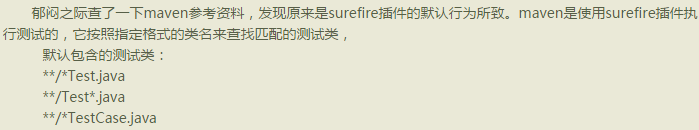
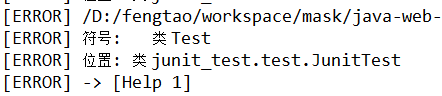
1. 很多情况下,我们可能会写junit测试



我们在执行 package 或者 install命令时 会自动执行maven test命令.



就会提示找不到这个类,原因是 junit的包scope是test,并不会打包到项目中,

解决办法有两个:1、把scope去掉;2、利用maven-surefire-plugin插件(更新完记得update maven项目生效)

<!-- maven-surefire-plugin -->

<!-- eclipse整合maven之怎么打包的时候跳过测试junit -->

<plugin>

<inherited>true</inherited>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.19.1</version>

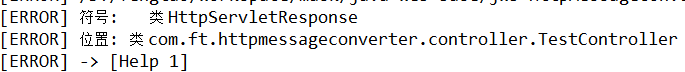
<configuration>

<skip>true</skip>跳过测试

</configuration>

</plugin>

二、



这种情况一般是我们在controller中用到了HttpServletRequest 或 HttpServletResponse 一般是容器(tomcat提供的),因此pom我们也没有写这个依赖,但是不写又报错

解决办法是:

<!--javax.servlet-api-->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

附:

\* compile，缺省值，适用于所有阶段，会随着项目一起发布。   
    \* provided，类似compile，期望JDK、容器或使用者会提供这个依赖。如servlet.jar。   
    \* runtime，只在运行时使用，如JDBC驱动，适用运行和测试阶段。   
    \* test，只在测试时使用，用于编译和运行测试代码。不会随项目发布。   
    \* system，类似provided，需要显式提供包含依赖的jar，Maven不会在Repository中查找它。

三、我们在打包一个工程时,如果依赖的外部工程A、B我们首先需要对A,B install操作(如果本地库或者私库没有A,B的jar话)这样未免显得有些麻烦.因此:,

对于复杂的父子maven工程 我们可以使用对整个项目进行install 然后在进行打包。