

Maven-项目管理工具

Maven简介

- Maven这个词翻译为(“ 专家” ,“ 内行”), 是跨平台的项目管理工具。
- Maven主要服务于基于Java平台的**项目构建** , **依赖管理** , **项目信息管理**。

项目构建

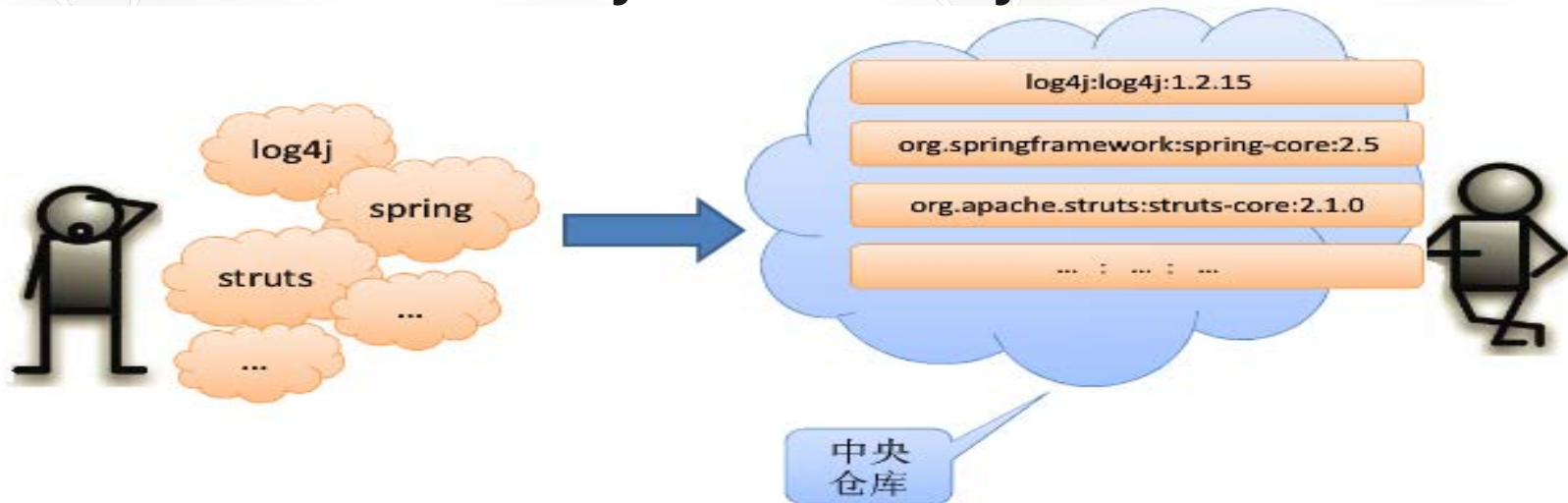
- 项目构建过程包括：
- 【清理项目】→【编译项目】→【测试项目】→【生成测试报告】→【打包项目】→【部署项目】这几个步骤，这六个步骤就是一个项目的完整构建过程。



- 理想的项目构建是高度自动化，跨平台，可重用的组件，标准化的，使用maven就可以帮我们完成上述所说的项目构建过程。

依赖管理

- 依赖指的是jar包之间的相互依赖，比如我们搭建一个Struts2的开发框架时，光光有struts2-core-2.3.16.3.jar这个jar包是不行的，struts2-core-2.3.16.3.jar还依赖其它的jar包，依赖管理指的就是使用Maven来管理项目中使用到的jar包，Maven管理的方式就是“自动下载项目所需要的jar包，统一管理jar包之间的依赖关系”。



项目信息管理

- 工程版本控制系统消息
- 缺陷跟踪系统消息
- 开发者信息
- 许可证信息
- Javadoc
- 测试覆盖
- 代码静态分析报告
- 单元测试覆盖率

Maven下载与安装

Maven的安装-windows安装Maven

- 检查JDK安装
- 首先确保电脑上安装JDK(1.6+)的，Maven运行在JDK之上，运行如下命令来检查Java的安装。
- C:\Users\mantis> echo %JAVA_HOME%
C:\Users\mantis> java -version

```
C:\Users\mantis>echo %JAVA_HOME%
C:\Program Files\Java\jdk1.7.0_06

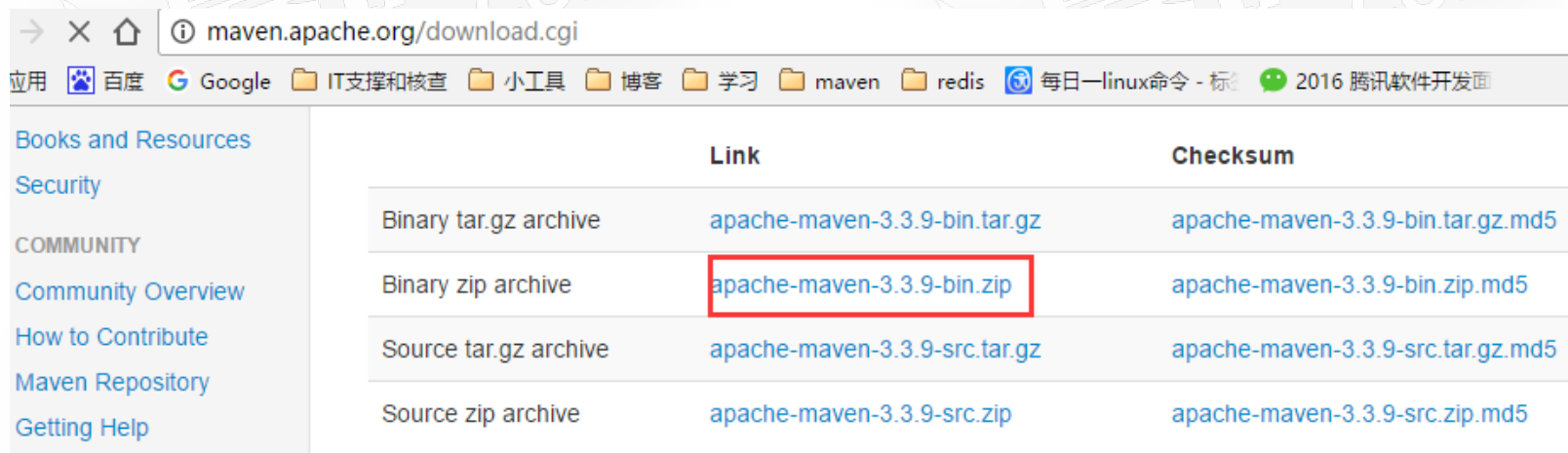
C:\Users\mantis>java -version
java version "1.8.0_77"
Java(TM) SE Runtime Environment (build 1.8.0_77-b03)
Java HotSpot(TM) 64-Bit Server VM (build 25.77-b03, mixed mode)

C:\Users\mantis>_
```

上述命令首先检查环境变量JAVA_HOME是否指向了正确的JDK目录，接着尝试运行Java命令。如果Windows无法执行java命令，就需要检查java是否安装了。

Maven的下载

下载地址：<http://maven.apache.org/download.cgi>

A screenshot of a web browser displaying the Maven download page. The browser's address bar shows the URL 'maven.apache.org/download.cgi'. The page has a sidebar on the left with links like 'Books and Resources', 'Security', and 'COMMUNITY'. The main content area features a table with three columns: 'Link' and 'Checksum'. The table lists four download options: 'Binary tar.gz archive', 'Binary zip archive', 'Source tar.gz archive', and 'Source zip archive'. The 'Binary zip archive' row is highlighted with a red rectangle, and its link 'apache-maven-3.3.9-bin.zip' is also highlighted with a red rectangle.

	Link	Checksum
Binary tar.gz archive	apache-maven-3.3.9-bin.tar.gz	apache-maven-3.3.9-bin.tar.gz.md5
Binary zip archive	apache-maven-3.3.9-bin.zip	apache-maven-3.3.9-bin.zip.md5
Source tar.gz archive	apache-maven-3.3.9-src.tar.gz	apache-maven-3.3.9-src.tar.gz.md5
Source zip archive	apache-maven-3.3.9-src.zip	apache-maven-3.3.9-src.zip.md5

Maven的下载

→ × ① maven.apache.org/download.cgi

应用 百度 Google IT支撑和核查 小工具 博客 学习 maven redis 每日一linux命令 - 标 2016 腾讯软件开发面

Books and Resources

Security

COMMUNITY

Community Overview

How to Contribute

Maven Repository

Getting Help

	Link	Checksum
Binary tar.gz archive	apache-maven-3.3.9-bin.tar.gz	apache-maven-3.3.9-bin.tar.gz.md5
Binary zip archive	apache-maven-3.3.9-bin.zip	apache-maven-3.3.9-bin.zip.md5
Source tar.gz archive	apache-maven-3.3.9-src.tar.gz	apache-maven-3.3.9-src.tar.gz.md5
Source zip archive	apache-maven-3.3.9-src.zip	apache-maven-3.3.9-src.zip.md5

Maven的下载

- 解压 压缩包，可以看到Maven的组成目录。

技术 > maven > maven培训 > apache-maven > apache-maven-3.3.9-bin > apache-maven-3.3.9

名称	修改日期	类型	大小
bin	2017/1/3 17:43	文件夹	
boot	2017/1/3 17:43	文件夹	
conf	2017/1/3 17:43	文件夹	
lib	2017/1/3 17:43	文件夹	
LICENSE	2015/11/10 11:44	文件	19 KB
NOTICE	2015/11/10 11:44	文件	1 KB
README.txt	2015/11/10 11:38	TXT 文件	3 KB

- Maven目录分析
- bin：含有mvn运行的脚本
- boot：含有plexus-classworlds类加载器框架
- conf：含有settings.xml配置文件
- lib：含有Maven运行时所需要的java类库
- LICENSE.txt, NOTICE.txt, README.txt针对Maven版本，第三方软件等简要介绍

Maven的下载

- 解压 压缩包，可以看到Maven的组成目录。

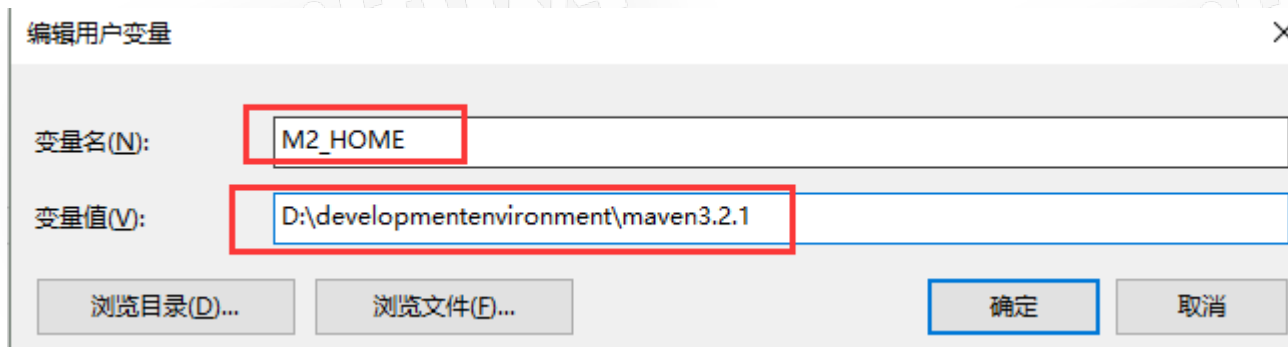
技术 > maven > maven培训 > apache-maven > apache-maven-3.3.9-bin > apache-maven-3.3.9

名称	修改日期	类型	大小
bin	2017/1/3 17:43	文件夹	
boot	2017/1/3 17:43	文件夹	
conf	2017/1/3 17:43	文件夹	
lib	2017/1/3 17:43	文件夹	
LICENSE	2015/11/10 11:44	文件	19 KB
NOTICE	2015/11/10 11:44	文件	1 KB
README.txt	2015/11/10 11:38	TXT 文件	3 KB

- Maven目录分析
- bin：含有mvn运行的脚本
- boot：含有plexus-classworlds类加载器框架
- conf：含有settings.xml配置文件
- lib：含有Maven运行时所需要的java类库
- LICENSE.txt, NOTICE.txt, README.txt针对Maven版本，第三方软件等简要介绍

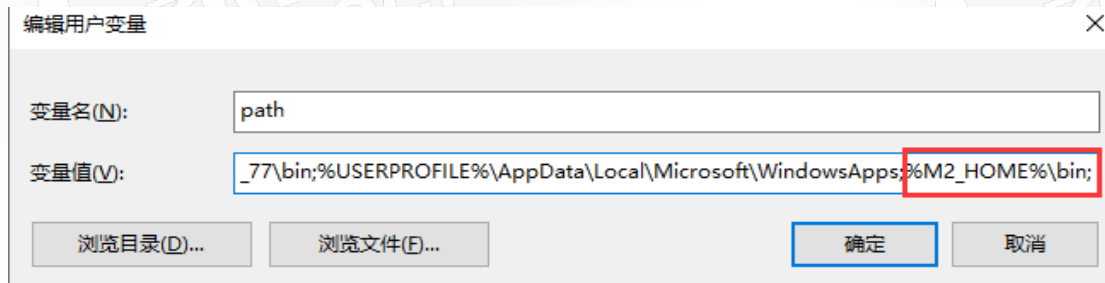
本地安装

- 新增环境变量M2_HOME，如下图所示：



The screenshot shows the 'Edit User Variable' dialog box. The 'Variable name (N):' field contains 'M2_HOME' and is highlighted with a red box. The 'Variable value (V):' field contains 'D:\developmentenvironment\maven3.2.1' and is also highlighted with a red box. At the bottom, there are buttons for 'Browse directory (D)...', 'Browse file (F)...', 'OK' (highlighted with a blue box), and 'Cancel'.

- 设置环境变量Path，将%MAVEN_HOME%\bin加入Path中，一定要注意要用分号；与其他值隔开，如下图所示：



The screenshot shows the 'Edit User Variable' dialog box for the 'Path' variable. The 'Variable name (N):' field contains 'path'. The 'Variable value (V):' field contains '_77\bin;%USERPROFILE%\AppData\Local\Microsoft\WindowsApps;%M2_HOME%\bin;' and is highlighted with a red box. At the bottom, there are buttons for 'Browse directory (D)...', 'Browse file (F)...', 'OK' (highlighted with a blue box), and 'Cancel'.

- 新开一个cmd窗口，运行如下命令检查Maven的安装情况。

```
C:\Users\mantis>mvn -v
Apache Maven 3.2.1 (ea8b2b07643dbb1b84b6d16elf08391b666bc1e9; 2014-02-15T01:37:52+08:00)
Maven home: D:\developmentenvironment\maven3.2.1
Java version: 1.8.0_77, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.8.0_77\jre
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 10", version: "10.0", arch: "amd64", family: "dos"
C:\Users\mantis>
```

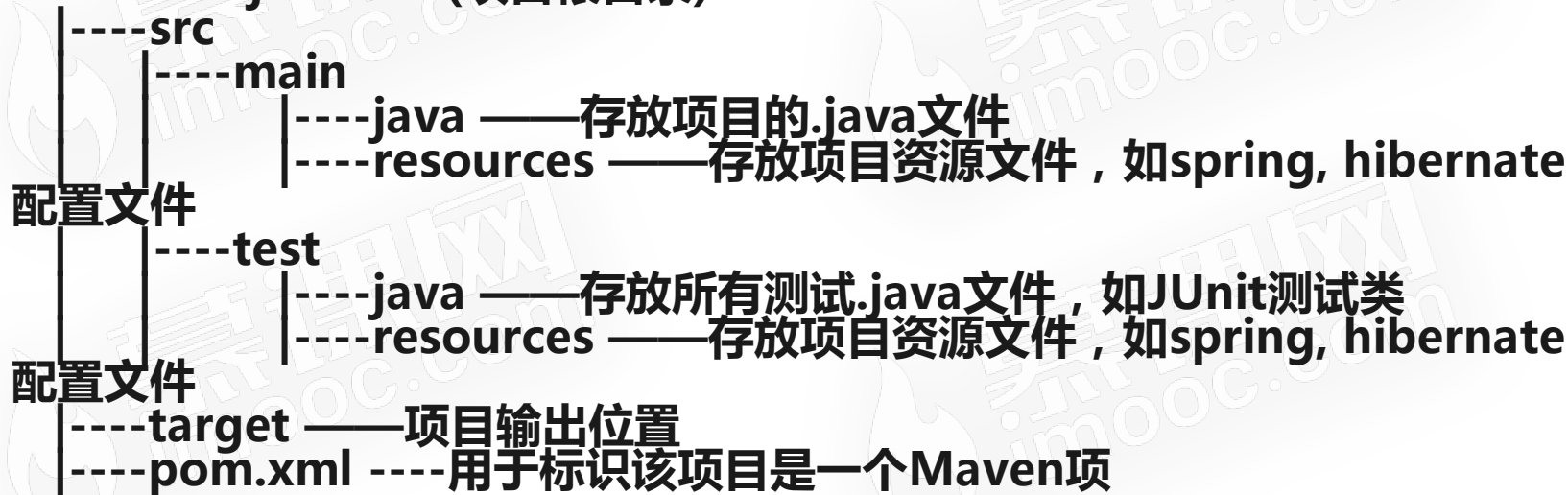
- 能够出现这样的信息就说明Maven的安装已经成功了。

Maven手动和自动构建项目,编译,清除,打包

Maven手动和自动构建项目,编译,清除,打包

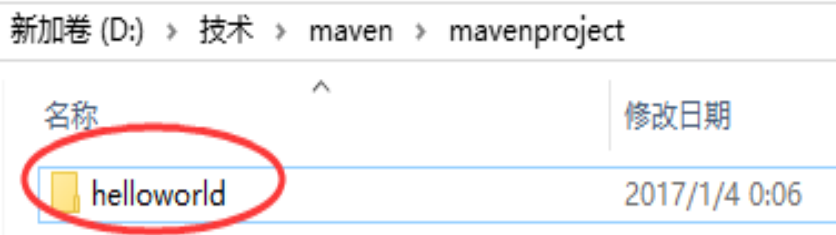
Maven项目的目录约定

- MavenProjectRoot(项目根目录)



手动创建Maven项目，使用Maven编译

1、创建helloworld文件夹，如下图所示：

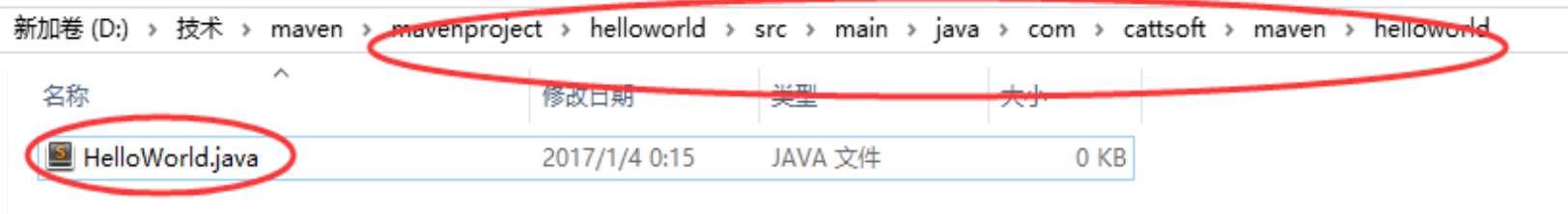


2、打开文件夹，新建一个名为pom.xml的文件，输入如下内容：

```
1 <project xmlns="http://maven.apache.org/POM/4.0.0"
2   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.cattsoft.maven</groupId> <!--groupId指的是项目名的项目组，默认就是包名-->
8   <artifactId>helloworld</artifactId> <!-- 项目名称 -->
9   <version>0.0.1-SNAPSHOT</version>
10  <name>helloworld</name>
11  <url>http://maven.apache.org</url>
```


编写主代码

- 项目的主代码和测试代码不同，主代码会被打包到最终的构件中(如jar)，而测试代码只在测试时用到，不会被打包。默认情况下，主代码位于src/main/java目录，我们遵循Maven约定，创建该目录，然后在该目录下创建文件src/main/java/com/cattsoft/maven/helloworld/HelloWorld.java



- HelloWorld 的主代码如下图。


```
3 // **
4 * Hello world!
5 *
6 */
7 public class HelloWorld
8 {
9     public static void main( String[] args )
10     {
11         System.out.println( "Hello World!" );
12     }
13 }
```

编译执行

- 在项目根目录下执行命令：`mvn compile`

```
D:\技术\maven\mavenproject\helloworld>mvn compile
[INFO] Scanning for projects...
[INFO]
[INFO] Using the builder org.apache.maven.lifecycle.internal.builder.singlethreaded.
count of 1
[INFO]
[INFO] -----
[INFO] Building helloworld 0.0.1-SNAPSHOT
[INFO] -----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ helloworld ---
[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. bu
[INFO] skip non existing resourceDirectory D:\技术\maven\mavenproject\helloworld\src
[INFO]
[INFO] --- maven-compiler-plugin:2.5.1:compile (default-compile) @ helloworld ---
[WARNING] File encoding has not been set, using platform encoding GBK, i.e. build is
[INFO] Compiling 1 source file to D:\技术\maven\mavenproject\helloworld\target\class
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO]
[INFO] Total time: 1.090 s
[INFO] Finished at: 2017-01-04T00:27:04+08:00
[INFO] Final Memory: 13M/220M
[INFO] -----
D:\技术\maven\mavenproject\helloworld>
```

- 编译成功生产target目录，以及编译好的.class字节码文件。

新加卷 (D:) > 技术 > maven > mavenproject > helloworld > target > classes > com > cattsoft > maven > helloworld				
名称	修改日期	类型	大小	
 HelloWorld.class	2017/1/4 0:27	CLASS 文件	1 KB	

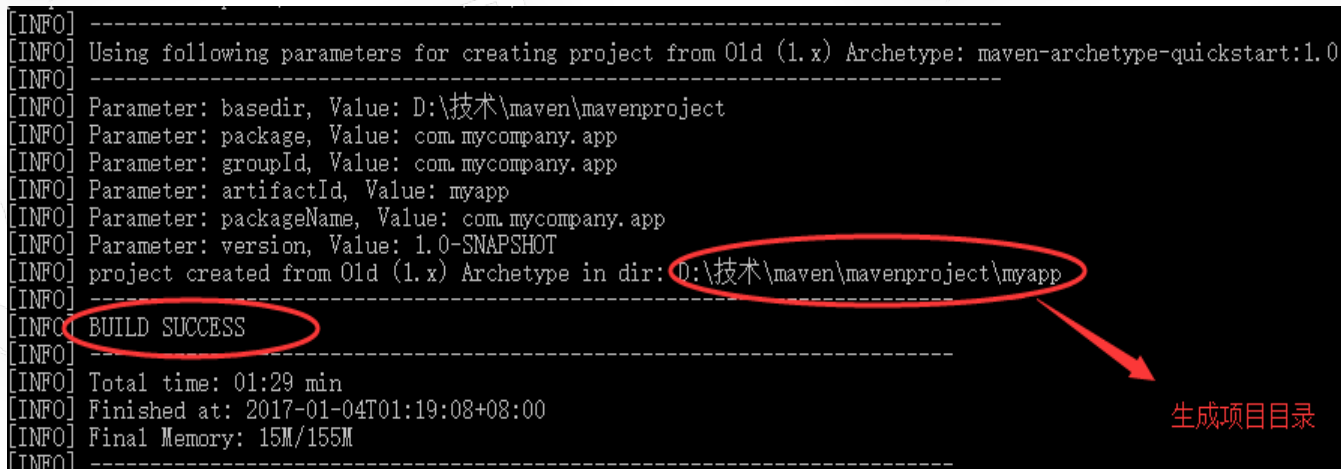
使用“**mvn clean**”命令清除编译结果，也就是把编译生成的target文件夹删掉。

```
D:\技术\maven\mavenproject\helloworld>mvn clean
[INFO] Scanning for projects...
[INFO]
[INFO] Using the builder org.apache.maven.lifecycle.internal.builder.singlethreaded.
count of 1
[INFO]
[INFO] -----
[INFO] Building helloworld 0.0.1-SNAPSHOT
[INFO] -----
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ helloworld ---
[INFO] Deleting D:\技术\maven\mavenproject\helloworld\target
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 0.288 s
[INFO] Finished at: 2017-01-04T00:29:51+08:00
[INFO] Final Memory: 6M/155M
[INFO] -----
D:\技术\maven\mavenproject\helloworld>
```

Maven自动构建Java Project项目

- maven作为一个高度自动化构建工具，本身提供了构建项目的功能，下面就来体验一下使用maven构建项目的过程。
- 构建java项目：
- 1、使用mvn archetype:generate，命令如下所示：
- **mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=myapp -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false**
- 2、使用"mvn archetype:create"命令创建一个java项目的过程如下图所示：

```
[INFO] -----
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.0
[INFO] -----
[INFO] Parameter: basedir, Value: D:\技术\maven\mavenproject
[INFO] Parameter: package, Value: com.mycompany.app
[INFO] Parameter: groupId, Value: com.mycompany.app
[INFO] Parameter: artifactId, Value: myapp
[INFO] Parameter: packageName, Value: com.mycompany.app
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: D:\技术\maven\mavenproject\myapp
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:29 min
[INFO] Finished at: 2017-01-04T01:19:08+08:00
[INFO] Final Memory: 15M/155M
[INFO] -----
```



生成项目目录

完结

在实际开发中，我们可以利用一些现成的验证码框架进行搭建会更快更有效率，但是对知识的追求我们应该保持不断探索的精神。