<https://blog.csdn.net/paullinjie/article/details/53188943>

**摘要**

这个技巧阐明了如何不直接处理清单文件而将一个不能运行jar包转换成一个可以执行的jar包。学会如何写一段转换jar包的程序,将你的jar包转换成你能使用java －jar 命令运行jar包或象在windows系统上那样通过双击来运行jar包。

[@more@]

你可以很容易地将一个应用的所有的类和资源打包到一个jar文件中去。事实上，这只是打包的一个原因。另一个原因是让用户很容易地执行包中的应用。那么在java的世界里，为什么jar是第二类公民――仅仅用作打包――当他们能成为第一类公民的时候，能和本地执行程序一样吗？

如果想执行jar文件，可以使用java命令的－jar选项。例如，你有一个可以运行的myjar.jar文件。因为该文件是可以执行的，所以你可以用如下命令执行之：java –jar myjar.jar。此外，安装在windows JRE将会把jar文件和JVM关联起来，以便通过双击来运行jar应用。

现在的问题是：如何把jar做成可以运行的包？

**清单文件和主类条目**

在大多数jar文件中，在目录META-INF里有个叫MANIFEST.MF的文件。在该文件中，有一个叫Main－Class的特殊条目，它说明了java －jar命令执行的类。

问题是你必须正确地在清单文件中加入这个特殊的条目――它必须是在特定的位置和特定的格式。可是，好多人不喜欢修改配置文件。

**用API修改改配置**

从java1.2之后，一个叫java.util.jar的包可以操作jar文件（注：它架构在java.util.zip包之上）。Jar包能让你通过Manifest类很容易地操作专门的清单文件。

让我们使用这些API写一个程序。首先，这个程序必须知道三件事：

1.      我们想执行的main类（这个文件必须存在于jar包中）；

3.<span style="7pt " times="" new="" roman"""="">      我们要输出的新jar包的名称，因为不是简单地覆盖原文件；

**编写代码**

上面的列表将组成我们程序的参数。在这里，让我们选择一个合适名字，MakeRunnable咋样？

**检查送入main的参数**

假设我们的main入口是标准的main（String[]）方法。我们首先应该检查程序的参数：

|  |
| --- |
| if (args.length != 3) {          System.out.println("Usage: MakeJarRunnable "                             + "  ");          System.exit(0);  } |

由于对后面的程序执行非常重要，一定要注意这个参数列表是如何解释的。参数的顺序和内容不是一成不变的；如果你改变了它们，记得要修改其它代码。

**存取jar和它的清单文件**

首先，我们必须创建一些知道jar和清单文件的对象：

|  |
| --- |
| //创建JarInputStream对象,获取它的清单  JarInputStream jarIn = new JarInputStream(new FileInputStream(args[0]));  Manifest manifest = jarIn.getManifest();  if (manifest == null) {  //如果清单不存在  manifest = new Manifest();  } |

**设置Main类的属性**

我们把Main-Class条目放到清单文件里main属性部分。一旦我们从mainfest对象中得到这个属性集，我们就能设置合适的main类。然而，如果Main－Class属性存在于原始的JAR文件中怎么办？这个程序仅仅打印出一个警告信息并退出。或许，我们可以添加一个命令行参数告诉程序使用新值而不是使用以前的一个：

|  |
| --- |
| Attributes a = manifest.getMainAttributes();      String oldMainClass = a.putValue("Main-Class", args[1]);      //如果旧值存取，显示提示信息并退出      if (oldMainClass != null) {          System.out.println("Warning: old Main-Class value is: "                             + oldMainClass);          System.exit(1);  } |

**输出新的JAR包**

我们需要创建一个新的jar文件，因此我们必须使用JarOutputStream类。注意：我们必须确保输出文件和输入文件不相同。作为可选方案，应该考虑如果两个文件同名，程序应该提示用户是否覆盖原始文件。我将这个作为练习留给读者。下面是代码。

System.out.println("Writing to " + args[2] + "...");

JarOutputStream jarOut = new JarOutputStream(new FileOutputStream(args[2]),  manifest);

我们必须将输入jar中每一个条目写到输出jar文件中去，因此，在所有的条目上进行枚举：

|  |
| --- |
| //为了从输入中转移数据而创建读缓存  byte[] buf = new byte[4096];      //枚举所有条目      JarEntry entry;      while ((entry = jarIn.getNextJarEntry()) != null) {          //排除旧jar文件中的清单文件          if ("META-INF/MANIFEST.MF".equals(entry.getName())) continue;          //把条目写到输出jar文件中去          jarOut.putNextEntry(entry);          int read;          while ((read = jarIn.read(buf)) != -1) {              jarOut.write(buf, 0, read);          }          jarOut.closeEntry();      }      //刷新和关闭所有的流  jarOut.flush();  jarOut.close();  jarIn.close(); |

**完成程序**

当然，我们必须将这些代码放到一个类中的main方法中，以及引入一些需要的包。资源一节提供了完整的程序。

**用法例子**

让我们用一个例子说明如何使用这个程序。假设你有一个main入口在叫做HelloRunnableWorld（类全名）类之中的应用，以及你已经创建了一个叫做myjar.jar的jar包，它包含了整个应用。在这个包上运行MakeJarRunnable，如下所示：

|  |
| --- |
| java MakeJarRunnable myjar.jar HelloRunnableWorld myjar\_r.jar |

此外，如前所述，注意我是如何安排参数列表顺序的。如果你忘记了顺序，运行一下这个不带参数的程序，它将会显示一个用法信息。

试着用java －jar命令运行myjar.jar文件，之后在myjar\_r.ja文件上。注意他们的不同！完成之后，研究在每一个jar包中的清单文件(META-INF/MANIFEST.MF)。

建议：试着将MakeJarRunnable制作成一个可以运行的Jar文件！

**用它处理你想运行的jar包**

通过双击或者使用简单的命令运行一个jar包，总是要比把它放到<span lang="EN