Oct 13, 2017 - 5 minute read - FAQ Maven Java For Testers

Simple ways to add and work with a `.jar` file in your local maven setup

TL;DR Hack - add as a library in IntelliJ project. Tactic - add as system scope in maven. Tactic/Strategic - install locally to .m2. Strategic - use a repository management tool, publish to maven central

Sometimes you want to work with a `jar` file that isn't hosted in maven central.

It might be a 3rd party jar, it might be one that you have written.

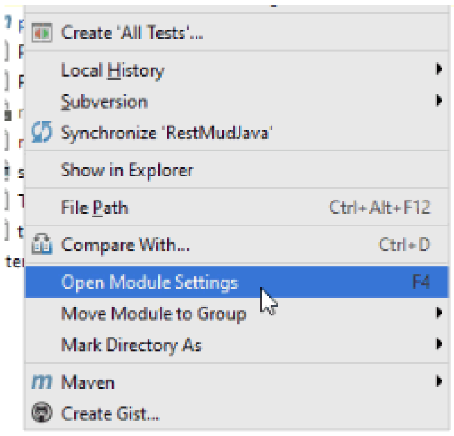
Regardless.

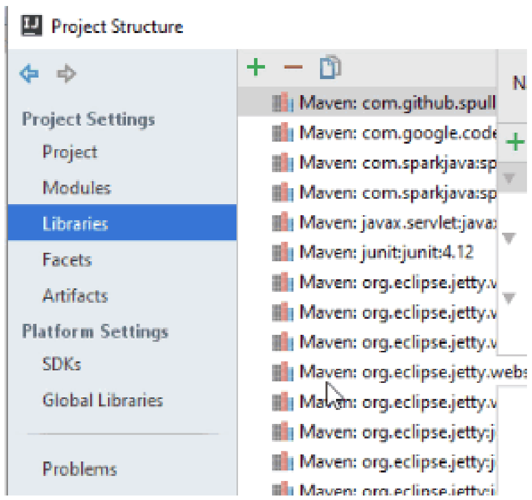
You have a lot of options for this. The approaches that I have used:

- add `.jar` files as an IntelliJ project dependency
- install it locally to your `.m2` repository
- add it to your project as a system scoped file
- use a repository management tool: like Nexus or Archiva
- publish the dependency to maven central

Quick Hack - add jar as an IntelliJ project dependency

For very quick hacks, add the `.jar` as an IntelliJ project dependency and bypass maven.





I demonstrate this in my free Java Desktop Application Technical Testing and you can read a blog post containing this information.

This is a very tactical approach:

- it doesn’t scale
- it doesn’t help you work with other people
- it isn’t very good for CI or version control

But it might help you get your immediate work done:

- experiment
- try something out
- get a task completed

And then adopt one of the following approaches if it works.

Add to project as system scoped file

As a short term tactic, I have also added the `.jar` as a system scoped file.

I did this in the past when working with a 'bug fix' version of Selenium WebDriver that had not yet propagated through to maven central, but which was available for download.

```
<dependency>
  <groupId>selenium_2_53_1</groupId>
  <artifactId>com.seleniumhq.selenium_2_53_1 </artifactId>
  <version>2.53.1</version>
  <scope>system</scope>
  <systemPath>
    C:/Users/Alan/Downloads/selenium-2.53.1/selenium-server-standalone-2.53.1.jar
  </systemPath>
</dependency>
```

This can be a useful tactic, but I don't think it really scales strategically.

e.g. http://seleniumsimplified.com/2016/06/use_selenium_webdriver_jar_locally/

Install .jar locally to your `.m2` repository

To have the `.jar` available as a dependency that I can bring in using a normal maven include, I can install the `.jar` locally to my `.m2` directory and repository.

This is explained on the maven website:

- https://maven.apache.org/guides/mini/guide-3rd-party-jars-local.html

And I use this approach at the moment when working with my RestMud game engine

I have split my game into two projects:

- game engine (which is now open source on github)

    - e.g. https://github.com/eviltester/restmud-game-engine
- RestMud game, Web Server and REST API

I have not released the game engine to Maven central, but the code is available on github, as is a release `.jar` file.

I can build a snapshot `.jar` locally for my current work.

Install it into my `.m2` folder

```
mvn install:install-file \
-Dfile=target/restmud-engine-1.4-SNAPSHOT-jar-with-dependencies.jar \
-DpomFile=pom.xml
```

If I didn't have the `pom.xml` file, I could still do this, I just add the details from the `pom.xml` into my command line:

```
mvn install:install-file \
-Dfile=target/restmud-engine-1.4-SNAPSHOT-jar-with-dependencies.jar \
-DgroupId=uk.co.compendiumdev \
-DartifactId=restmud-engine \
-Dversion=1.4-SNAPSHOT \
-Dpackaging=jar
```

And if I want the source code jar to be associated with the `.jar` (which I usually do) then I add the following argument to the command line:

```
-Dsources=target/restmud-engine-1.4-SNAPSHOT-sources.jar
```

This allows me to keep the `pom.xml`of my projects which use the `.jar` to remain as though the `.jar` was on a repository manager or in maven central.

I think this is a good tactical approach, that supports a longer term strategic development of your development and automated execution approach.

Use a repository management tool

You could install a dependency management tool like nexus or Archiva.

Then your `.jar` files are installed into this repository which is accessible by your team, and CI process, and not just your local development machine.

Maven docs on Repositories:

- https://maven.apache.org/repository-management.html
    - Archiva https://archiva.apache.org/index.cgi
    - Nexus https://www.sonatype.com/download-oss-sonatype

You have to:

- install the dependency management tool
- configure the `pom.xml` to have a `<repositories>` section and point to your dependency management tool

You'll probably want to evaluate which of the repository management tools works best for you and your environment.

This is a much more strategic approach and is good for team work and continuous integration processes.

Publish to Maven Central

This is probably the most strategic long term approach, but requires you to make your work public.

All of the other approaches mentioned allow you to keep your work to yourself.

It is quite a long process, so I won't describe it here, but I have a full write up on my blog.

http://blog.javafortesters.com/2016/10/how-to-create-and-release-jar-to-maven.html

Summary

- The ultimate short term hack - add it as a dependency in your IDE
- For quick hacks - add it as `system` scoped maven dependency

- For personal work, or moving towards a strategic approach, install locally to `.m2`
- For longer term tactical work as a team, use a repository manager
- For strategic open source work, release to maven central

How to work with a `.jar` file in your local maven setup : Add, Install, ...
Watch later
Share
Apache Maven Project
Maven
Add .jar files to Maven

Watch on YouTube

Useful Links

Hack Tactic: Adding `.jar` to IntelliJ

- https://www.compendiumdev.co.uk/page.php?title=casestudyjavadesktop
- http://blog.eviltester.com/2016/02/lessons-learned-testing-house-of-test.html

Install `.jar` into your local `.m2` folder or as System scope

- https://maven.apache.org/guides/mini/guide-3rd-party-jars-local.html

- https://stackoverflow.com/questions/4955635/how-to-add-local-jar-files-to-a-maven-project
- http://seleniumsimplified.com/2016/06/use_selenium_webdriver_jar_locally/
- http://roufid.com/3-ways-to-add-local-jar-to-maven-project

Example code that has the mvn install instructions in it is my restmud-game-engine

Using a Repository Manager

- https://maven.apache.org/repository-management.html

Releasing your Jar to maven central

- http://blog.javafortesters.com/2016/10/how-to-create-and-release-jar-to-maven.html

---

Comments

On DZone, Florien Enner mentioned the approach he likes to use:

> My personal favorite is to install the jar into a local folder (e.g. a 'lib' folder next to 'src') that can be added as an external repo in Maven. That way you can just check it into version control and don't need to setup a full repo or require every computer to install a jar into the m2 repo. You can do that by specifying e.g. -DlocalRepositoryPath="./" See e.g. https://stackoverflow.com/q/3642023

Share this post on: Twitter , Facebook , LinkedIn

Posted by Alan Richardson Oct 13, 2017 - in FAQ Maven Java For Testers

---

<< How to Diff Java Code in IntelliJ - 3 ways to use the Compare Tool

What is Regression Testing? Software Testing FAQs and Definitions >>

You will need a Github account to comment. Or you can contact me with your comment.

I reserve the right to delete spam comments e.g. if your comment adds no value and its purpose is simply to create a backlink to another site offering training, or courses, or etc.

2 Comments - *powered by utteranc.es*

LittleLittleQ commented on Mar 13, 2019

Hi, I'd like to know after installing jar locally to `.m2` repository initially, do we need to reinstall it while the jar is rebuilt in the later?

eviltester commented on Mar 13, 2019 Owner

Yes. If the jar is rebuilt and it is being pulled into a project from the maven repo then it would be need to be reinstalled back into the the local .m2 repository.

Write Preview

Sign in to comment

Styling with Markdown is supported

Sign in to comment

Hire Me - I'm an Expert

- Coaching,
- Consultancy,
- Mentoring,
- Test Automation,
- Health Checks,
- Code Reviews,
- Agile Testing

Contact

Contact Me

Join Email List

Email Address

Subscribe

Email Privacy Details.

Online Training Courses

Selenium WebDriver With Java (15+ hours) for only $229

Technical Web Testing Course ($10)

Evil Tester Talks Bundle ($10)

See 4 More Courses

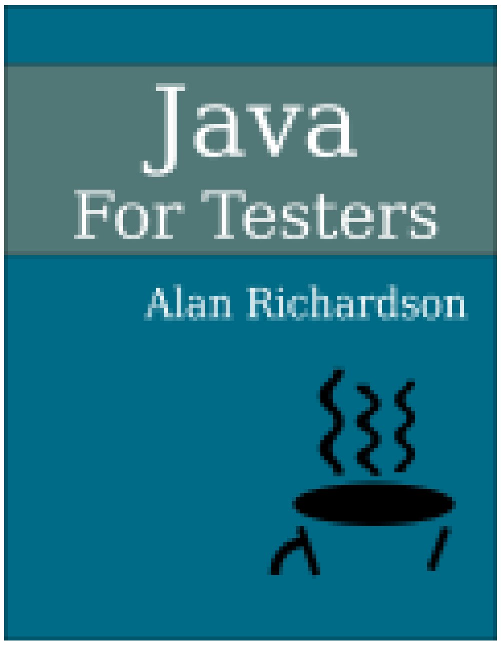
Join Our Mentoring Patreon Community

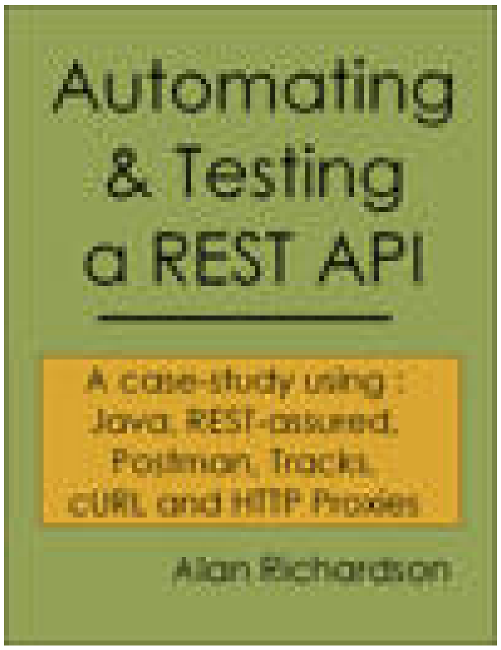
Gain access to hints, tips, and prompts for improving your testing skills. Regular updates, multiple times a week for as little as $1 per month.

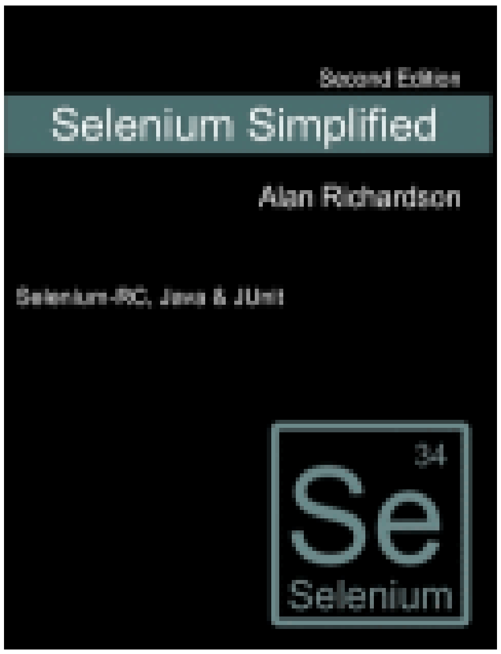
Learn more about Patreon Mentoring

Books

Dear
Evil
Tester
PROVOCATIVE ADVICE THAT COULD CHANGE YOUR APPROACH TO TESTING FOREVER
Alan Richardson




Automating & Testing a REST API
A case-study using : Java, REST-assured, Postman, Tracks, cURL and HTTP Proxies
Alan Richardson



Learn more about our books

Free Agile Testing FAQs eBook

How does Software Testing fit into Agile Projects? Download our free brochure and find out.

Follow

Podcasts and Videos

Podcasts

Subscribe to my YouTube Channel

Web Testing Resource List

Need some motivation?

Try the Sloganizer

"Of course I'm not good, I am GOOD! Don't you?"

Recent Blog Posts

I need a tool... no, you need to work on the system

An Introduction to API Based Documentation Automating

June 2019 EvilTester.com and Patreon Content Summary

How to write a link checker in the browser with Vanilla JavaScript

Overview of AccelQ Web Automating

May 2019 EvilTester.com and Patreon Content Summary

April 2019 EvilTester.com and Patreon Content Summary

What to do if your Chrome Extension is Rejected from Chrome Store

- All Blog Categories
- Blog Post Archive List

Also...

Recommended Reading List