|  |
| --- |
| 恒生电子股份有限公司 |
| UI组件开发指南 |
| JRESPLUS-UI |
|  |
| **研发中心** |
| **2016/2/16** |

|  |
| --- |
| [在此处键入文档摘要。摘要通常为文档内容的简短概括。在此处键入文档摘要。摘要通常为文档内容的简短概括。] |

目录

[1. 概述 2](#_Toc445680383)

[1.1. 目的 2](#_Toc445680384)

[1.2. 范围 2](#_Toc445680385)

[1.3. 定义 2](#_Toc445680386)

[1.4. 读者 2](#_Toc445680387)

[1.5. 参考资料 2](#_Toc445680388)

[2. 开发准备 2](#_Toc445680389)

[2.1. IDE环境搭建 2](#_Toc445680390)

[2.1.1. JDK安装 3](#_Toc445680391)

[2.1.2. TOMCAT安装 3](#_Toc445680392)

[2.1.3. Eclipse安装 3](#_Toc445680393)

[2.1.4. Maven安装 3](#_Toc445680394)

[2.1.5. VeloEclipse安装 4](#_Toc445680395)

[2.1.6. Eclipse配置 4](#_Toc445680396)

[2.2. 组件开发工程模板 6](#_Toc445680397)

[2.3. 组件依赖引入 7](#_Toc445680398)

[3. 组件开发 8](#_Toc445680399)

[3.1. 组件描述和API定义 8](#_Toc445680400)

[3.1.1. 组件描述 8](#_Toc445680401)

[3.1.2. API定义 8](#_Toc445680402)

[3.2. 组件开发 8](#_Toc445680403)

[3.2.1. JS组件开发 8](#_Toc445680404)

[3.2.2. CSS组件开发 9](#_Toc445680405)

[3.2.3. VM组件开发 10](#_Toc445680406)

[4. 组件开发调试 10](#_Toc445680407)

[5. 组件集成和发布 11](#_Toc445680408)

[6. 附录 11](#_Toc445680409)

[6.1. Maven setting.xml配置 11](#_Toc445680410)

# 概述

## 目的

本指南的目的是为了指导组件框架维护者和业务应用开发者，根据jresplus-ui组件开发规范，从事系统UI组件和业务UI组件的开发和封装工作。

## 范围

指南适用公司内部所有使用JRESPLUS-UI框架进行页面开发的产品。指南中不涉及组件开发时所使用的JS、CSS、VM技术语言的基本语法介绍，此内容请开发者自行查阅各语言的相关教材。指南主要描述开发jresplus-ui组件工程时的框架标准及建议约定。

## 定义

## 读者

**组件开发者：**从事生产UI组件的软件开发人员。

**组件框架维护者：**组件框架的开发和维护人员。

## 参考资料

《jresplus-ui组件框架规范.docx》

《jresplus-ui组件样式分离规范.docx》

# 开发准备

## IDE环境搭建

指南推荐的IDE开发环境见下表：

|  |  |  |  |
| --- | --- | --- | --- |
| 工具或插件 | 分类 | 版本 | 说明 |
| Sun JSDK | 运行环境 | V1.7+ | IDE依赖的Java虚拟机环境 |
| Tomcat | 运行环境 | V7.0+ | 开发工程运行容器 |
| Eclipse | 开发工具 | V4.0+ | IDE主开发环境 |
| Apache Maven | 开发工具 | V3.0.2+ | 用于依赖管理和应用集成 |
| VeloEclipse | Eclipse插件 | V2.0.8 | 用于VM代码开发 |
| maven-ant-plugin | Maven插件 | V1.6+ | 用于在Maven脚本中执行Ant任务（用于预编译脚本执行） |

### JDK安装

先通过oracle网站<http://www.oracle.com/technetwork/java/javase/downloads/index.html> 下载JDK安装文件。

执行安装文件，建议在安装JDK时，安装主目录路径中不要出现空格等特殊字符。安装完成后，打开操作系统的环境变量设置，配置：

JAVA\_HOME=[JDK安装主目录]

PATH=[%PATH%];%JAVA\_HOME%\bin ----这里[%PATH%]表示保留原来的PATH配置，在后面追加[%JAVA\_HOME%\bin]配置

CALSSPATH=%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar

### TOMCAT安装

先通过tomcat网站<http://tomcat.apache.org/download-80.cgi> 下载TOMCAT压缩包文件（download-80.cgi表示当前下载的是V8.0版本的tomcat，可根据需要调整下载版本）。

解压下载文件，解压后主目录路径中不要出现空格等特殊字符。

### Eclipse安装

先通过eclipse网站<http://www.eclipse.org/downloads/> 下载Eclipse压缩包文件。（建议下载[Eclipse IDE for Java EE Developers](http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/mars2)产品）

解压下载文件，解压后主目录路径中不要出现空格等特殊字符。

### Maven安装

先通过Maven网站<http://maven.apache.org/download.cgi> 下载Maven压缩文件包。

解压文件到本地目录，建议解压后的目录路径中不要出现空格等特殊字符。完成后，打开操作系统的环境变量设置，配置：

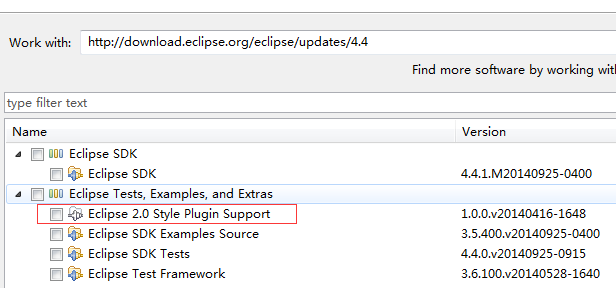
M2\_HOME=[Maven解压路径]

PATH=[%PATH%];%M2\_HOME%\bin ----这里[%PATH%]表示保留原来的PATH配置，在后面追加[%M2\_HOME%\bin]配置

进入[%M2\_HOME%]\conf目录，打开setting.xml文件进行Maven本地仓库配置。配置参见附录[8.1.Maven setting.xml配置](#_Maven_setting.xml配置)。其中localRepository用于指定本地Maven仓库的放置位置，需要根据本地Maven的安装路径重新设置。

### VeloEclipse安装

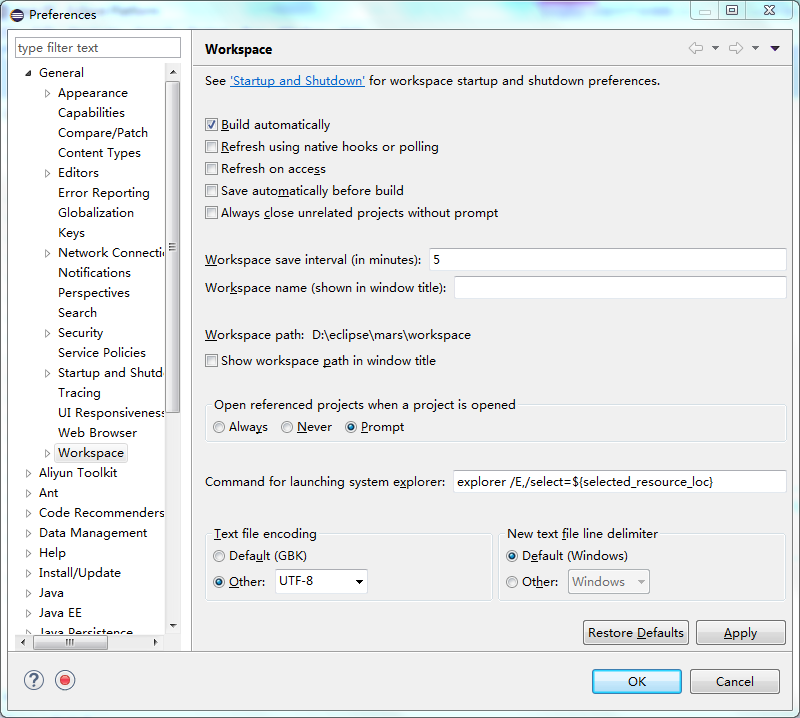
1. 打开Eclipse工具，选择菜单🡪Help🡪install new software🡪Add 输入 <http://download.eclipse.org/eclipse/updates/4.4> （这里链接末尾的4.4指安装的Eclipse版本号，如果实际）
2. 仅勾选Eclipse 2.0 Style Plugin Support 如下图：

[](http://www.ijiuwen.com/upload/file/2014/12/28/FILE_sENb_740987689205760.png)

1. 离线安装veloeclipse 插件。插件包可以从组件开发工程模板的site目录下获得。解压插件包后得到eclipse目录，将目录中的features、plugins两个目录复制到Eclipse工具的主目录中。

### Eclipse配置

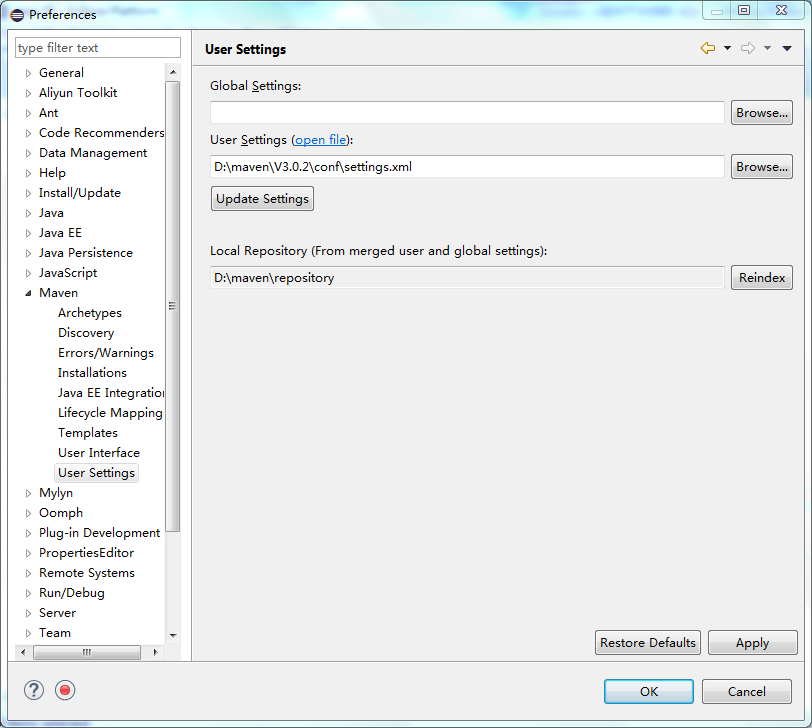
1. 设置字符集：打开Eclipse工具，选择菜单🡪Window🡪Preferences🡪General🡪Workspace，如图：



设置字符集

设置Eclipse默认字符集为UTF-8。

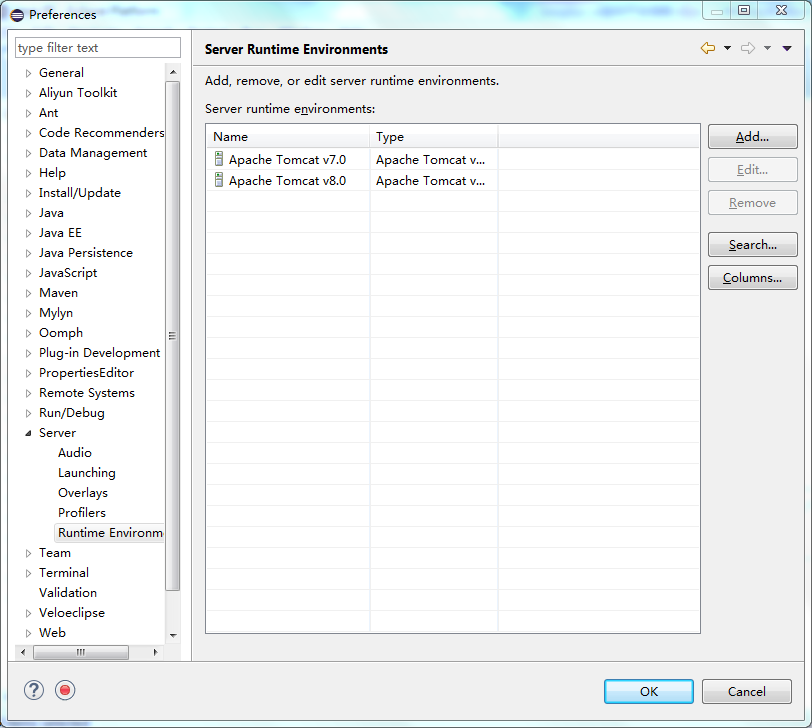
1. 配置Maven：打开Eclipse工具，选择菜单🡪Window🡪Preferences🡪Maven🡪User Settings，如图：



设置Maven配置文件

选择[%M2\_HOME%\conf\setting.xml]文件做为Eclipse工具的Maven基本配置文件。

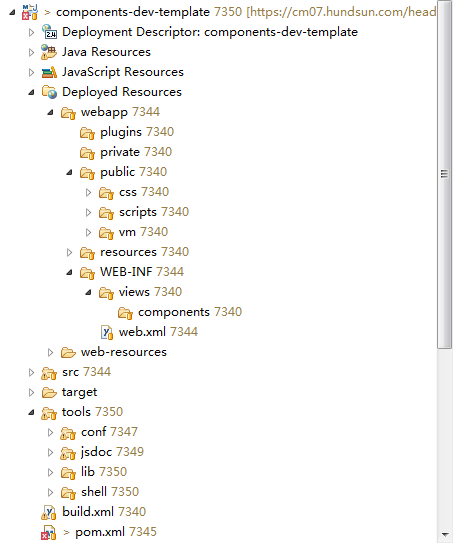
1. 配置TOMCAT：打开Eclipse工具，选择菜单🡪Window🡪Preferences🡪Server🡪Runtime Environm，如图：



可以在这里添加已安装的Tomcat服务器。(目前Eclipse V4.5下最高支持tomcat v8.0版本的整合)

## 组件开发工程模板

组件框架为组件开发者提供组件开发工程模板：



组件开发工程模板请到jrespjus ui网站进行下载

<http://rdc.hundsun.com/jresui>

下载完成后进行解压，再使用Eclipse菜单File🡪Import🡪Maven🡪Existing Maven Projects进行工程导入即可。

## 组件依赖引入

组件工程采用Maven仓库管理引用和发布的组件包。组件开发者在开发组件过程中，如果遇到自定义组件是基于其它组件进行扩展的场景，需要在开发组件工程中对被扩展的组件包进行依赖引入。

开发者可以在工程的pom.xml配置文件里添加：

|  |
| --- |
| <dependency>  <groupId>[分组标识]</groupId>  <artifactId>[组件包标识]</artifactId>  <version>[版本号]</version>  <classifier>components</classifier>  </dependency> |

注意：

* classifier必填components。由于组件包是使用maven的assembly插件进行打包的，在assembly脚本中定义了classifier的值为components，所以需要在引用时也加上此值。
* 组件依赖包在mavne中通过传递依赖将间接依赖的组件包引入开发工程。

# 组件开发

## 组件描述和API定义

### 组件描述

正式开始组件开发的第一个工作是定义组件。组件开发者通过META-INF\components.xml进行组件描述活动。开发者可参照《jresplus-ui组件框架规范.docx》4.1.组件定义小节进行组件xml描述。

### API定义

组件公开的API采用代码注释方式写在控件实现的JS源码文件中。开发者可参照《jresplus-ui组件框架规范.docx》2.3.格式化注释小节进行组件API的定义。

## 组件开发

### JS组件开发

1. **开发目录说明：**

组件开发者通过两个目录编辑自己的JS组件代码：

* webapp\WEB-INF\public\scripts\：此目录用来编写JS组件中需要暴露给外部的组件公开部分。

此目录的资源在组件集成发布时会被打包至组件jar包的public\scripts\目录中。

在后续组件包使用过程中，public\scripts\目录中的JS代码可以被引用者直接调用；框架工具脚本默认会扫描此目录的JS代码，为组件生成对外的API文档，并对此目录的JS资源进行压缩合并。

* webapp\WEB-INF\private\scripts\：此目录用来编写JS组件中私有部分代码。

此目录的资源在组件集成发布时会被打包至组件jar包的private\scripts\目录中。

在后续组件包使用过程中，public\scripts\目录中的JS代码不允许引用者直接调用；且工具脚本对此目录不做任何处理。

1. **文件命名规范：**

JAVASCRIPT源程序是以.js为后缀，以能说明功能的英文单词命名，如果需要多个单词，则每个单词的第一个字母大写，如：EducationModify.js；个别单词过长可适当缩写，如：information可缩写成info，experience可以缩写成exp。

1. **JS代码命名规范：**

|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 规则 | 样例 | 是否要求严格合规 |
| JS类 | 驼峰式 | ModuleClass() | 是 |
| JS公开方法 | 混合式 | getPosition() | 是 |
| JS公开属性 | 混合式 | frameStyle | 是 |
| JS公开常量 | 大写式 | DEFAULT\_FRAME\_LAYOUT | 是 |
| JS私有方法 | 混合式 | getMixedCase() | 否 |
| JS私有属性 | 混合式 | mixedCase | 否 |
| JS方法参数 | 混合式 | mixedCase | 否 |
| JS方法内部变量 | 混合式 | mixedCase | 否 |

除了列表中的命名规则约定，建议开发者在开发过程中可以遵守：

1. 所有变量必须是有意义的英文，严厉禁止拼音
2. 变量允许使用公认英文缩写
3. 枚举类型时，枚举的命名必须有意义
4. 私有类的变量属性成员 建议使用混合式命名，并在前面加下划线
5. 函数开头必须是有意义的动词或动词短语
6. "on"只能用作事件的命名前缀
7. 处理错误的变量，建议使用“Error”后缀
8. 初始化用的函数建议使用“init”前缀
9. 保留字以及特有的dom属性不要作为变量名
10. “is”前缀的变量名应该为布尔值
11. ”str”前缀的变量名应该为字符串
12. “arr”前缀的变量名应该为数组
13. “num”或“count”前缀的变量名应该为数字
14. 循环重复变量建议使用 "i", "j", "k" （依次类推）等名称的变量(全世界公认)
15. 能缩写的单词尽量缩写
16. **其它编码规范：**
17. 常量和枚举必须在最前端定义，merge时注意，必须把常量与枚举定义的文件放在文件列表的第一位
18. 临时的全局变量放到一个全局的哈希表里，方便变量回收
19. 所有全局变量必须初始化，尽量少用全局变量
20. 不推荐使用eval来执行脚本，除非用来解析json数据
21. 避免频繁使用previousSibling和nextSibling

### CSS组件开发

1. **开发目录说明：**

组件开发者通过两个目录编辑自己的CSS代码：

* webapp\WEB-INF\public\css\：此目录用来编写需要暴露给外部的样式公开部分。

此目录的资源在组件集成发布时会被打包至组件jar包的public\css\目录中。

* webapp\WEB-INF\private\css\：此目录用来编写样式的私有部分。

此目录的资源在组件集成发布时会被打包至组件jar包的private\css\目录中。

1. **开发相关规范：**

参见《jresplus-ui组件样式分离规范.doc》3.2.CSS编码规范小节。

### VM组件开发

1. **开发目录说明：**

组件开发者通过两个目录编辑自己的VM组件代码：

* webapp\WEB-INF\public\vm\：此目录用来编写vm组件中需要暴露给外部的组件公开部分。

此目录的资源在组件集成发布时会被打包至组件jar包的public\vm\目录中。

在后续组件包使用过程中，public\vm\目录中的vm代码可以被引用者直接调用；框架工具脚本默认会扫描此目录的vm代码，对此目录的vm资源进行压缩合并。

* webapp\WEB-INF\private\vm\：此目录用来编写vm组件中私有部分代码。

此目录的资源在组件集成发布时会被打包至组件jar包的private\vm\目录中。

在后续组件包使用过程中，public\vm\目录中的vm代码不允许引用者直接调用；且工具脚本对此目录不做任何处理。

1. **VM中的宏命名规范：**

|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 规则 | 样例 | 是否要求严格合规 |
| 公开宏 | 下划线式 | panel\_end | 是 |
| 私有宏 | 下划线式 | form\_grid\_start | 否 |
| 宏参数 | 混合式 | $uiRadiogroupConfig | 否 |
| 宏内部变量 | 混合式 | $checkStr | 否 |

1. **VM中的HTML编码规范：**

参见《jresplus-ui组件样式分离规范.doc》3.1.HTML编码规范小节。

# 组件开发调试

组件开发者在开发调试前，需要使用tools\shell\目录下的”pre\_compiled.bat”脚本进行预编译。预编译脚本负责对组件资源进行合并及压缩：

* public\scripts\目录下的JS资源被合并后放入components目录下，生成合并压缩脚本[组件标识].mini.js和合并调试脚本[组件标识].js。
* public\css\目录下的CSS资源被合并后放入components目录下，生成合并压缩脚本[组件标识].mini.css和合并调试脚本[组件标识].css。
* public\vm\目录下的VM资源被合并后放入components目录下，生成合并宏文件[组件标识].vm。

然后，组件开发者需要修改运行时配置文件WEB-INF\conf\server.properties中配置项：

|  |
| --- |
| #是否是开发模式  system.dev.mode=true |

最后，组件开发者可以在Eclipse中配置的Server服务器中运行此工程，在浏览器中进行开发调试。

# 组件集成和发布

组件开发工程模板基于Maven提供了组件集成POM配置，组件开发者使用以下命令进行组件工程的集成和发布：

mvn clean：清除集成构建目录

mvn package：编译组件工程

mvn install：打包组件工程，并将组件包发布到本地仓库

mvn deploy：将组件包发布到远程仓库

# 附录

## Maven setting.xml配置

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <!--  Licensed to the Apache Software Foundation (ASF) under one  or more contributor license agreements. See the NOTICE file  distributed with this work for additional information  regarding copyright ownership. The ASF licenses this file  to you under the Apache License, Version 2.0 (the  "License"); you may not use this file except in compliance  with the License. You may obtain a copy of the License at  http://www.apache.org/licenses/LICENSE-2.0  Unless required by applicable law or agreed to in writing,  software distributed under the License is distributed on an  "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY  KIND, either express or implied. See the License for the  specific language governing permissions and limitations  under the License.  -->  <!--  | This is the configuration file for Maven. It can be specified at two levels:  |  | 1. User Level. This settings.xml file provides configuration for a single user,  | and is normally provided in ${user.home}/.m2/settings.xml.  |  | NOTE: This location can be overridden with the CLI option:  |  | -s /path/to/user/settings.xml  |  | 2. Global Level. This settings.xml file provides configuration for all Maven  | users on a machine (assuming they're all using the same Maven  | installation). It's normally provided in  | ${maven.home}/conf/settings.xml.  |  | NOTE: This location can be overridden with the CLI option:  |  | -gs /path/to/global/settings.xml  |  | The sections in this sample file are intended to give you a running start at  | getting the most out of your Maven installation. Where appropriate, the default  | values (values used when the setting is not specified) are provided.  |  |-->  <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">  <!-- localRepository  | The path to the local repository maven will use to store artifacts.  |  | Default: ${user.home}/.m2/repository  <localRepository>/path/to/local/repo</localRepository>  -->  <localRepository>D:\maven\V3.3.3\repo</localRepository>  <!-- interactiveMode  | This will determine whether maven prompts you when it needs input. If set to false,  | maven will use a sensible default value, perhaps based on some other setting, for  | the parameter in question.  |  | Default: true  <interactiveMode>true</interactiveMode>  -->  <!-- offline  | Determines whether maven should attempt to connect to the network when executing a build.  | This will have an effect on artifact downloads, artifact deployment, and others.  |  | Default: false  <offline>false</offline>  -->  <!-- pluginGroups  | This is a list of additional group identifiers that will be searched when resolving plugins by their prefix, i.e.  | when invoking a command line like "mvn prefix:goal". Maven will automatically add the group identifiers  | "org.apache.maven.plugins" and "org.codehaus.mojo" if these are not already contained in the list.  |-->  <pluginGroups>  <!-- pluginGroup  | Specifies a further group identifier to use for plugin lookup.  <pluginGroup>com.your.plugins</pluginGroup>  -->  </pluginGroups>  <!-- proxies  | This is a list of proxies which can be used on this machine to connect to the network.  | Unless otherwise specified (by system property or command-line switch), the first proxy  | specification in this list marked as active will be used.  |-->  <proxies>  <!-- proxy  | Specification for one proxy, to be used in connecting to the network.  |  <proxy>  <id>optional</id>  <active>true</active>  <protocol>http</protocol>  <username>proxyuser</username>  <password>proxypass</password>  <host>proxy.host.net</host>  <port>80</port>  <nonProxyHosts>local.net|some.host.com</nonProxyHosts>  </proxy>  -->  </proxies>  <!-- servers  | This is a list of authentication profiles, keyed by the server-id used within the system.  | Authentication profiles can be used whenever maven must make a connection to a remote server.  |-->  <servers>  <!-- server  | Specifies the authentication information to use when connecting to a particular server, identified by  | a unique name within the system (referred to by the 'id' attribute below).  |  | NOTE: You should either specify username/password OR privateKey/passphrase, since these pairings are  | used together.  |  <server>  <id>deploymentRepo</id>  <username>repouser</username>  <password>repopwd</password>  </server>  -->  <!-- Another sample, using keys to authenticate.  <server>  <id>siteServer</id>  <privateKey>/path/to/private/key</privateKey>  <passphrase>optional; leave empty if not used.</passphrase>  </server>  -->  <server>  <id>nexus</id>  <username>readOnly</username>  <password>hscmreadonly</password>  </server>    <server>  <id>hundsun\_rd\_hosted\_release</id>  <username>readOnly</username>  <password>nexus@0831</password>  </server>  <server>  <id>hundsun\_rd\_hosted\_snapshot</id>  <username>readOnly</username>  <password>nexus@0831</password>  </server>  </servers>  <!-- mirrors  | This is a list of mirrors to be used in downloading artifacts from remote repositories.  |  | It works like this: a POM may declare a repository to use in resolving certain artifacts.  | However, this repository may have problems with heavy traffic at times, so people have mirrored  | it to several places.  |  | That repository definition will have a unique id, so we can create a mirror reference for that  | repository, to be used as an alternate download site. The mirror site will be the preferred  | server for that repository.  |-->  <mirrors>  <!-- mirror  | Specifies a repository mirror site to use instead of a given repository. The repository that  | this mirror serves has an ID that matches the mirrorOf element of this mirror. IDs are used  | for inheritance and direct lookup purposes, and must be unique across the set of mirrors.  |  <mirror>  <id>mirrorId</id>  <mirrorOf>repositoryId</mirrorOf>  <name>Human Readable Name for this Mirror.</name>  <url>http://my.repository.com/repo/path</url>  </mirror>  -->  <mirror>  <!--This sends everything else to /public -->  <id>nexus</id>  <mirrorOf>\*</mirrorOf>  <url>http://repos.hundsun.com:6060/nexus/content/groups/public</url>  </mirror>  </mirrors>  <!-- profiles  | This is a list of profiles which can be activated in a variety of ways, and which can modify  | the build process. Profiles provided in the settings.xml are intended to provide local machine-  | specific paths and repository locations which allow the build to work in the local environment.  |  | For example, if you have an integration testing plugin - like cactus - that needs to know where  | your Tomcat instance is installed, you can provide a variable here such that the variable is  | dereferenced during the build process to configure the cactus plugin.  |  | As noted above, profiles can be activated in a variety of ways. One way - the activeProfiles  | section of this document (settings.xml) - will be discussed later. Another way essentially  | relies on the detection of a system property, either matching a particular value for the property,  | or merely testing its existence. Profiles can also be activated by JDK version prefix, where a  | value of '1.4' might activate a profile when the build is executed on a JDK version of '1.4.2\_07'.  | Finally, the list of active profiles can be specified directly from the command line.  |  | NOTE: For profiles defined in the settings.xml, you are restricted to specifying only artifact  | repositories, plugin repositories, and free-form properties to be used as configuration  | variables for plugins in the POM.  |  |-->  <profiles>  <!-- profile  | Specifies a set of introductions to the build process, to be activated using one or more of the  | mechanisms described above. For inheritance purposes, and to activate profiles via <activatedProfiles/>  | or the command line, profiles have to have an ID that is unique.  |  | An encouraged best practice for profile identification is to use a consistent naming convention  | for profiles, such as 'env-dev', 'env-test', 'env-production', 'user-jdcasey', 'user-brett', etc.  | This will make it more intuitive to understand what the set of introduced profiles is attempting  | to accomplish, particularly when you only have a list of profile id's for debug.  |  | This profile example uses the JDK version to trigger activation, and provides a JDK-specific repo.  <profile>  <id>jdk-1.4</id>  <activation>  <jdk>1.4</jdk>  </activation>  <repositories>  <repository>  <id>jdk14</id>  <name>Repository for JDK 1.4 builds</name>  <url>http://www.myhost.com/maven/jdk14</url>  <layout>default</layout>  <snapshotPolicy>always</snapshotPolicy>  </repository>  </repositories>  </profile>  -->  <!--  | Here is another profile, activated by the system property 'target-env' with a value of 'dev',  | which provides a specific path to the Tomcat instance. To use this, your plugin configuration  | might hypothetically look like:  |  | ...  | <plugin>  | <groupId>org.myco.myplugins</groupId>  | <artifactId>myplugin</artifactId>  |  | <configuration>  | <tomcatLocation>${tomcatPath}</tomcatLocation>  | </configuration>  | </plugin>  | ...  |  | NOTE: If you just wanted to inject this configuration whenever someone set 'target-env' to  | anything, you could just leave off the <value/> inside the activation-property.  |  <profile>  <id>env-dev</id>  <activation>  <property>  <name>target-env</name>  <value>dev</value>  </property>  </activation>  <properties>  <tomcatPath>/path/to/tomcat/instance</tomcatPath>  </properties>  </profile>  -->    <profile>  <id>nexus</id>  <!--Enable snapshots for the built in central repo to direct -->  <!--all requests to nexus via the mirror -->  <repositories>  <repository>  <id>hundsun-central-repos</id>  <url>http://repos.hundsun.com:6060/nexus/content/groups/public</url>  <releases><enabled>true</enabled></releases>  <snapshots><enabled>true</enabled></snapshots>  </repository>  </repositories>    <pluginRepositories>  <pluginRepository>  <id>hundsun-central-repos</id>  <url>http://repos.hundsun.com:6060/nexus/content/groups/public</url>  <releases><enabled>true</enabled></releases>  <snapshots><enabled>true</enabled></snapshots>  </pluginRepository>  </pluginRepositories>  </profile>  </profiles>  <!-- activeProfiles  | List of profiles that are active for all builds.  |  <activeProfiles>  <activeProfile>alwaysActiveProfile</activeProfile>  <activeProfile>anotherAlwaysActiveProfile</activeProfile>  </activeProfiles>  -->    <activeProfiles>  <activeProfile>nexus</activeProfile>  </activeProfiles>  </settings> |