**Maven基础-默认中央仓库[settings.xml 配置详解 ]**

2013年09月26日 21:35:41 [yiluoak\_47](https://me.csdn.net/yiluoAK_47) 阅读数：118349更多

个人分类： [maven](https://blog.csdn.net/yiluoAK_47/article/category/1343252)

首先：Maven中央仓库的搜索全部公共jar包的地址是，http://search.maven.org/

===Maven基础-默认中央仓库===================================

当我们创建一个简单的Maven项目后(只需要在pom.xml配置好依赖)，运行mvn clean install就可以把项目构建好，不需要我们手工下载任何jar，这全靠中央仓库的存在，它会自动从仓库下载。这个仓库的定义是在*${M2\_HOME}/lib/maven-2.0.10-uber.jar* 里面。你可以在里面找到*\org\apache\maven\project\pom-4.0.0.xml* 这个文件，在这个文件里面定义了默认中央仓库的地址：

**[html]** [view plaincopy](http://blog.csdn.net/earth3008/article/details/9262409)

1. **<repositories>**
2. **<repository>**
3. **<id>** central**</id>**
4. **<name>** Maven Repository Switchboard**</name>**
5. **<layout>** default**</layout>**
6. **<url>** http://repo1.maven.org/maven2**</url>**
7. **<snapshots>**
8. **<enabled>** false**</enabled>**
9. **</snapshots>**
10. **</repository>**
11. **</repositories>**

如果你想覆盖中央仓库的默认地址，那么这里我们就会使用的镜像了，还在setting.xml里面配置：

**[html]** [view plaincopy](http://blog.csdn.net/earth3008/article/details/9262409)

1. **<settings>**
2. …
3. **<mirrors>**
4. **<mirror>**
5. **<id>** maven-net-cn**</id>**
6. **<name>** Maven China Mirror**</name>**
7. **<url>** http://maven.net.cn/content/groups/public/**</url>**
8. **<mirrorOf>** central**</mirrorOf>**
9. **</mirror>**
10. **</mirrors>**
11. …
12. **</settings>**

<mirrorOf>,表示只为central仓库做镜像，如果想为所有的仓库做镜像那么可以改为：

<mirrorOf>\*</mirrorOf>

**===Maven配置详解===================================**

maven2 比起maven1 来说，需要配置的文件少多了，主要集中在pom.xml和settings.xml中。  
    先来说说settings.xml，settings.xml对于maven来说相当于全局性的配置，用于所有的项目。在maven2中存在两个settings.xml，一个位于maven2的安装目录conf下面，作为全局性配置。对于团队设置，保持一致的定义是关键，所以maven2/conf下面的settings.xml就作为团队共同的配置文件。保证所有的团队成员都拥有相同的配置。当然对于每个成员，都需要特殊的自定义设置，如用户信息，所以另外一个settings.xml就作为本地配置。默认的位置为：${user.dir}/.m2/settings.xml目录中（${user.dir} 指windows 中的用户目录）。  
    settings.xml基本结构如下：  
    <settings xmlns="http://maven.apache.org/POM/4.0.0"  
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0   
                               http://maven.apache.org/xsd/settings-1.0.0.xsd">  
  <localRepository/>  
  <interactiveMode/>  
  <usePluginRegistry/>  
  <offline/>  
  <pluginGroups/>  
  <servers/>  
  <mirrors/>  
  <proxies/>  
  <profiles/>  
  <activeProfiles/>  
</settings>  
简单介绍一下几个主要的配置因素：  
**localRepository：**表示本地库的保存位置，也就是maven2主要的jar保存位置，默认在${user.dir}/.m2/repository，如果需要另外设置，就换成其他的路径。  
**offline：**如果不想每次编译，都去查找远程中心库，那就设置为true。当然前提是你已经下载了必须的依赖包。  
**Servers**  
   在POM中的 distributionManagement元素定义了开发库。然而，特定的username和pwd不能使用于pom.xml，所以通过此配置来保存server信息  
  <servers>  
    <server>  
      <id>server001</id>  
      <username>my\_login</username>  
      <password>my\_password</password>  
      <privateKey>${usr.home}/.ssh/id\_dsa</privateKey>  
      <passphrase>some\_passphrase</passphrase>  
      <filePermissions>664</filePermissions>  
      <directoryPermissions>775</directoryPermissions>  
      <configuration></configuration>  
    </server>  
  </servers> 

* id:server 的id,用于匹配distributionManagement库id，比较重要。
* username, password:用于登陆此服务器的用户名和密码
* privateKey, passphrase：设置private key，以及passphrase
* filePermissions, directoryPermissions：当库文件或者目录创建后，需要使用权限进行访问。参照unix文件许可，如664和775

**Mirrors**   
表示镜像库，指定库的镜像，用于增加其他库  
  <mirrors>  
    <mirror>  
      <id>planetmirror.com</id>  
      <name>PlanetMirror Australia</name>  
      <url>http://downloads.planetmirror.com/pub/maven2</url>  
      <mirrorOf>central</mirrorOf>  
    </mirror>  
  </mirrors>

* id,name:唯一的标志，用于区别镜像
* url:镜像的url
* mirrorOf：此镜像指向的服务id

**Proxies**   
此设置，主要用于无法直接访问中心的库用户配置。  
  <proxies>  
    <proxy>  
      <id>myproxy</id>  
      <active>true</active>  
      <protocol>http</protocol>  
      <host>proxy.somewhere.com</host>  
      <port>8080</port>  
      <username>proxyuser</username>  
      <password>somepassword</password>  
      <nonProxyHosts>\*.google.com|ibiblio.org</nonProxyHosts>  
    </proxy>  
  </proxies>

* id:代理的标志
* active：是否激活代理
* protocol, host, port:protocol://host:port 代理
* username, password：用户名和密码
* nonProxyHosts: 不需要代理的host

**Profiles**   
  类似于pom.xml中的profile元素，主要包括activation,repositories,pluginRepositories 和properties元素  
  刚开始接触的时候，可能会比较迷惑，其实这是maven2中比较强大的功能。从字面上来说，就是个性配置。  
  单独定义profile后，并不会生效，需要通过满足条件来激活。  
 **repositories 和pluginRepositories**  
 定义其他开发库和插件开发库。对于团队来说，肯定有自己的开发库。可以通过此配置来定义。  
 如下的配置，定义了本地开发库，用于release 发布。  
    <repositories>  
        <repository>  
          <id>repo-local</id>  
       <name>Internal 开发库</name>  
       <url>http://192.168.0.2:8082/repo-local</url>  
          <releases>  
            <enabled>true</enabled>  
            <updatePolicy>never</updatePolicy>  
            <checksumPolicy>warn</checksumPolicy>  
          </releases>  
          <snapshots>  
            <enabled>false</enabled>  
          </snapshots>  
          <layout>default</layout>  
        </repository>  
      </repositories>  
      <pluginRepositories>  
    <pluginRepository>  
    <id>repo-local</id>  
    <name>Internal 开发库</name>  
    <url>http://192.168.0.2:8082/repo-local</url>  
    <releases>  
            <enabled>true</enabled>  
            <updatePolicy>never</updatePolicy>  
            <checksumPolicy>warn</checksumPolicy>  
    </releases>  
    <snapshots>  
    <enabled>false</enabled>  
    </snapshots>  
    <layout>default</layout>  
    </pluginRepository>  
    </pluginRepositories>  
releases, snapshots:每个产品的版本的Release或者snapshot(注：release和snapshot的区别，release一般是比较稳定的版本，而snapshot基本上不稳定，只是作为快照）  
  
**properties**  
  maven 的properties作为placeholder值，如ant的properties。  
包括以下的5种类型值：

1. env.X，返回当前的环境变量
2. project.x:返回pom中定义的元素值，如project.version
3. settings.x：返回settings.xml中定义的元素
4. java 系统属性：所有经过java.lang.System.getProperties()返回的值
5. x：用户自己设定的值

**Activation**   
  用于激活此profile  
  <activation>  
        <activeByDefault>false</activeByDefault>  
        <jdk>1.5</jdk>  
        <os>  
          <name>Windows XP</name>  
          <family>Windows</family>  
          <arch>x86</arch>  
          <version>5.1.2600</version>  
        </os>  
        <property>  
          <name>mavenVersion</name>  
          <value>2.0.3</value>  
        </property>  
        <file>  
          <exists>${basedir}/file2.properties</exists>  
          <missing>${basedir}/file1.properties</missing>  
        </file>  
      </activation>

* jdk:如果匹配指定的jdk版本，将会激活
* os:操作系统
* property：如果maven能检测到相应的属性
* file: 用于判断文件是否存在或者不存在

除了使用activation来激活profile，同样可以通过activeProfiles来激活  
**Active Profiles**  
表示激活的profile,通过profile id来指定。  
  <activeProfiles>  
    <activeProfile>env-test</activeProfile> 指定的profile id  
  </activeProfiles>