[使用maven profile实现多环境可移植构建](http://blog.csdn.net/mhmyqn/article/details/24501281)

在开发过程中，我们的软件会面对不同的运行环境，比如开发环境、测试环境、生产环境，而我们的软件在不同的环境中，有的配置可能会不一样，比如数据源配置、日志文件配置、以及一些软件运行过程中的基本配置，那每次我们将软件部署到不同的环境时，都需要修改相应的配置文件，这样来回修改，是个很麻烦的事情。有没有一种方法能够让我们不用修改配置就能发布到不同的环境中呢？当然有，这就是接下来要做的事。

当然，这里的前提是使用maven做为构建工具。

使用maven来实现多环境的构建可移植性，需要借助maven提供的profile功能，通过不同的环境激活不同的profile来达到构建的可移植性。

一、配置profile

首先是profile配置，在pom.xml中添加如下profile的配置：（注意用maven更新项目）

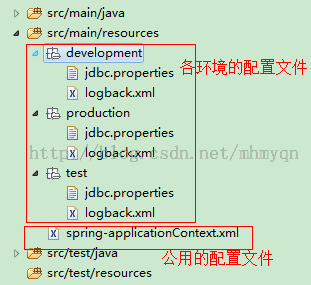
1. **<profiles>**
2. **<profile>**
3. <!-- 本地开发环境 -->
4. **<id>**development**</id>**
5. **<properties>**
6. **<profiles.active>**development**</profiles.active>**
7. **<deploy.url>**http://localhost:8080/mobileaccount**</deploy.url>**
8. **</properties>**
9. **<activation>**
10. **<activeByDefault>**true**</activeByDefault>**
11. **</activation>**
12. **</profile>**
13. **<profile>**
14. <!-- 测试环境 -->
15. **<id>**test**</id>**
16. **<properties>**
17. **<profiles.active>**test**</profiles.active>**
18. **<deploy.url>**http://localhost:8080/mobileaccount**</deploy.url>**
19. **</properties>**
20. **</profile>**
21. **<profile>**
22. <!-- 生产环境 -->
23. **<id>**production**</id>**
24. **<properties>**
25. **<profiles.active>**production**</profiles.active>**
26. **<deploy.url>**http://localhost:8080/mobileaccount**</deploy.url>**
27. **</properties>**
28. **</profile>**
29. **</profiles>**

这里定义了三个环境，分别是development（开发环境）、test（测试环境）、production（生产环境），其中开发环境是默认激活的（activeByDefault为true），这样如果在不指定profile时默认是开发环境。

同时每个profile还定义了两个属性，其中profiles.active表示被激活的profile的名称，deploy.url表示发布服务器的地址。我们需要在下面使用到这两个属性。

另外host和port分别是发布服务器的主机地址和端口号。

 二、配置文件

针对不同的环境，我们定义不同的配置文件，而这些配置文件都做为资源文件放到maven工程的resources目录下，即src/main/resources目录下，且各个环境的配置分别放到相应的目录下，而所有环境都公用的配置，直接放到src/main/resources目录下即可。如下图所示：  


如图所示，开发环境、测试环境、生产环境的配置文件分别放到src/main/resources目录下的development、test、production三个子目录中，而所有环境都公用的配置文件spring-applicationContext.xml直接放到src/main/resources目录下。其中jdbc.properties配置数据源、logback.xml配置日志。

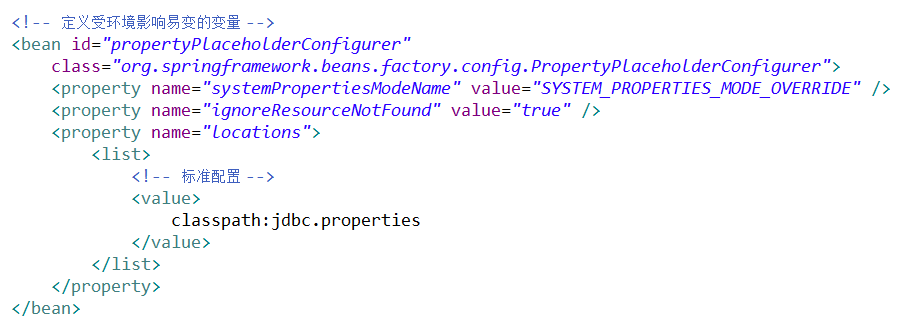
三、maven资源插件配置 并 在Spring中配置数据源

在pom中的build节点下，配置资源文件的位置，如下所示：

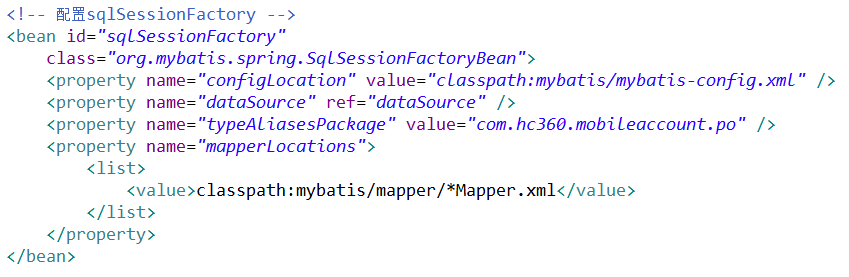
1. **<build>**
2. **<pluginManagement>**
3. **<plugins>**
4. **<!--** 跳过测试 **-->**
5. **<plugin>**
6. **<groupId>org.apache.maven.plugins</groupId>**
7. **<artifactId>maven-surefire-plugin</artifactId>**
8. **<version>2.17</version>**
9. **<configuration>**
10. **<skip>true</skip>**
11. **</configuration>**
12. **</plugin>**
13. **<!--** compiler插件, 设定JDK版本 **-->**
14. **<plugin>**
15. **<groupId>**org.apache.maven.plugins**</groupId>**
16. **<artifactId>**maven-compiler-plugin**</artifactId>**
17. **<version>**2.3.2**</version>**
18. **<configuration>**
19. **<source>**${jdk.version}**</source>**
20. **<target>**${jdk.version}<**/target>**
21. **<encoding>**${project.build.sourceEncoding}**</encoding>**
22. **</configuration>**
23. **</plugin>**
24. **</plugins>**
25. **</pluginManagement>**
26. **<resources>**
27. **<resource>**
28. **<directory>**src/main/resources**</directory>**
29. <!-- 资源根目录排除各环境的配置，使用单独的资源目录来指定 -->
30. **<excludes>**
31. **<exclude>**test/\***</exclude>**
32. **<exclude>**production/\***</exclude>**
33. **<exclude>**development/\***</exclude>**
34. **</excludes>**
35. **</resource>**
36. **<resource>**
37. <!-- ${profiles.active}取名字 -->
38. **<directory>**src/main/resources/${profiles.active}**</directory>**
39. **</resource>**
40. **</resources>**
41. **</build>**

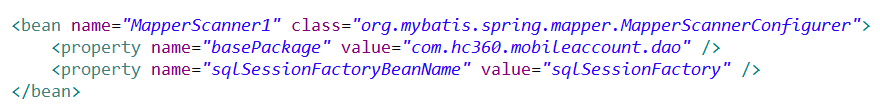
首先第一个资源文件位置src/main/resources需要排队提各个环境的配置文件，各个环境的配置我们在第二个<resource>节点中通过前面在profile中配置的profiles.active属性来指定。即src/main/resources/${profiles.active}。这样在激活指定的profile时，会加载指定目录下的配置文件，如当前激活的是production profile，那么这个资源目录就是src/main/resources/production。这样就达到了不同环境加载不同配置的目的。

在Spring中配置数据源









四、配置tomcat-maven-plugin插件（可选）

1. **<plugin>**
2. **<groupId>**org.codehaus.mojo**</groupId>**
3. **<artifactId>**tomcat-maven-plugin**</artifactId>**
4. **<version>**1.2-SNAPSHOT**</version>**
5. **<configuration>**
6. **<url>**${deploy.url}**</url>**
7. **<server>**tomcat**</server>**
8. **<path>**/appcontext**</path>**
9. **</configuration>**
10. **</plugin>**

其中发布的<url>节点就是在前面profile中配置的deploy.url属性，这样不同的环境就指定了不同的发布地址。<server>和<path>节点分别是发布服务器的用户配置的id以及应用的context名称。

 五、构建或发布

所有需要的配置就完成了，下面是见证奇迹的时候了。通过在运行maven命令时指定不同的profile即可构建不同环境需要的war包或发布到不同的环境了 。如：

mvn clean package -Pproduction即构建出生产环境需要的war包

mvn tomcat:redeploy -Ptest 即发布到测试环境

由于默认的profile是development，所以如果我们不指定profile，那么加载就是开发环境deployment下的配置文件了。即我们在本地开发测试时，不用关心profile的问题。

而且本地开发时在eclipse中使用tomcat插件来进行热部署时也不需要额外的配置。真正的做到了根据不同环境来自动切换，即可移植的构建。

另外，在进行持续集成时，使用hudson集成maven同样是非常非常方便的。