Maven构建多环境应用

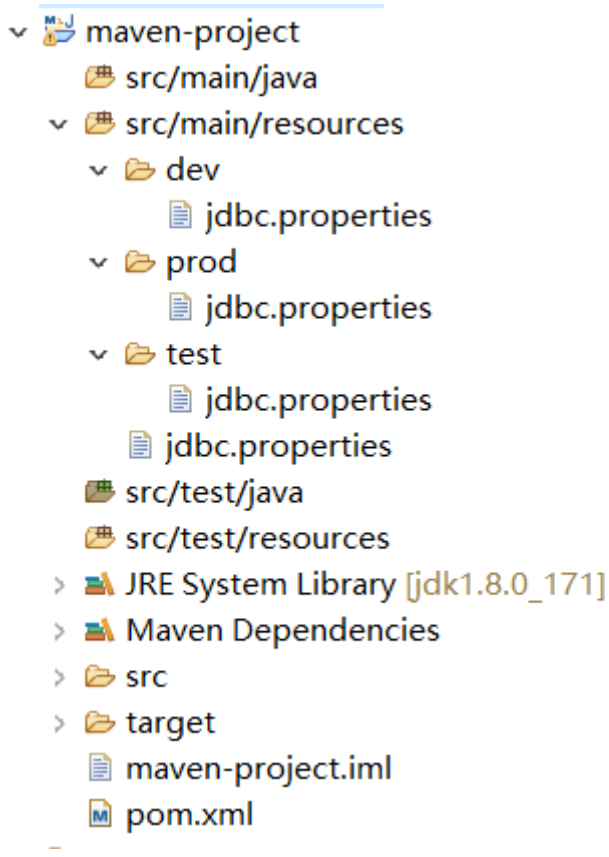
第1节 多环境的介绍

我们的项目在开发过程中往往会有多个环境,比如说项目在开发的时候使用开发环境进行测试(例如开发的数据库),测试环境,生产环境等,如果我们只有一份配置文件的话,在进行多个环境切换的时候需要手动的设置这些环境的配置参数,这样不止比较繁琐,还容易出错,所以引入了我们的Maven多环境配置

第2节 多环境配置

2.1 方式一(多环境对应多配置文件)

针对项目怎么进行多环境配置,接下来看下面的项目结构



这是使用maven工具创建的一个web工程,在src/main/resources文件夹下创建了多个文件夹(每个文件夹里放的是不同环境的配置)和一个默认的配置文件作为演示.

在进行多环境打包之前,首先我们要知道一些maven的基本常识,在Java项目中上面展示的文件夹结构

```
    src/main/java
    src/main/resources
    src/test/java
    src/test/resources
```

是maven工具创建JavaWeb项目默认的官方指定的结构,一般情况我们不会手动的去改变,不同的文件夹放置的文件是不同的,比如java文件夹放置我们的Java源代码,resources文件夹一般放置项目中用到的配置文件.所以这里将多环境的配置文件都放在src/main/resources文件夹目录下,并且为了区分多个环境的配置文件,这里自定义了三个文件夹(dev、prod、test),放置不同环境的配置文件.在src/main/resources根目录下还有一个jdbc.properties配置文件，作为默认的配置文件.就是在我们打包的时候如果没有选择任何环境的配置文件,那么走这个默认的.

maven的打包命令:

```
mvn package
或者
mvn install
```

maven工具在执行上面的打包命令时,默认的将src/main/resources文件夹下的所有资源文件都打到war包中,所以要想打指定的配置,那么需要一系列的配置,具体配置如下:

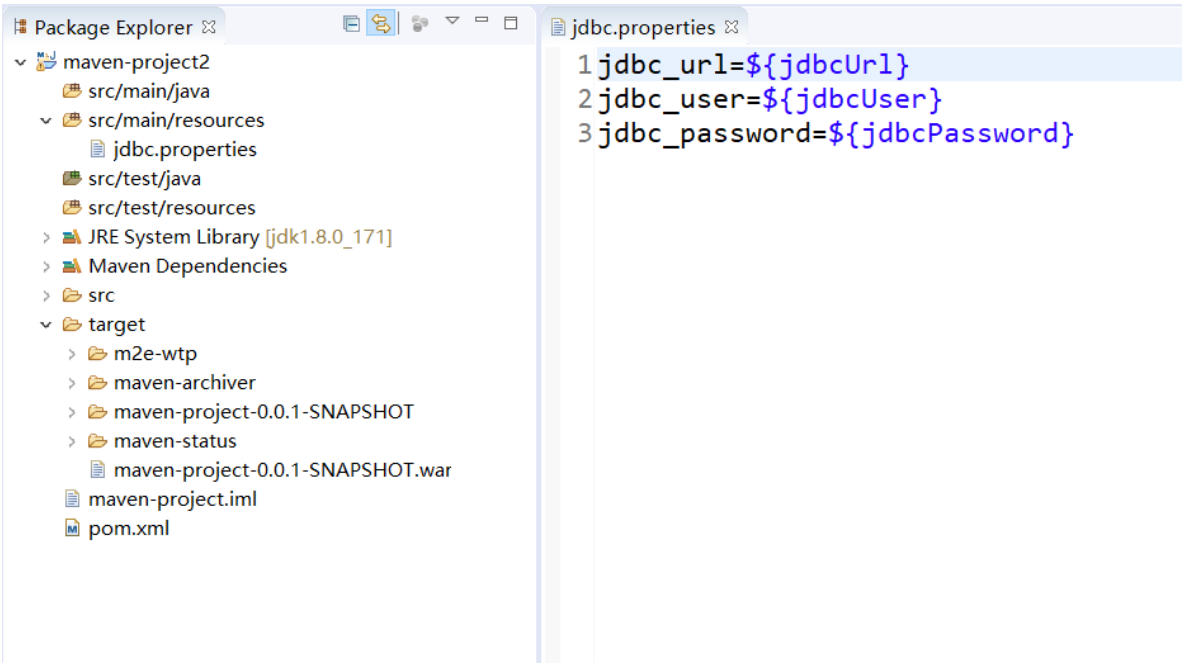
> pom.xml配置如下:

```
<build>
    <finalName>maven-dev-pro-test</finalName>
    <resources>
        <resource>
            <directory>src/main/resources</directory>
            <!--
     排除默认的包含,
     在Maven打包时,默认将resources包下的所有资源进行打包
     而我们需要进行选择性打包,将指定环境配置打到包中
     -->
            <excludes>
                <!--
      resources文件夹下默认的资源会在使用maven打包命名时直接全部进行打包
      而我们想分类打包,根据不同的参数将不同环境的配置文件打包,所以将下面的一些资源先排除掉
     -->
                <exclude>dev/*</exclude>
                <exclude>test/*</exclude>
                <exclude>prod/*</exclude>
            </excludes>
        </resource>
        <resource>
            <!--
     env: profile中自定义的一个标签变量
     ${env}: 在pom.xml中可以取自定义变量的值
     -->
            <directory>src/main/resources/${env}</directory>
        </resource>
    </resources>
</build>
<!--
  多环境打包配置
  -->
<profiles>
    <profile>
        <!-- dev: 自定义名称,代表开发环境 -->
        <id>dev</id>
        <properties>
            <!--
     <env>: 自定义标签参数
```

```
        dev: 自定义标签参数值
        -->
                <env>dev</env>
            </properties>
            <activation>
                <!-- 默认激活该 profile 也就是dev环境-->
                <activeByDefault>true</activeByDefault>
            </activation>
        </profile>
        <profile>
            <!-- test: 自定义名称,代表测试环境 -->
            <id>test</id>
            <properties>
                <env>test</env>
            </properties>
        </profile>
        <profile>
            <!-- prod: 自定义名称,代表线上生产环境 -->
            <id>prod</id>
            <properties>
                <env>prod</env>
            </properties>
        </profile>
    </profiles>
```

2.2 方式二(多环境对应一个配置文件)

当使用单个配置文件配置多环境的时候，需要定义变量,在进行打包时候给变量赋值，然后修改配置文件中的配置,配置事例如下。



> pom.xml配置文件如下:

```
<build>
    <resources>
        <resource>
            <!-- 资源配置文件的位置 -->
```

```
            <directory>src/main/resources</directory>
            <!-- 是否替换资源中的属性 -->
            <filtering>true</filtering>
        </resource>
    </resources>
</build>
<profiles>
    <profile>
        <id>dev</id>
        <!-- 自定义变量,在打包时候设置jdbc.properties中的变量值 -->
        <properties>
            <jdbcUrl>jdbc:mysql://192.168.1.100:3306/dev</jdbcUrl>
            <jdbcUser>root</jdbcUser>
            <jdbcPassword>dev</jdbcPassword>
        </properties>
        <activation>
            <activeByDefault>true</activeByDefault>
        </activation>
    </profile>
    <profile>
        <id>test</id>
        <properties>
            <jdbcUrl>jdbc:mysql://192.168.1.100:3306/test</jdbcUrl>
            <jdbcUser>root</jdbcUser>
            <jdbcPassword>test</jdbcPassword>
        </properties>
    </profile>
    <profile>
        <id>prod</id>
        <properties>
            <jdbcUrl>jdbc:mysql://192.168.1.100:3306/prod</jdbcUrl>
            <jdbcUser>root</jdbcUser>
            <jdbcPassword>prod</jdbcPassword>
        </properties>
    </profile>
</profiles>
```

2.3 maven启动多环境打包方式

```
mvn clean install -Pdev -Dmaven.test.skip=true
mvn clean package -Ptest -Dmaven.test.skip=true
```