
Build Automation

- A Brief Introduction to Maven

CS480 Software Engineering

<http://cs480.yusun.io>

January 30, 2015

Yu Sun, Ph.D.

<http://yusun.io>

yusun@cpp.edu



CAL POLY POMONA

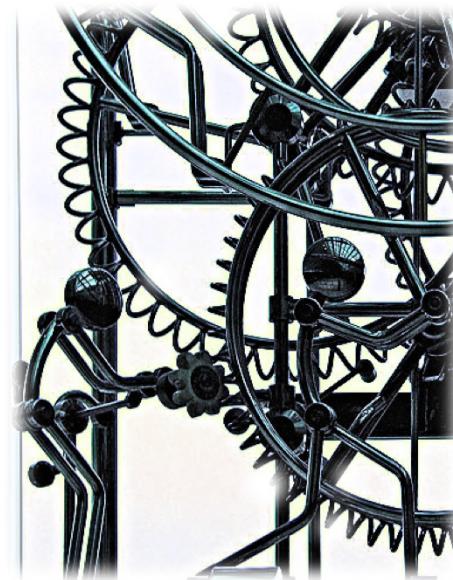
Very Important in Practice

- ◆ Gateway to participate in real software development
- ◆ You cannot start coding without knowing how to use the build automation tools in the industry



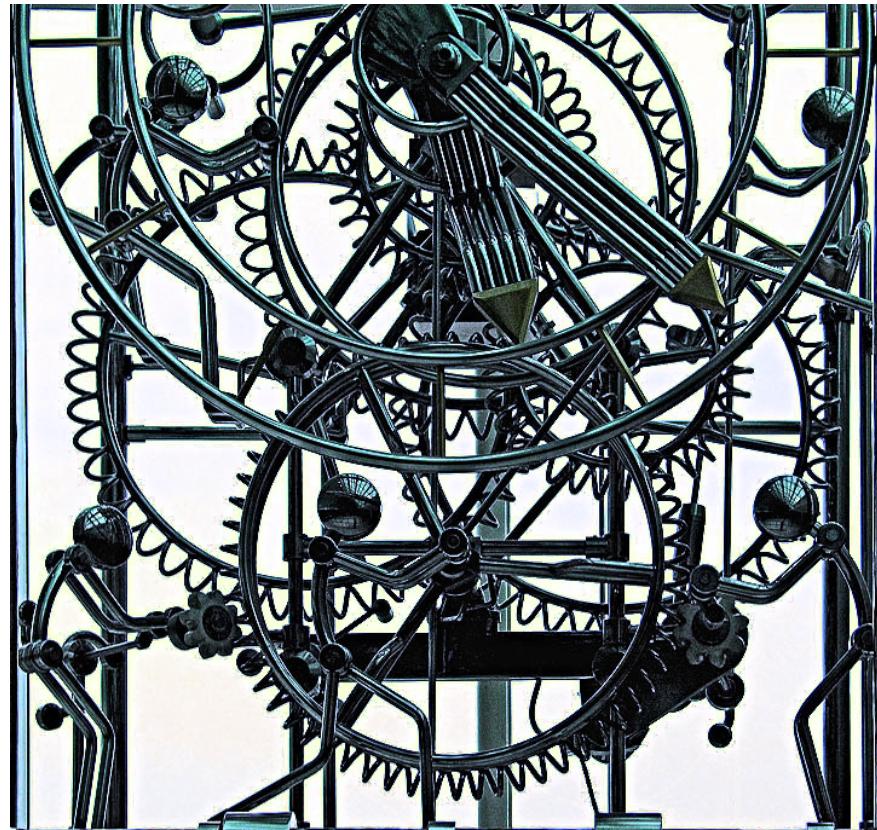
What is Build Automation (Maven)?

- ◆ Build lifecycle
- ◆ Dependency management tool
- ◆ Artefact repository
- ◆ Collection of plugins
- ◆ Project reporting tool
- ◆ Set of conventions
- ◆ Distilled experience



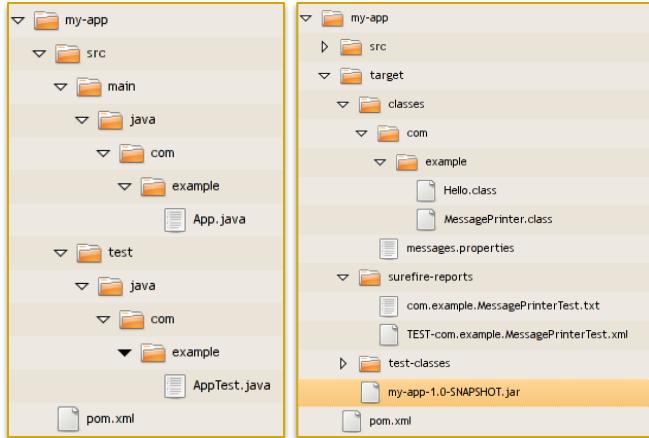
What Else is Build Automation(Maven)?

- ◆ Succinct command line tool
- ◆ Designed for Java/Java EE/other
- ◆ Holder/publisher of project documentation
- ◆ Generator of project metrics
- ◆ Customisable: environment, lifecycle, etc.
- ◆ Inheritable
- ◆ Declarative
- ◆ Encourager of modularity and reuse
- ◆ Integrated with SCM tools
- ◆ Integrated with IDEs
- ◆ Integrated with Ant
- ◆ System of repositories
- ◆ Project kick starter
- ◆ Release manager
- ◆ Deployer
- ◆ Enabler of portable build knowledge
- ◆ Encourager of best practice
- ◆ Community
- ◆ Not perfect



We Focus on the Important

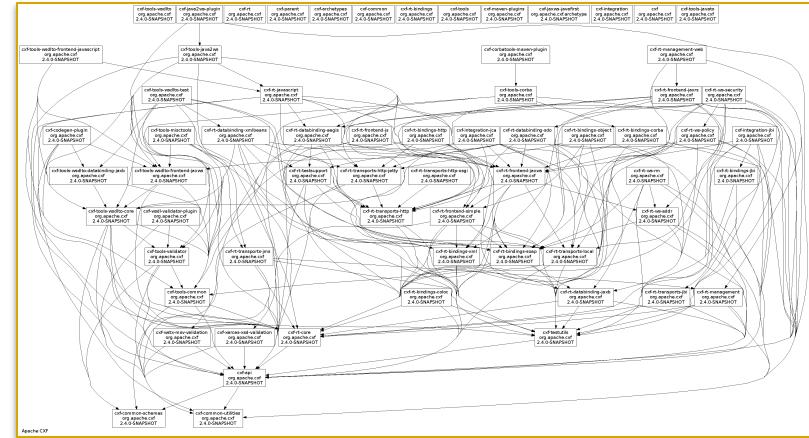
■ Project Structure Management



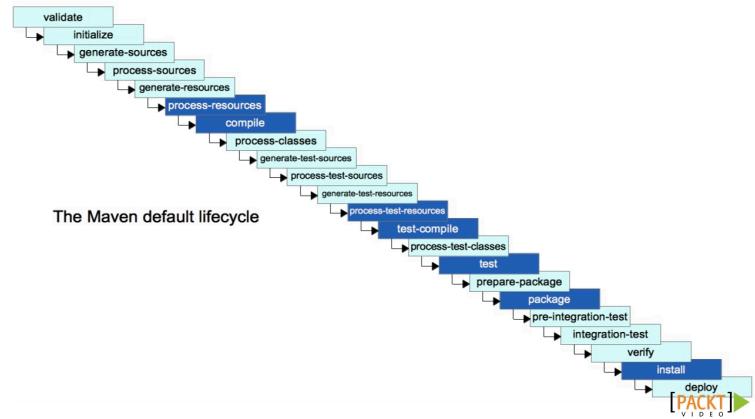
■ Software Artifact Packing



■ Dependency Management



■ Build Lifecycle Management



Why Project Structure Management

- ◆ When software grows, project files and artifacts have to be well organized



Why Project Structure Management

- ◆ The project structure should be standardized
- ◆ Anyone can quickly build others' projects and collaborate



Create Maven Project

- ◆ Use Eclipse Maven plugin
- ◆ `mvn archetype:generate`

Directory Structure Convention



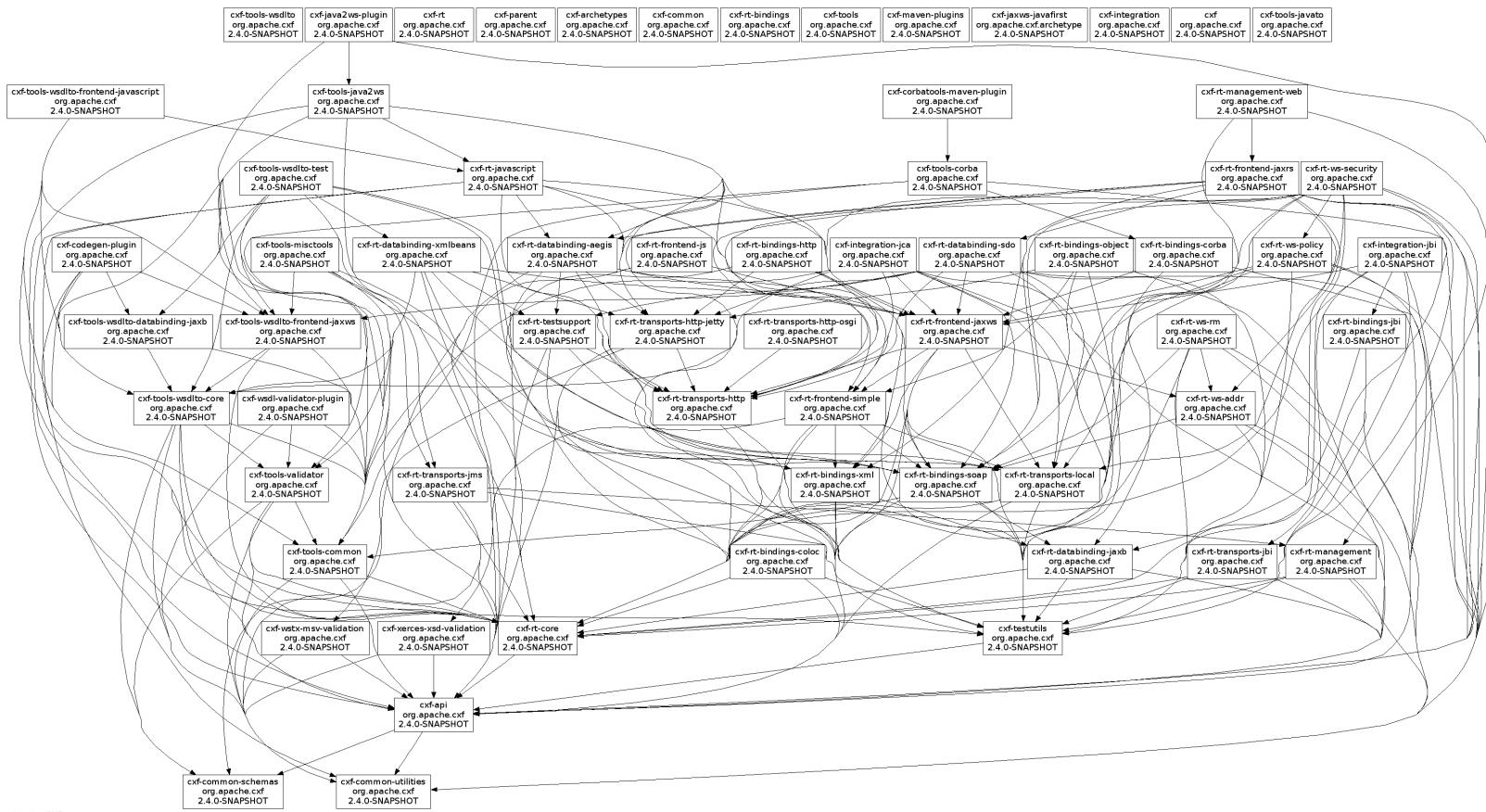
- ◆ Java sources:
src/main/java
- ◆ Unit tests:
src/test/java
- ◆ pom.xml

Maven POM

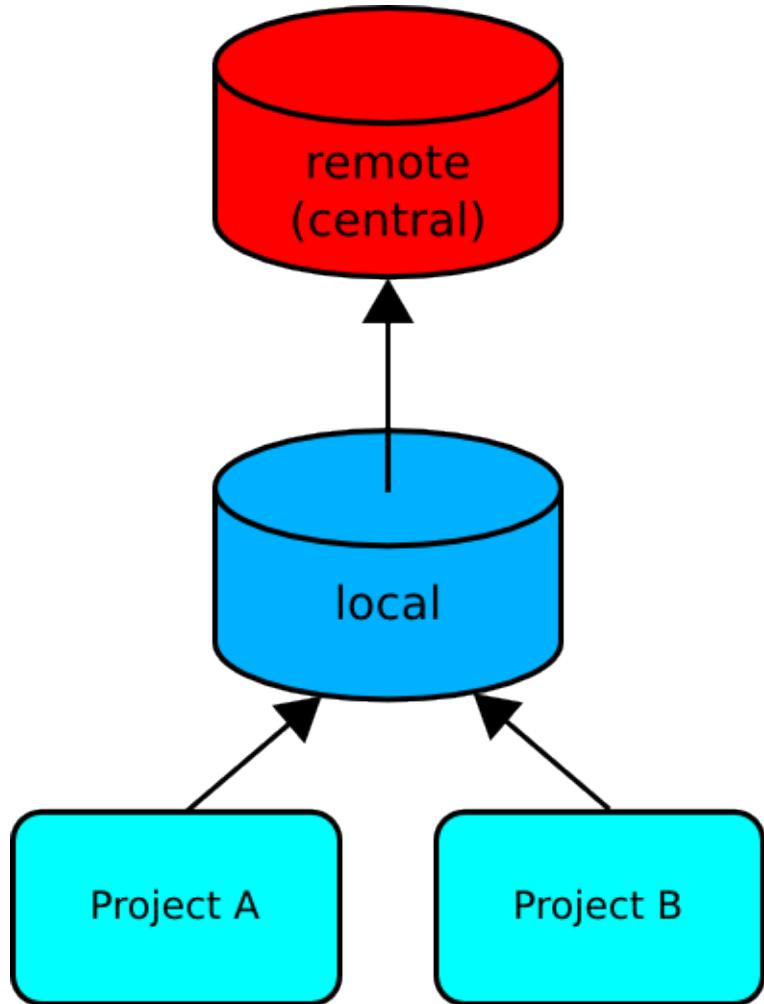
- ◆ Stands for Project Object Model
- ◆ Describes a project
 - ◆ Name and Version
 - ◆ Artifact Type
 - ◆ Source Code Locations
 - ◆ Dependencies
 - ◆ Plugins
 - ◆ Profiles (Alternate build configurations)
- ◆ Uses XML by Default

Dependency Management

- ◆ Maven revolutionized Java dependency management
 - ◆ No more checking libraries into version control



Maven Repositories



- ◆ Repositories store artifacts:
 - ◆ plugins
 - ◆ project dependencies
- ◆ Central: <http://repo1.maven.org/maven2>
(or mirror)
- ◆ Local: `~/.m2/repository`
- ◆ The first execution of a plugin, or requirement for a dependency pulls the artifact from central and caches it locally

Adding a Dependency

- ◆ Dependencies consist of:
 - ◆ GAV
 - ◆ Scope: compile, test, provided (default=compile)
 - ◆ Type: jar, pom, war, ear, zip (default=jar)

```
<project>
  ...
  <dependencies>
    <dependency>
      <groupId>javax.servlet</groupId>
      <artifactId>servlet-api</artifactId>
      <version>2.5</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
</project>
```

Software Artifact Packing

- ◆ Make your software deliverable

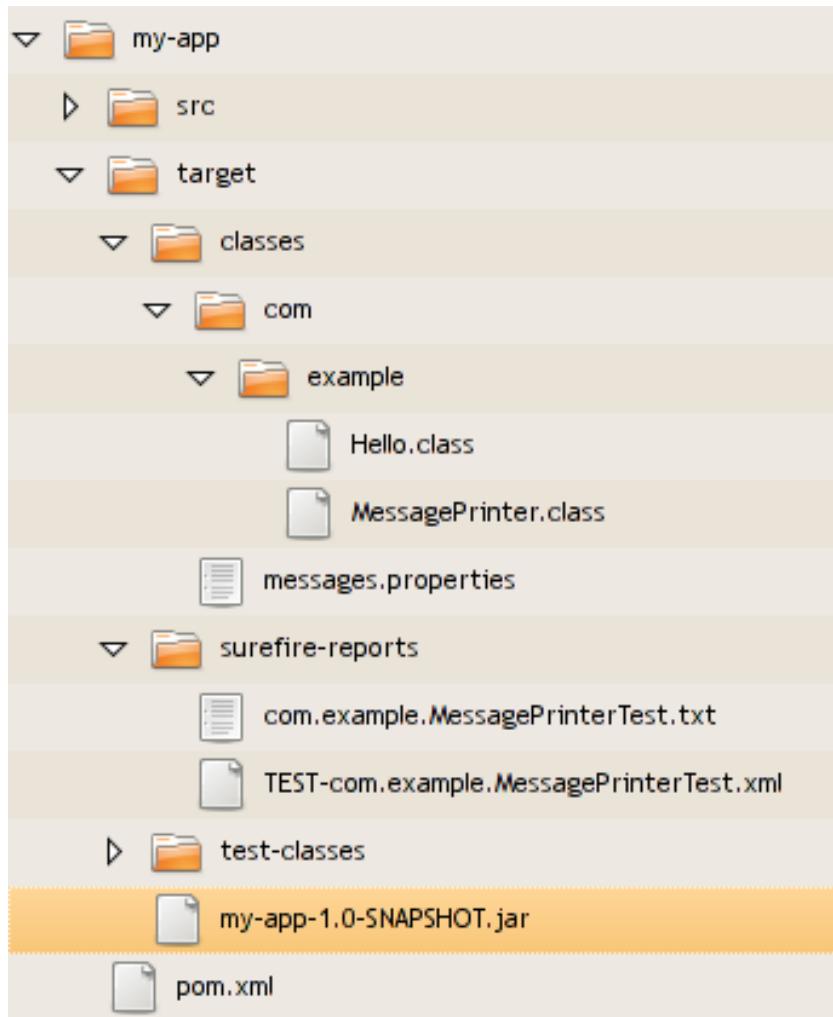


Ready for Take Off



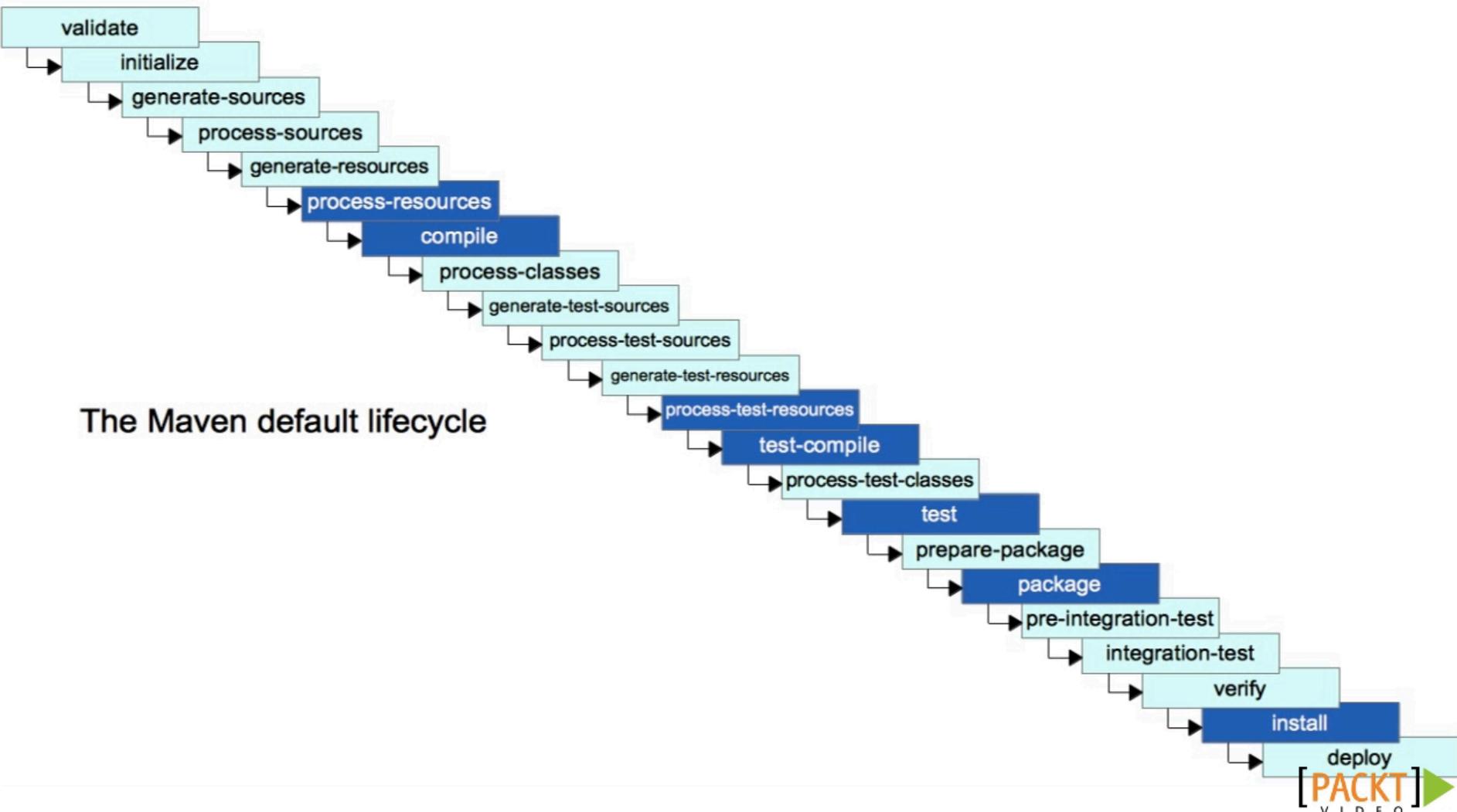
mvn package

The Finished Product



- ◆ Classes and test classes compiled
 - ◆ Resources copied to classes directory
 - ◆ Test reports created
 - ◆ Jar file created
- ```
$ java -jar my-app-1.0-SNAPSHOT.jar
 Hello World!
```

# Build Lifecycle Management



The Maven default lifecycle

# Maven Build Lifecycle

---

- ◆ A Maven build follow a lifecycle
- ◆ Default lifecycle
  - ◆ generate-sources/generate-resources
  - ◆ compile
  - ◆ test
  - ◆ package
  - ◆ integration-test (pre and post)
  - ◆ Install
  - ◆ deploy
- ◆ There is also a Clean lifecycle

# Example Maven Goals

---

- ◆ To invoke a Maven build you set a lifecycle “goal”
- ◆ mvn install
  - ◆ Invokes generate\* and compile, test, package, integration-test, install
- ◆ mvn clean
  - ◆ Invokes just clean
- ◆ mvn clean compile
  - ◆ Clean old builds and execute generate\*, compile
- ◆ mvn compile install
  - ◆ Invokes generate\*, compile, test, integration-test, package, install
- ◆ mvn test clean
  - ◆ Invokes generate\*, compile, test then cleans