Buildtools 2: Language specific buildtools

Joseph Hallett

January 13, 2023

Last time...

We talked about *Make* as a tool for compiling code

- We discussed how *Make* could be used to shift documents between different formats

But there is one thing *Make*'s can't do...

Versioning

Modern development makes extensive use of external libraries…
But *Make* is rubbish at dealing with them:

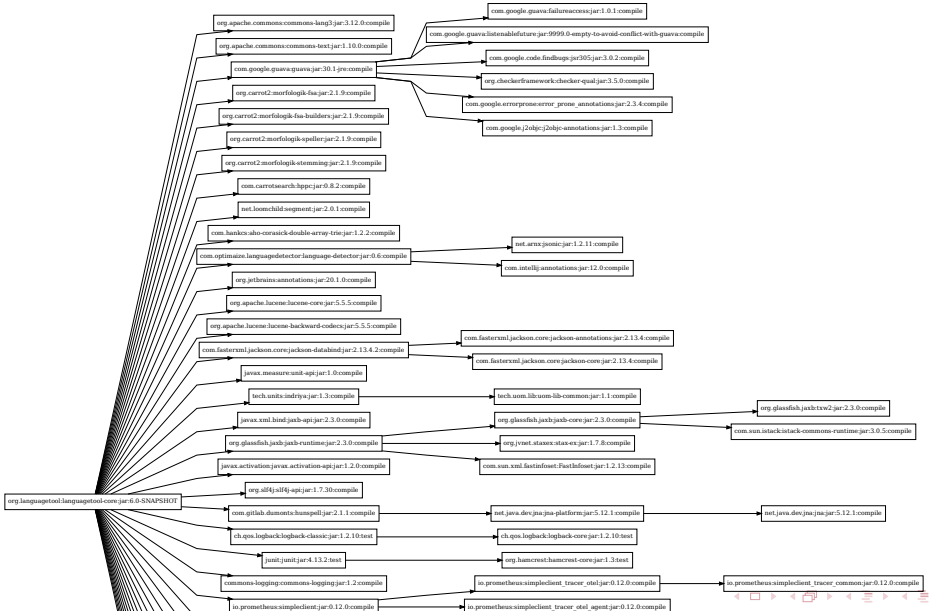
- Doesn't know how to fetch dependencies
- Doesn't track versions beyond *source is newer than object*

LanguageTool is a cool little Java grammar checker:

- How many libraries does *just the core* of the tool make use of?

```
mvn dependency:tree -D outputType=dot | dot -Tpdf
```

This is surely too many?



In the old days...

Traditionally you'd have to go download all the dependencies by hand...

- And then compile and install them
- Very tedious and error prone

So we automated it!

Modern build tooling

(Almost) every language comes with its own library management tooling

- Lets developers specify dependencies
- Tells compiler how to rebuild the project

...which means *for every language you use you need to learn its build tools*...

- Yay?

(Honestly, I still use *Make* but I'm old and cantankerous)

So now we have...

Commonlisp	ASDF and Quicklisp
Go	Gobuild
Haskell	Cabal
Java	Ant, Maven, Gradle...
JavaScript	NPM
Perl	CPAN
Python	Distutils and `requirements.txt`
R	CRAN
Ruby	Gem
Rust	Cargo
LaTeX	CTAN and TeXlive

...and *many* more.

And they're all different

Very little similarity between *any* of them.

- You need to learn the ones you use.
- We'll play in the labs with *Maven* for Java a little bit

Maven Quickstart

```
mkdir /tmp/src
cd /tmp/src
mvn archetype:generate \
    -DgroupId=uk.ac.bristol.cs \
    -DartifactId=hello \
    -DarchetypeArtifactId=maven-archetype-quickstart \
    -DinteractiveMode=false
```

```
INFO Scanning for projects\ldots{}
INFO
INFO ------------------< org.apache.maven:standalone-pom >-------------------
INFO Building Maven Stub Project (No POM) 1
INFO --------------------------------[ pom ]---------------------------------
INFO
INFO >>> maven-archetype-plugin:3.2.1:generate (default-cli) > generate-sources @ standalone-
INFO
INFO <<< maven-archetype-plugin:3.2.1:generate (default-cli) < generate-sources @ standalone-
INFO
INFO
INFO --- maven-archetype-plugin:3.2.1:generate (default-cli) @ standalone-pom ---
INFO Generating project in Batch mode
INFO ----------------------------------------------------------------------------
```

...and after spewing all that...

```
find /tmp/src -type f
```

- (”/tmp/src/hello/pom.xml”)
- (”/tmp/src/hello/src/main/java/uk/ac/bristol/cs/App.java”)
- (”/tmp/src/hello/src/test/java/uk/ac/bristol/cs/AppTest.java”)

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>uk.ac.bristol.cs</groupId>
  <artifactId>hello</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>hello</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

And if we try and build...

```
mvn package
[INFO] Scanning for projects\ldots{}
[INFO]
[INFO] -----------------------< uk.ac.bristol.cs:hello >-----------------------
[INFO] Building hello 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ hello ---
[WARNING] Using platform encoding (US-ASCII actually) to copy filtered resources, i.e. build is
[INFO] skip non existing resourceDirectory /tmp/src/hello/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ hello ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding US-ASCII, i.e. build is platf
[INFO] Compiling 1 source file to /tmp/src/hello/target/classes
[INFO] -------------------------------------------------------------
[ERROR] COMPILATION ERROR :
[INFO] -------------------------------------------------------------
[ERROR] Source option 5 is no longer supported. Use 7 or later.
[ERROR] Target option 5 is no longer supported. Use 7 or later.
[INFO] 2 errors
```

Lets fix that for it shall we?

(It's either that or installing an ancient Java compiler...)

```
ed /tmp/src/hello/pom.xml <<EOF
10i
  <properties>
    <maven.compiler.source>17</maven.compiler.source>
    <maven.compiler.target>17</maven.compiler.target>
  </properties>
.
wq
EOF
```

639 778

And if we try and build, again...

```
mvn package
[INFO] Scanning for projects\ldots{}
[INFO]
[INFO] ------------------------< uk.ac.bristol.cs:hello >------------------------
[INFO] Building hello 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ hello ---
[WARNING] Using platform encoding (US-ASCII actually) to copy filtered resources, i.e. build is
[INFO] skip non existing resourceDirectory /tmp/src/hello/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ hello ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding US-ASCII, i.e. build is platf
[INFO] Compiling 1 source file to /tmp/src/hello/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ hello ---
[WARNING] Using platform encoding (US-ASCII actually) to copy filtered resources, i.e. build is
[INFO] skip non existing resourceDirectory /tmp/src/hello/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ hello ---
```

Does anyone actually know why Java stuff is so ridiculously verbose?

```
find /tmp/src -type f
```

- (”/tmp/src/hello/pom.xml”)
- (”/tmp/src/hello/src/main/java/uk/ac/bristol/cs/App.java”)
- (”/tmp/src/hello/src/test/java/uk/ac/bristol/cs/AppTest.java”)
- (”/tmp/src/hello/target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst”)
- (”/tmp/src/hello/target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst”)
- (”/tmp/src/hello/target/maven-status/maven-compiler-plugin/testCompile/default-testCompile/createdFiles.lst”)
- (”/tmp/src/hello/target/maven-status/maven-compiler-plugin/testCompile/default-testCompile/inputFiles.lst”)
- (”/tmp/src/hello/target/classes/uk/ac/bristol/cs/App.class”)
- (”/tmp/src/hello/target/test-classes/uk/ac/bristol/cs/AppTest.class”)
- (”/tmp/src/hello/target/surefire-reports/uk.ac.bristol.cs.AppTest.txt”)
- (”/tmp/src/hello/target/surefire-reports/TEST-uk.ac.bristol.cs.AppTest.xml”)
- (”/tmp/src/hello/target/maven-archiver/pom.properties”)
- (”/tmp/src/hello/target/hello-1.0-SNAPSHOT.jar”)

Other useful commands

`mvn test` run the test suite

`mvn install` install the JAR into your local JAR packages

`mvn clean` delete everything

And if I'm being a bit snarky…

`https://gradle.org` A *better* Java build tool

(That doesn't work everywhere and is much worse than Maven when you try and do more complex things…)

Wrap up

- Language specific build tools exist
- You should probably use them
- (but I still use good ol' make a lot more)

Aside

Sometimes you'll find you pull a project and it uses a certain build system and you just know you're going to have to spend a day fighting it.
...please don't use CMake.