1. Maven
2. 概述

Maven是一个项目管理工具，它包含了一个项目对象模型，一组标准集合，一个项目生命周期，一个依赖管理系统和用来运行定义在生命周期中插件目标的逻辑。

1. 作用

构建工程，管理jar包，编译代码，自动运行单元测试，打包，生成报表，部署项目，生成Web站点。

1. Maven的依赖管理

传统的web项目的jar包是包含在项目中的，Maven项目的jar包是不包含在项目中的，所有的jar包存储在一个jar包仓库中，项目中只存储jar包的坐标，项目通过jar包的坐标来调用jar包。

1. Maven的项目构建

项目的构建指的是项目从编译，测试，运行，打包，安装到部署的整个过程。Maven可以使用一个命令mvn轻松完成。

1. Maven目录分析
2. bin目录

可执行文件，其中有构建项目的mvn命令。

1. boot目录

maven自身运行所需要的类加载器。

1. conf目录

存储配置文件的。

4． lib目录

依赖的jar包。

1. jar包仓库
2. 仓库的种类

Maven项目的jar包仓库分为本地仓库，中央仓库和远程仓库。本地仓库指的是在自己硬盘上的仓库，中央仓库是互联网上的仓库，远程仓库是所处局域网搭建的仓库（私服），如果私服中没有想要的jar 包，则可以从中央仓库中下载，也可以从本地仓库中上传。工作环境中一般使用的是公司搭建的远程仓库，只要是处于同一个局域网下，就可以访问到远程仓库。

1. 修改本地仓库路径

在Maven的安装文件夹的conf目录下，有一个settings.xml配置文件，找到”localRepository”关键字，默认在C盘中自动创建，我们可以在下下面一行中修改为自己的磁盘路径。如：

<localRepository>D:\JAVA\apache-maven-3.5.2\Repository</localRepository>

1. Maven项目的标准目录规范
2. 核心代码

核心代码应该存放在 src/main/java目录下。

1. 配置文件

核心代码的配置文件应该放置在src/main/resources目录下。

1. 测试代码

测试代码应该放置在 src/test/java目录下。

1. 测试配置文件

测试配置文件应该放置在 src/test/resources目录下。

1. web页面资源

web的页面资源如js,css,图片等应该放置在 src/main/webapp目录下。

1. Maven常用命令
2. mvn clean

在该项目文件夹下执行clean命令，会将该项目已编译的部分删除，有助于项目的移植。

1. mvn compile

编译项目，将编译后的文件放置在target目录下。

1. mvn test

编译测试代码，且同时一起编译了核心代码。

1. mvn package

打包项目，同时编译了核心代码和测试代码，并打包成了war包存储在target目录下。

1. mvn install

编译了核心代码和测试代码，打包了项目，同时将该war包安装到了本地仓库中。

1. Maven概念模型
2. 项目对象模型

POM，可以通过Maven项目文件夹中的pom.xml文件来查看。该文件中包含了项目自身的信息，项目所依赖的jar包信息，项目运行环境信息，如jdk,tomcat信息。

1. 依赖管理模型

项目运行所依赖的jar包信息。如公司组织的名称groupId，项目名称artifactId，版本号version。

<dependency>

<groupId>javax.servlet.jsp<groupId>

<artifactId>jsp-api</artifactId>

<version>2.0</version>

</dependency>

依赖管理模型中有三个仓库，每次请求jar包时首先请求本地仓库，如果本地仓库中没有就接着像中央仓库请求，如果没有联网，则向远程仓库请求，如果远程仓库中没有，则从中央仓库下载或者从本地仓库中上传。

1. IDEA中创建Maven项目
2. 通过骨架创建Maven项目
3. 新建项目Project，选择Maven项目，右侧是IDEA提供的Maven项目模板，我们可以勾选 “Create from archetype”使用这些模板，当然使用的前提是已联网或者设置IDEA的Maven下的Runner的VM-option为 -Darchtype=Internal。
4. 设置组名，项目名，版本号，Maven的目录位置，Maven的设置Settings.xml位置，Maven的本地仓库位置。
5. 等待IDEA加载，最终提示项目生成成功。
6. 生成Maven项目后，应该有.idea目录和src目录，src目录存放着main和test目录，main目录下存放着java目录和resources目录，resources目录是存放配置文件的，需要我们手动添加，并设置该目录为 resources root，方法点击右键，选择 make Directory as，选择Resources Root。
7. 项目完全创建完成。
8. 不通过骨架创建Maven项目

不勾选 “create from archtype”，直接创建项目即可。

1. 使用骨架创建Maven的web项目

选用webapp模板，创建Maven项目，需要我们自己创建src/main/java目录和src/test/java目录，同时设置这些目录为源码路径，方法是点击右键，选择 make Directory as ，选择 Sources Root。

还有resousrce配置文件目录需要手动添加。

注意，创建web项目有很多的坑，比如在加载框架的时候会非常的慢，这可能是因为Maven的镜像源的问题，我们可以设置Maven的镜像源为国内的镜像源，如阿里的Maven镜像，配置方法如下：

打开 maven 的配置文件（ windows 机器一般在 maven 安装目录的**conf/settings.xml** ），在<mirrors></mirrors>标签中添加 mirror 子节点:

<mirror>

<id>aliyunmaven</id>

<mirrorOf>\*</mirrorOf>

<name>阿里云公共仓库</name>

<url>https://maven.aliyun.com/repository/public</url>

</mirror>

若还是太慢了，可以在创建maven项目时添加参数键值对 archetypeCatalog ，值为internal，代表着加载本地的数据。同时注意选择模板不要选错了模板，要选择archetypes:maven-archetype-webapp，而不是cocoon-22-archetype-webapp！！！

1. Maven Web项目的配置
2. web.xml的配置

当我们想在src/main/java下添加一个Servlet文件时，web.xml配置文件自动生成了<servlet>配置信息，但需要我们手动添加Servlet的访问路径，如下所示：

<servlet>

<servlet-name>MyServlet</servlet-name>

<servlet-class>com.test.MyServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>MyServlet</servlet-name>

<url-pattern>/</url-pattern>

</servlet>

1. pom.xml

配置完web.xml后，需要配置项目的依赖信息，我们需要在pom.xml文件中配置依赖jar包的坐标，即设置dependency，如下所示：

<dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>servlet-api</artifactId>  
 <version>2.5</version>  
</dependency>  
  
<dependency>  
 <groupId>javax.servlet.jsp</groupId>  
 <artifactId>jsp-api</artifactId>  
 <version>2.0</version>  
</dependency>

这里能够自动提示的都是已经存储在本地仓库中的jar包，如果我们想要的jar 包本地仓库中没有，那么我们可以在百度中搜索“Maven中央仓库”，将dependency字段直接复制到pom.xml中，它会从网络中自动下载。

1. 执行Maven指令

项目配置好了，想要运行项目，就要通过Maven指令来操作。IDEA的右上角有一个Maven，点开可以看到选项有生命周期，即Maven项目的步骤命令，如clean，compile；我们可以直接双击执行这些命令。

1. 配置运行环境

为了配置Maven的运行环境，如Tomcat，JDK，我们可以在pom.xml的配置文件中配置插件信息，导入插件，如下所示，导入tomcat插件：

注意，是在<build><plugins>标签中

<!-- tomcat插件控制 -->  
<plugin>  
<groupId>org.apache.tomcat.maven</groupId>  
<artifactId>tomcat7-maven-plugin</artifactId>  
<version>2.2</version>  
<configuration>  
 <port>  
 8000  
 </port>  
 <path>/</path>  
</configuration>  
</plugin>

导入插件之后，就可以在Maven的命令菜单的plugins插件中看到新的插件，同样是双击即可使用。

1. jar包冲突

所谓的jar包冲突指的是我们在项目中导入的jar包和Maven中集成的插件中的jar包相冲突，我们可以通过设置项目中的jar包的作用范围为仅编译时生效即可。因为项目中必须要导入相关的包，否则会报错，限制了jar包的作用范围为编译后，项目中的jar包仅在编译时生效，而Maven中集成的插件中的jar包就在项目执行时生效，这样jar包的冲突问题就解决了。设置jar包的作用范围如下所示：

<scope>provided</scope>

其中设置的范围有以下几种：

compile 对于编译，测试，运行时有效，如 spring-core

test 对于测试有效，如Jnuit

provided 对于编译，测试有效, 如 servlet-api

runtime 对于测试，运行时有效，如JDBC驱动

system 对于编译，测试有效，如本地仓库之外的类库