一次关于 maven 包冲突问题解决记录

1. 问题

遇到一个问题本地IDE跑没有问题，打包到测试环境就会报空指针。 代码是一段基础库代码，无法修改:

```
Trans trans = null;
try{
     trans = getTrans(T.createTrans(param1,param2))
     trans.do()
}finally{
     trans.setStatus(...)
}
```

`T.createTrans(param1,param2)` 是不可能返回null的。 但 finally中的`trans.setStatus(...)`报空指针了,覆盖了原本的错误。 百思不得姐啊!

2. 解决过程

2.1 定位问题

后来寻思必然是 `T.createTrans(param1,param2)`出错了， debug发现并不会进入`T.createTrans(param1,param2)`方法，于是想到T没办法实例化， 没办法实例化可能是类找不到。于是 debug窗口进行evaluate执行一段代码`Class.forName("com.mycompany.T")`, 果然报错，报错原因是`org.codehaus.plexus.component.repository.exception.ComponentLookupException` 找不到定义报`NoClassDefFoundError`错误。

2.2 查找原因

在项目中， 这个类在 `plexus-container-default-1.6:jar` 中, 执行 `mvn dependency:tree` 查看依赖关系是:

```
+- com.mycompany.framework:base-dao-jdbcPool:jar:5.0.602:compile

  +- com.dianping.cat:cat-client:jar:1.955-idc:compile (version
selected from constraint [1.954-idc,1.997-idc-SNAPSHOT))
     +- org.unidal.framework:foundation-service:jar:2.3.0:compile
        +- org.codehaus.plexus:plexus-container-
default:jar:1.6:compile
```

难到打包没有包含这个jar包，发现有打包此文件。 项目是一个spring-boot项目，打包的项目project.jar中通过MANIFEST.MF设置Class-Path指定依赖jar，发现Class-Path中的版本不一样，是 `plexus-container-default-1.0-alpha-9.jar`，打包出来根本没有这个jar包,自然没有加入到classpath中，所以找不到。

2.3 依赖jar打包方式

依赖包拷贝是通过 `maven-assembly-plugin` 进行打包的：

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-assembly-plugin</artifactId>
    <configuration>
        <descriptors>
            <descriptor>src/main/assembly/assembly.xml</descriptor>
        </descriptors>
    </configuration>
    <executions>
        <execution>
            <id>make-assembly</id>
            <phase>package</phase>
            <goals>
                <goal>single</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

assembly.xml定义如下，拷贝 `scope=runtime` 的依赖包:

```
<assembly xmlns="http://maven.apache.org/ASSEMBLY/2.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/ASSEMBLY/2.0.0
http://maven.apache.org/xsd/assembly-2.0.0.xsd">
    <includeBaseDirectory>false</includeBaseDirectory>
    <formats>
        <format>dir</format>
    </formats>
    <fileSets>
        <fileSet>
            <directory>${project.build.directory}</directory>
            <outputDirectory>lib</outputDirectory>
            <includes>
                <include>*.jar</include>
            </includes>
        </fileSet>
    </fileSets>
    <dependencySets>
        <dependencySet>
            <outputDirectory>lib</outputDirectory>
            <scope>runtime</scope>
            <excludes>
                <exclude>${groupId}:${artifactId}</exclude>
            </excludes>
        </dependencySet>
    </dependencySets>
</assembly>
```

2.3 Class-Path 信息生成方式

项目jar包是通过 `maven-jar-plugin` 指定 `addClasspath` 进行打包, 因为没有配置 `manifestEntries`，则按照依赖关系自动将依赖包添加到 Class-Path属性中：

```
 <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-jar-plugin</artifactId>
    <version>3.1.1</version>
    <configuration>
        <archive>
            <manifest>
                <addClasspath>true</addClasspath>
                <mainClass>${start-class}</mainClass>
            </manifest>
        </archive>
    </configuration>
</plugin>
```

经查是 `jacoco-maven-plugin` 依赖了 `plexus-container-default-1.0-alpha-9.jar` 依赖关系如下,

```
[INFO] org.jacoco:jacoco-maven-plugin:maven-plugin:0.8.4
[INFO] +- org.apache.maven.shared:file-management:jar:1.2.1:compile
[INFO] |  +- org.codehaus.plexus:plexus-container-default:jar:1.0-
alpha-9:compile
```

将`plexus-container-default` 从 `jacoco-maven-plugin`中剔除依赖：

```
 <dependency>
    <groupId>org.jacoco</groupId>
    <artifactId>jacoco-maven-plugin</artifactId>
    <version>0.8.4</version>
    <scope>test</scope>
    <exclusions>
        <exclusion>
            <groupId>org.codehaus.plexus</groupId>
            <artifactId>plexus-container-default</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```

再重新打包，Class-Path信息就正确包含 `plexus-container-default-1.6:jar` 了。

> 由此可见 `maven-jar-plugin` 生成Class-Path信息的maven的最终依赖关系和 `maven-assembly-plugin` 插件的 `runtime` 依赖关系并不一致。 具体差异欢迎大家指教。

3. 总结

解决此问题过程中，逐步定义到具体问题： `代码逻辑问题 -> 缺少class问题 -> 打包问题 -> 依赖问题 -> 包冲突问题`

关于包冲突，maven默认采用 `最近优先` 的策略, 使用依赖树上优先找到的版本。 这种策略并不是maven的问题，用户要知道相关冲突并去解决。

> Currently, Maven supports the resolution of artifact versions by way of nearest-wins. That is, for any set of dependencies that share the same `groupId:artifactId:typeclassifier` in a particular artifact closure, the one declared nearest to the current project in the dependency tree will be selected for use.

3.1 Maven解决冲突方式

检查项目是否有冲突:

```
mvn dependency:tree -Dverbose | grep 'omitted for conflict'
```

针对特定项目查询冲突关系:

```
mvn dependency:tree -Dverbose -Dincludes=plexus-container-default
```

检查是否存在旧版本覆盖新版本的情况，如果存在则需要采用类似exclude旧版本或直接添加新版本依赖的方式解决。

参考

- maven-assembly-plugin
- maven-jar-plugin Set Up The Classpath
- Resolving conflicts using the dependency tree

作者

wongoo 2019-06-25