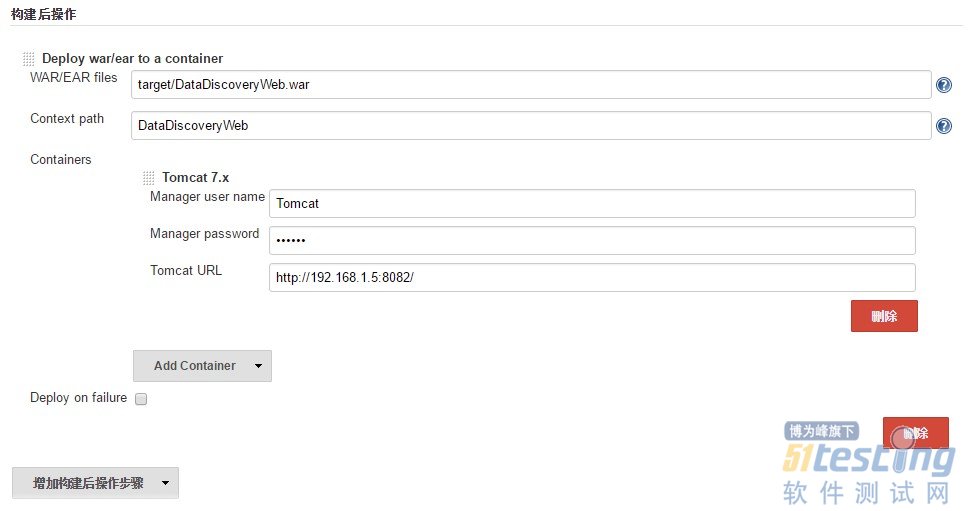
将Build模块的命令:

clean package -Dmaven.test.skip=true -D maven.javadoc.skip=true

改成：

clean package-P TestEnv-Dmaven.test.skip=true -D maven.javadoc.skip=true

3.  添加构建后操作：Deploy war/ear to a container

[](http://www.51testing.com/batch.download.php?aid=61737)

第一个参数war包文件路径为target/DataDiscoveryWeb.war（根据自己的项目确认），确认方式：

选择项目——>[工作](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)空间——>target——>找到war包

第二个参数填写访问名，即你想让发布后的系统以什么地址进行访问

4.  点击AddContainer，选择Tomcat 7.x

配置Tomcat的manager用户名和密码和地址，如按之前步骤操作，此处用户名和密码均为Tomcat

5.  点击保存

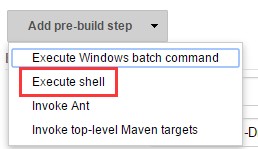
# 4.   自动部署配置（方式二 推荐）

## 4.1.Tomcat准备

同4.1

## 4.2.Jenkins配置

1.  进入项目配置页面，在Build操作前添加pre-build步骤，选择Execute [shell](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)脚本

[](http://www.51testing.com/batch.download.php?aid=61738)

格式：

[sed](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self) -i ‘s/需要修改的内容/想要修改成什么/g’文件位置（这里需要写src下的配置文件位置）

例如：

sed -i ‘s/192.168.1.4/192.168.1.5/g’ /root/.jenkins/jobs/XXX/workspace/src/main/webapp/WEB-INF/etc/datasource/data.properties

将src下的配置文件data.properties中的192.168.1.4改成192.168.1.5

2.  添加构建后操作：Deploy war/ear to a container

这里自动发布的步骤和4.1一致

3.  点击AddContainer，选择Tomcat 7.x

这里自动发布的步骤和4.1一致

# 5.   代码自动审查

## 5.1.[Sonar](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)安装

 参考sonarQube安装教程，安装sonarQube 6.0

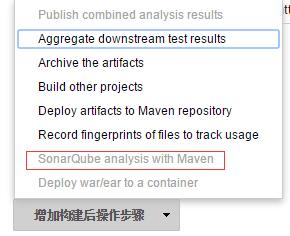
## 5.2.Sonar与Jenkins集成

1. 在Jenkins插件管理中下载[SonarQube Plugin](http://redirect.sonarsource.com/plugins/jenkins.html)插件

2. 在系统配置中配置sonarQube信息

3. 进入已经配置好的job，点击配置

4. 新增构建后操作步骤：

[](http://www.51testing.com/batch.download.php?aid=61740)

5. 再次执行构建后即可在sonarQube中查看项目的代码质量情况。

# 6.   触发构建

首先进入已经配置好的job，点击“立即构建”，即可触发构建

[](http://www.51testing.com/batch.download.php?aid=61739)

点击下方的编号，即可进入对应的构建中，点击Console Output，即可查看构建过程中日志。

# 7.   问题[记录](http://www.51testing.com/html/54/javascript:;" \t "http://www.51testing.com/html/54/_self)

1、**编码GBK的不可映射字符**

解决办法

org.apache.maven.plugins中添加<encoding>utf8</encoding>