Introduction to Maven

Produced by:

Dr. Siobhán Drohan (sdrohan@wit.ie)

Eamonn de Leastar (edeleastar@wit.ie)

Maven™

Definition and Objectives

What is Maven?

- a Yiddish word meaning

  accumulator of knowledge!

- a tool for building and managing any Java-based project (i.e. a software project management and comprehension tool).

http://maven.apache.org/what-is-maven.html

Maven’s Objectives

Maven’s goal is to allow a developer to understand the complete state of a development effort in the shortest period of time. It encompasses:

1. Making the build process easy
2. Providing a uniform build system
3. Providing quality project information
4. Providing guidelines for best practices development
5. Allowing transparent migration to new features

http://maven.apache.org/what-is-maven.html

Objective 1:
Making the build process easy

> *“While using Maven doesn’t eliminate the need to know about the underlying mechanisms, Maven does provide a lot of shielding from the details.”*

http://maven.apache.org/what-is-maven.html

Objective 2:
Providing a uniform build system

- Maven builds a project using its project object model (pom.xml) and a set of plugins shared by all projects using Maven:

  → *providing a uniform build system.*

- If you know how one Maven project builds you automatically know how all Maven projects build:

  → *saves immense time when involved in many projects.*

http://maven.apache.org/what-is-maven.html

Objective 3:
Providing quality project information

Maven provides plenty of useful project information, some from pom.xml, some generated from your project’s sources e.g.:

- Build Settings e.g. versions of JDK/JUnit, plugins, etc.
- List of Dependencies
- Unit test reports including code coverage
- etc.

http://maven.apache.org/what-is-maven.html

Objective 4: Providing guidelines for best practices development

- Maven aims to gather current principles for best practices development, and make it easy to guide a project in that direction e.g.

  - specification, execution, and reporting of unit tests are part of the normal build cycle using Maven. Current unit testing best practices were used as guidelines:

    - Keeping your test source code in a separate, but parallel source tree
    - Using test case naming conventions to locate and execute tests
    - Have test cases setup their environment and don't rely on customizing the build for test preparation.

- Maven also lays out your project's directory structure → once you learn the layout you can easily navigate any other Maven project.

http://maven.apache.org/what-is-maven.html

Objective 5: Allowing transparent migration to new features

Maven installation is easy to update so you can take advantage of any changes made to Maven itself.

http://maven.apache.org/what-is-maven.html

Some More Objectives…

- Provides a standard development infrastructure across projects.
- Make the development process transparent.
- Decrease training for new developers.
- Bring together tools in a uniform way.
- Prevent inconsistent setups.
- Divert energy to application development activities.
- Project setups are simple and reusable; new projects can be set up in a very short time.

Maven™

Project Object Model (pom.xml)

Project Object Model (pom.xml)

- XML representation of a Maven project; a one-stop-shop for all things concerning the project.

- It is, effectively:
  - the declarative manifestation of the "who", "what", and "where",
  - while the build lifecycle is the "when" and "how".

- pom.xml is found in the base directory of project.

https://maven.apache.org/pom.html#What_is_the_POM

Project Object Model (pom.xml)

- It tells Maven how to execute a project.
- Contains metadata about the project
  - Location of directories,
    Developers/Contributors
    Extra plugins required
    Special plugin configuration
    Jars required (3rd party and in-house)
    Repositories to search for plugins/jars, etc.
- A project's POM inherits from the Super POM.
  - All standard project information (e.g. directory structure) is held in the Super POM (principle).

Skeleton/Minimal pom.xml

```
<project>

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.mycompany.app</groupId>
  <artifactId>my-app</artifactId>
  <packaging>jar</packaging>
  <version>1.0</version>

  <dependencies>

   <dependency>
     <groupId>com.thoughtworks.xstream</groupId>
     <artifactId>xstream</artifactId>
     <version>1.4.10</version>
   </dependency>

  </dependencies>

</project>
```

Uniquely identify project in the repo

Coherent organization of dependencies

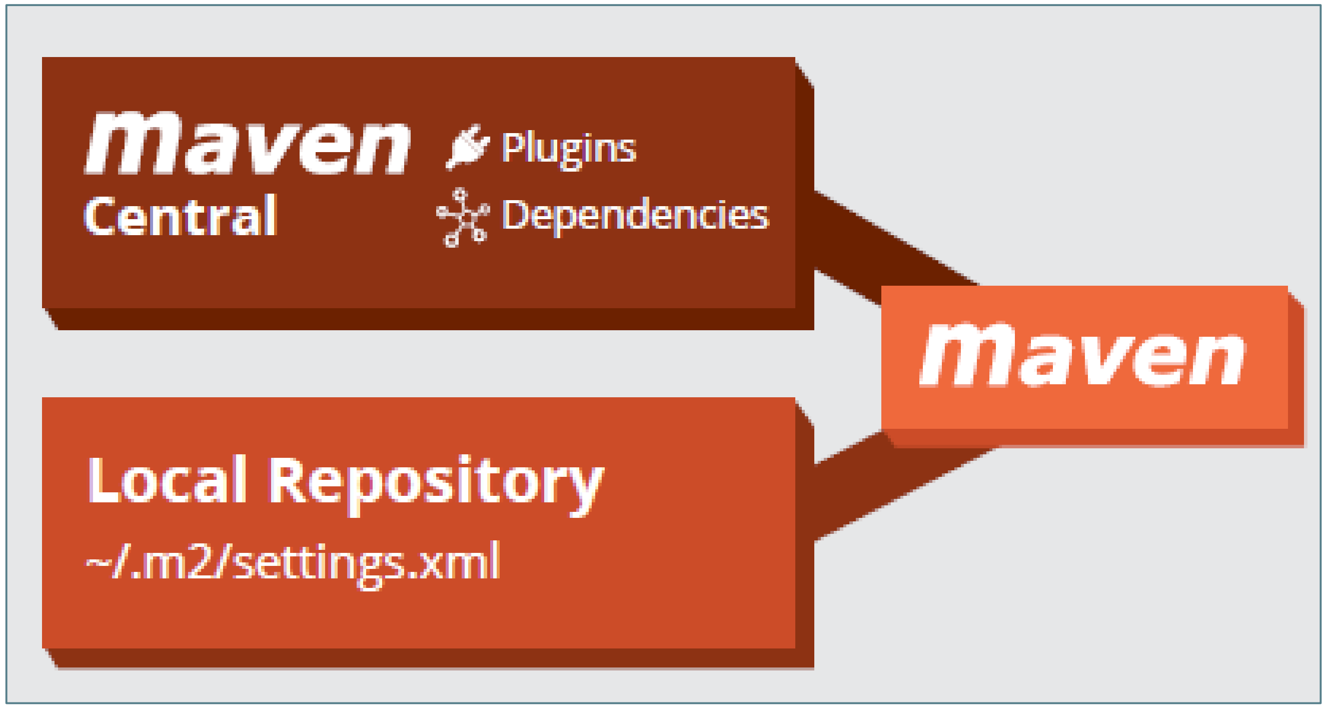
- Three related concepts: Artifact; Dependencies; Repositories

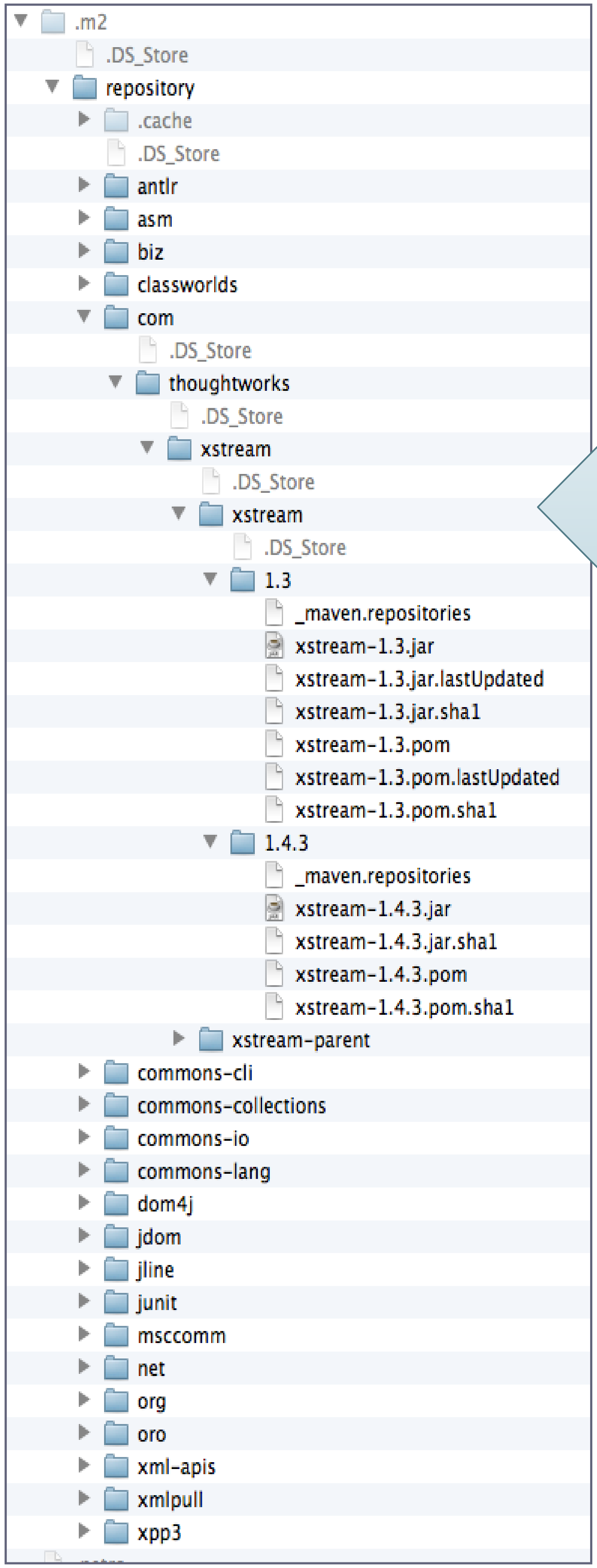
```
<project>
       .........
  <dependencies>

    <dependency>
      <groupId>com.thoughtworks.xstream</groupId>
      <artifactId>xstream</artifactId>
      <version>1.4.10</version>
    </dependency>

   </dependencies>

</project>
```

This project has a dependency on version **1.4.10** of the artifact with id **xstream**, produced by the **com.thoughtworks.xstream** group.

Coherent organization of dependencies

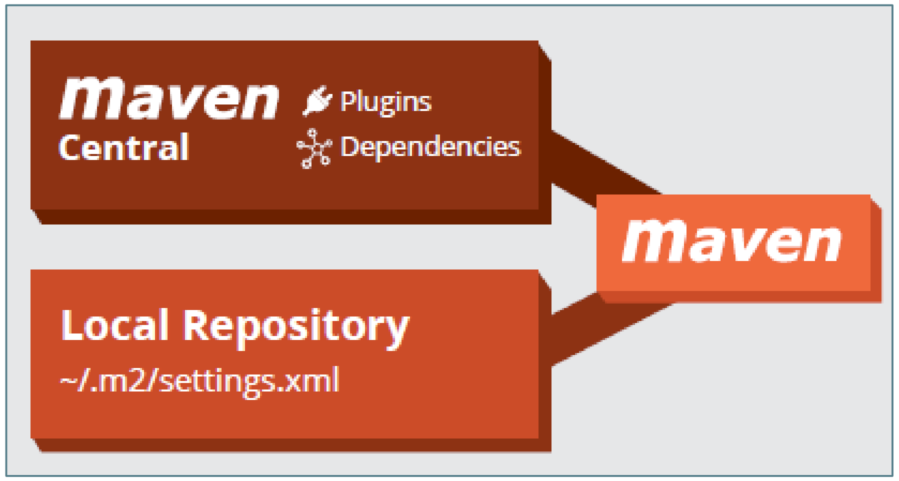
- All artifacts/dependencies are stored in repositories
  - Local and remote repositories
- The local repository is searched first, then remote ones.

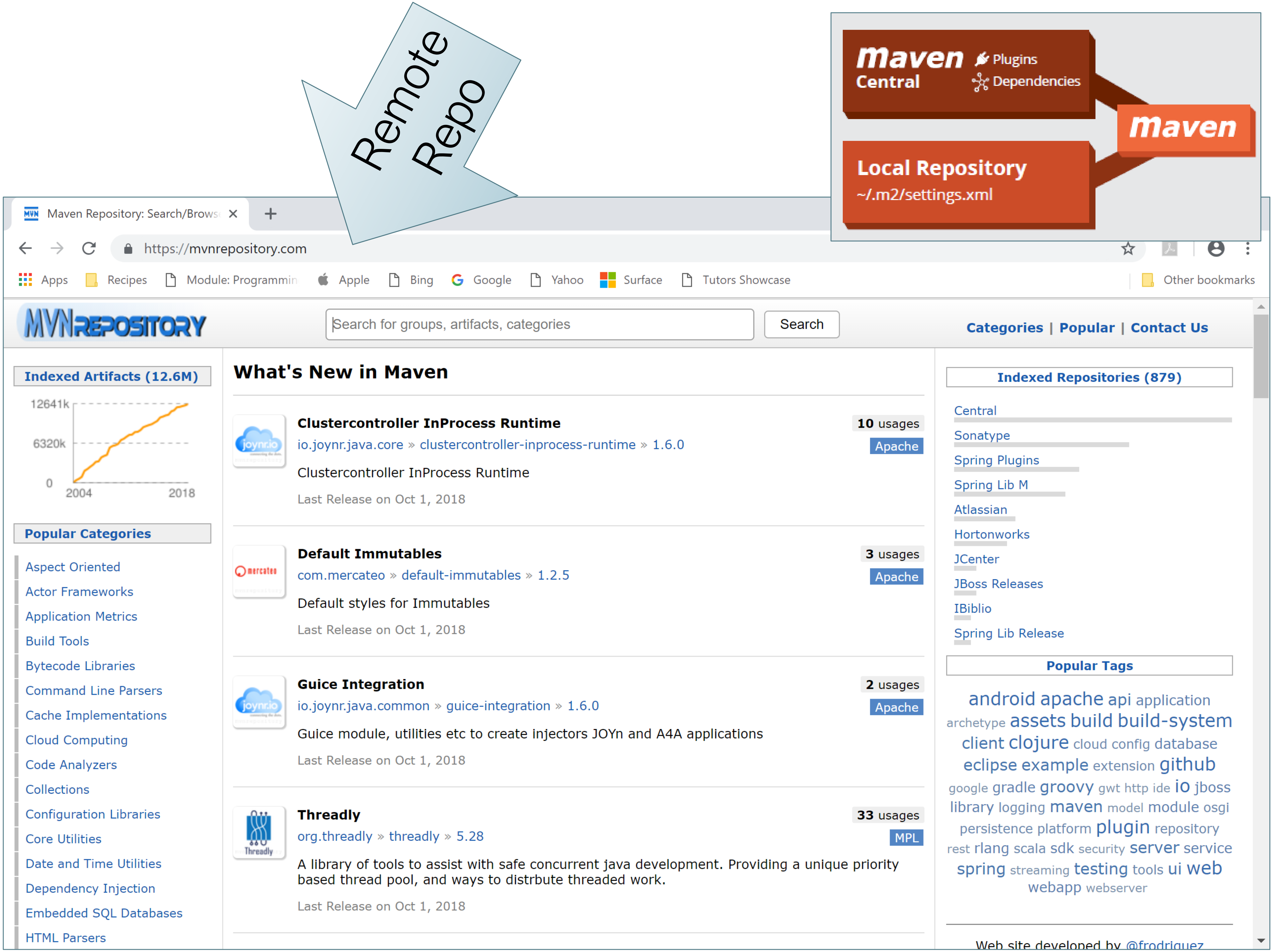




```
▼ .m2
    .DS_Store
  ▼ repository
    ▶ .cache
      .DS_Store
    ▶ antlr
    ▶ asm
    ▶ biz
    ▶ classworlds
    ▼ com
        .DS_Store
      ▼ thoughtworks
          .DS_Store
        ▼ xstream
            .DS_Store
          ▼ xstream
              .DS_Store
            ▼ 1.3
                _maven.repositories
                xstream-1.3.jar
                xstream-1.3.jar.lastUpdated
                xstream-1.3.jar.sha1
                xstream-1.3.pom
                xstream-1.3.pom.lastUpdated
                xstream-1.3.pom.sha1
            ▼ 1.4.3
                _maven.repositories
                xstream-1.4.3.jar
                xstream-1.4.3.jar.sha1
                xstream-1.4.3.pom
                xstream-1.4.3.pom.sha1
          ▶ xstream-parent
    ▶ commons-cli
    ▶ commons-collections
    ▶ commons-io
    ▶ commons-lang
    ▶ dom4j
    ▶ jdom
    ▶ jline
    ▶ junit
    ▶ msccomm
    ▶ net
    ▶ org
    ▶ oro
    ▶ xml-apis
    ▶ xmlpull
    ▶ xpp3
```

Local
Repo





Remote Repo

Maven Central

Plugins

Dependencies

Local Repository

~/.m2/settings.xml

Maven

MVNREPOSITORY

Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

Indexed Artifacts (12.6M)

Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers
- Collections
- Configuration Libraries
- Core Utilities
- Date and Time Utilities
- Dependency Injection
- Embedded SQL Databases
- HTML Parsers

What's New in Maven

Clustercontroller InProcess Runtime — **10** usages

io.joynr.java.core » clustercontroller-inprocess-runtime » 1.6.0 — Apache

Clustercontroller InProcess Runtime

Last Release on Oct 1, 2018

Default Immutables — **3** usages

com.mercateo » default-immutables » 1.2.5 — Apache

Default styles for Immutables

Last Release on Oct 1, 2018

Guice Integration — **2** usages

io.joynr.java.common » guice-integration » 1.6.0 — Apache

Guice module, utilities etc to create injectors JOYn and A4A applications

Last Release on Oct 1, 2018

Threadly — **33** usages

org.threadly » threadly » 5.28 — MPL

A library of tools to assist with safe concurrent java development. Providing a unique priority based thread pool, and ways to distrbute threaded work.

Last Release on Oct 1, 2018

Indexed Repositories (879)

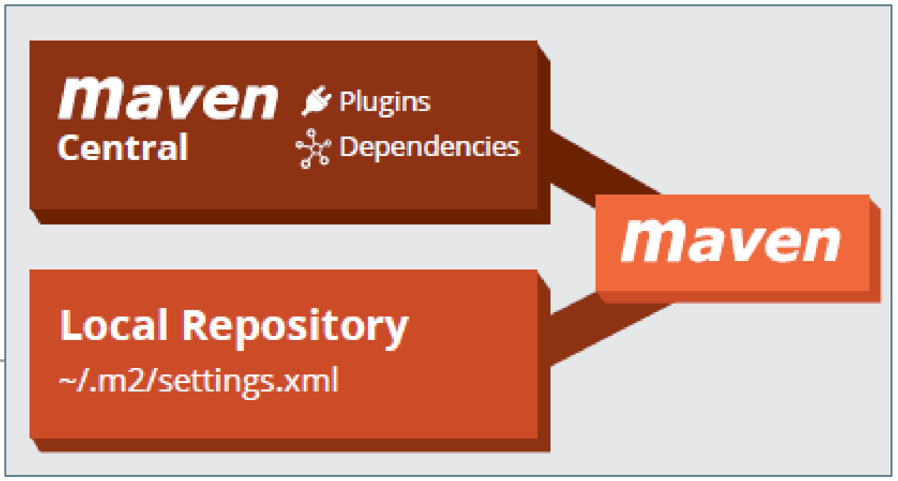
- Central
- Sonatype
- Spring Plugins
- Spring Lib M
- Atlassian
- Hortonworks
- JCenter
- JBoss Releases
- IBiblio
- Spring Lib Release

Popular Tags

android apache api application archetype assets build build-system client clojure cloud config database eclipse example extension github google gradle groovy gwt http ide io jboss library logging maven model module osgi persistence platform plugin repository rest rlang scala sdk security server service spring streaming testing tools ui web webapp webserver

Web site developed by @frodriguez

Coherent organization of dependencies



- Dependencies are automatically downloaded (from remote repositories) and installed (in local repository) for future use.

- Maven knows about some remote repositories, e.g.

  http://www.ibiblio.org/archive/2013/02/ibiblio-tagged-in-maven/

- Other remote repositories can be listed in the project POM or in Maven’s configuration file (setting.xml)

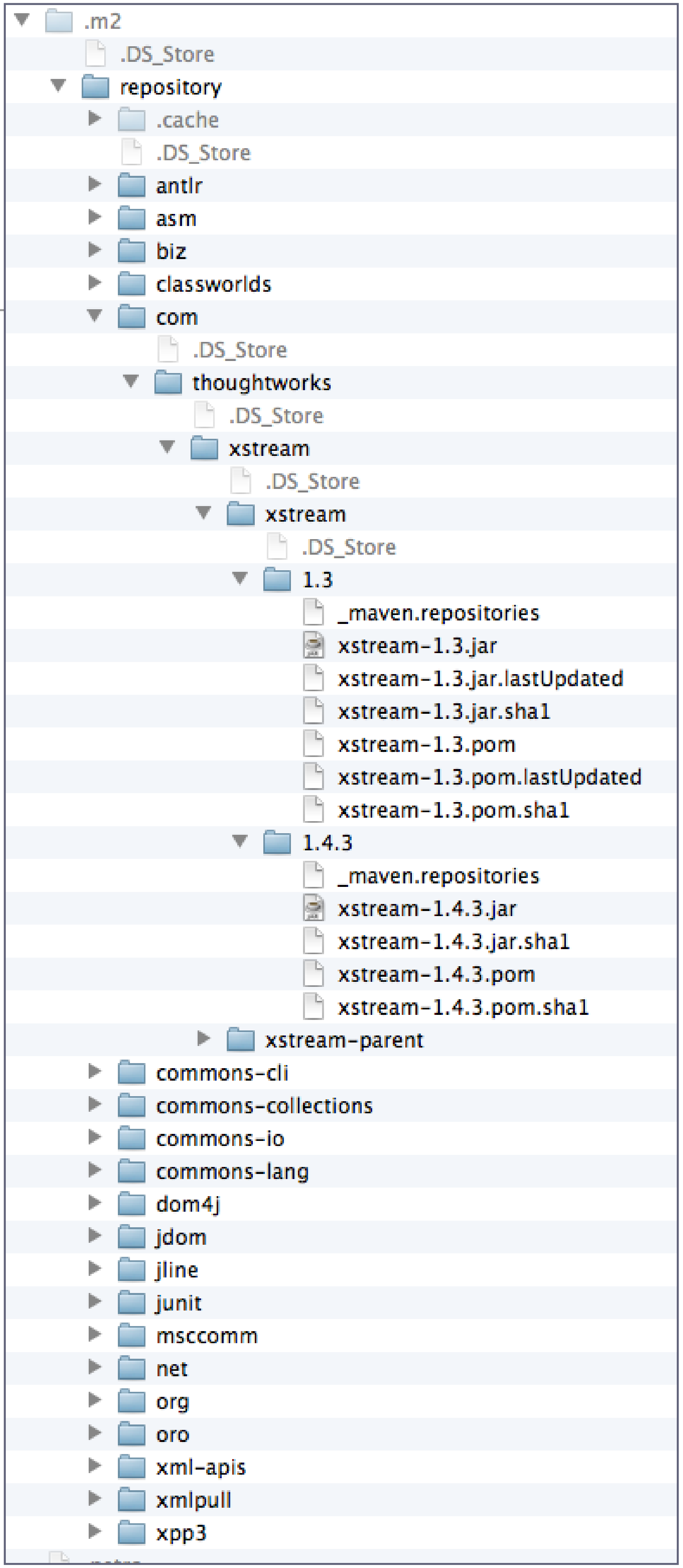
Local repositories

- After installing and running Maven for the first time a local repository is automatically created and populated with some standard artifacts.

- Default Local repository location:
  Home/.m2/repository

- Plugins are also stored in repositories.

- In theory a repository is an abstract storage mechanism, but in practice it is a directory structure in your file system.

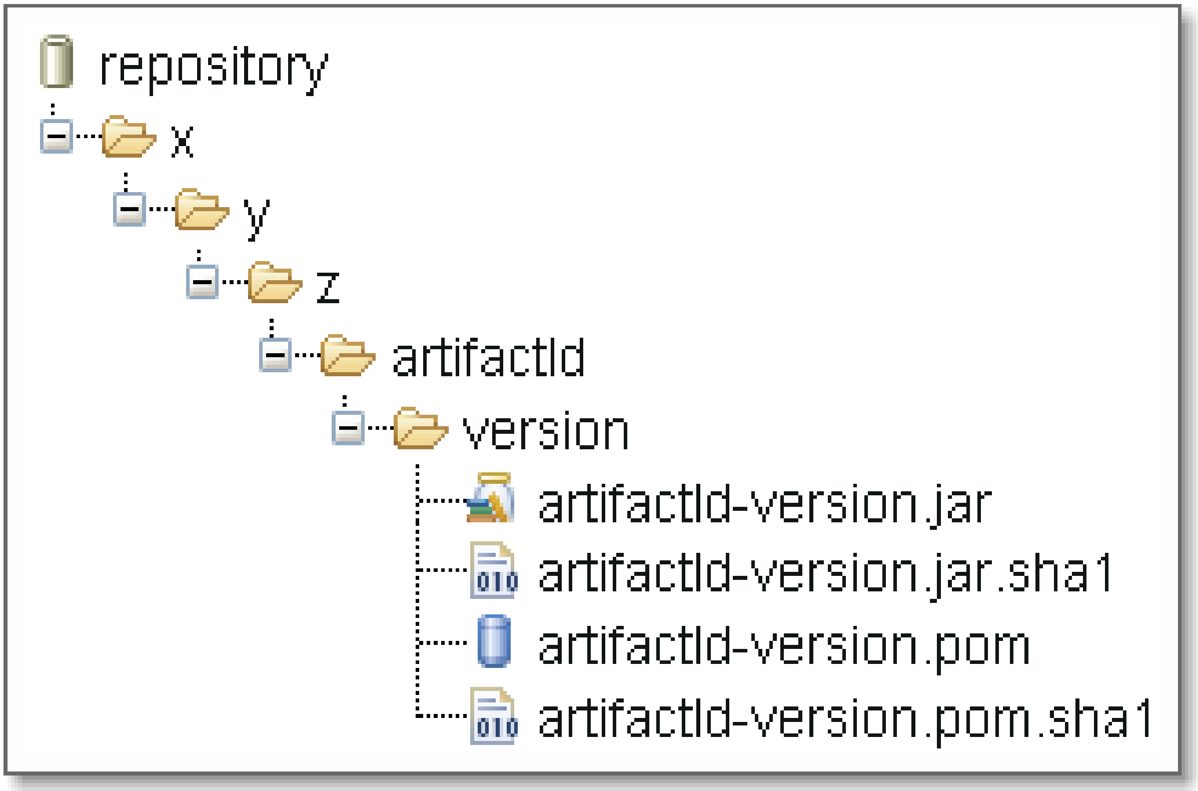


```
▼ .m2
    .DS_Store
  ▼ repository
    ▶ .cache
      .DS_Store
    ▶ antlr
    ▶ asm
    ▶ biz
    ▶ classworlds
    ▼ com
        .DS_Store
      ▼ thoughtworks
          .DS_Store
        ▼ xstream
            .DS_Store
          ▼ xstream
              .DS_Store
            ▼ 1.3
                _maven.repositories
                xstream-1.3.jar
                xstream-1.3.jar.lastUpdated
                xstream-1.3.jar.sha1
                xstream-1.3.pom
                xstream-1.3.pom.lastUpdated
                xstream-1.3.pom.sha1
            ▼ 1.4.3
                _maven.repositories
                xstream-1.4.3.jar
                xstream-1.4.3.jar.sha1
                xstream-1.4.3.pom
                xstream-1.4.3.pom.sha1
          ▶ xstream-parent
    ▶ commons-cli
    ▶ commons-collections
    ▶ commons-io
    ▶ commons-lang
    ▶ dom4j
    ▶ jdom
    ▶ jline
    ▶ junit
    ▶ msccomm
    ▶ net
    ▶ org
    ▶ oro
    ▶ xml-apis
    ▶ xmlpull
    ▶ xpp3
```

Repository structure

- Maven uses artifact’s id, group id. and version to navigate to the correct folder.

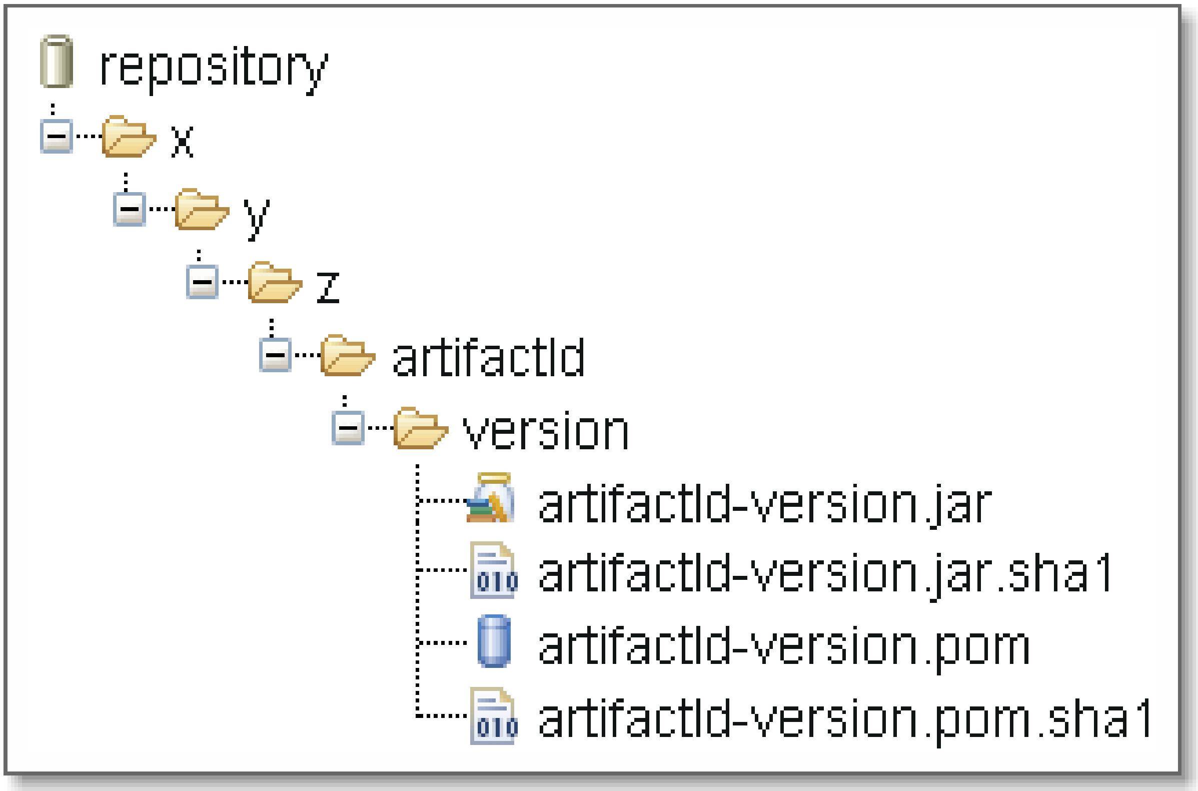
- If the groupId is a fully qualified domain name such as x.y.z then it is fully expanded.



```
repository
└─ x
   └─ y
      └─ z
         └─ artifactId
            └─ version
               ├─ artifactId-version.jar
               ├─ artifactId-version.jar.sha1
               ├─ artifactId-version.pom
               └─ artifactId-version.pom.sha1
```

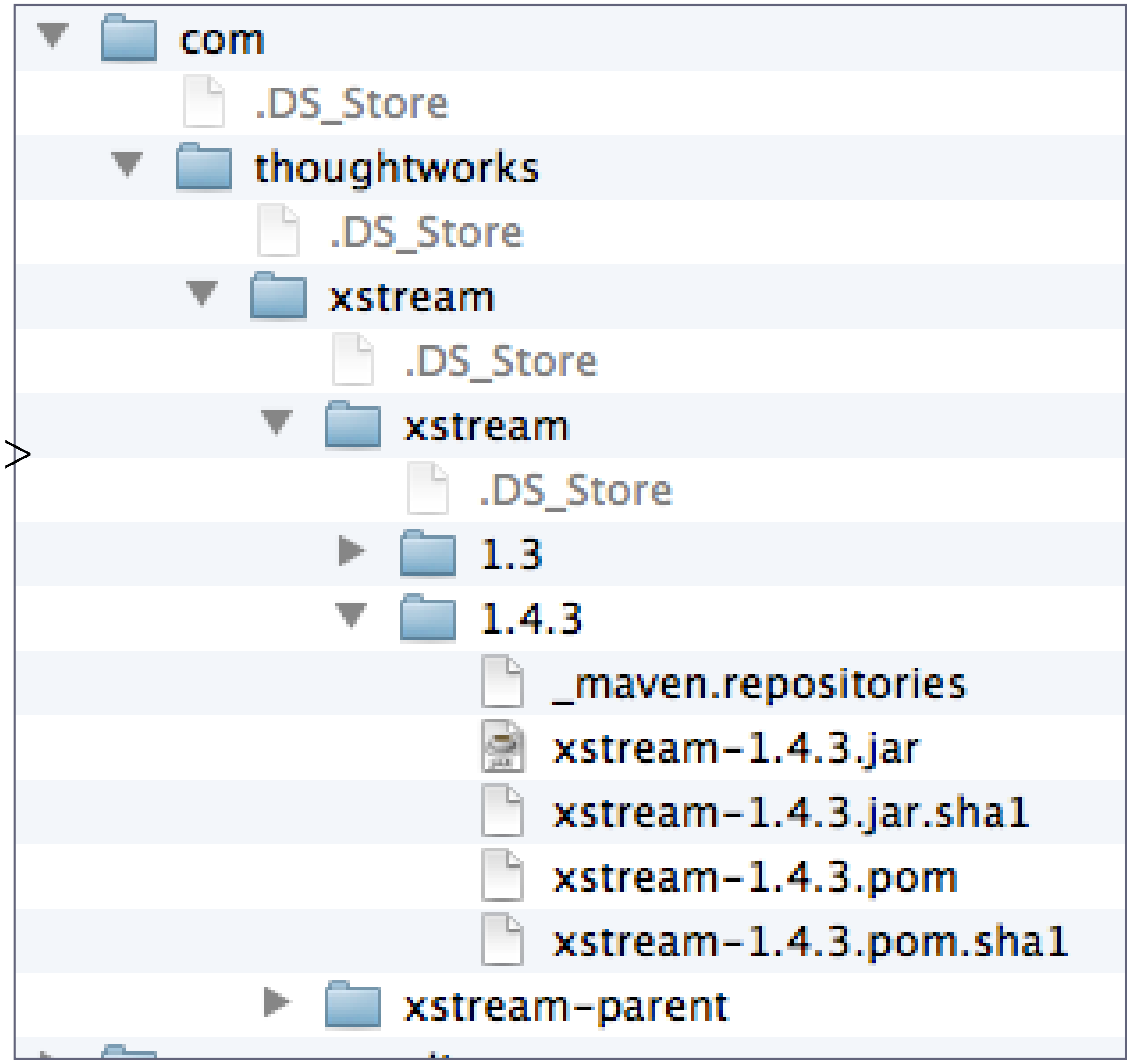
Repository structure

- Maven uses artifact's id, group id. and version to navigate to the correct folder.



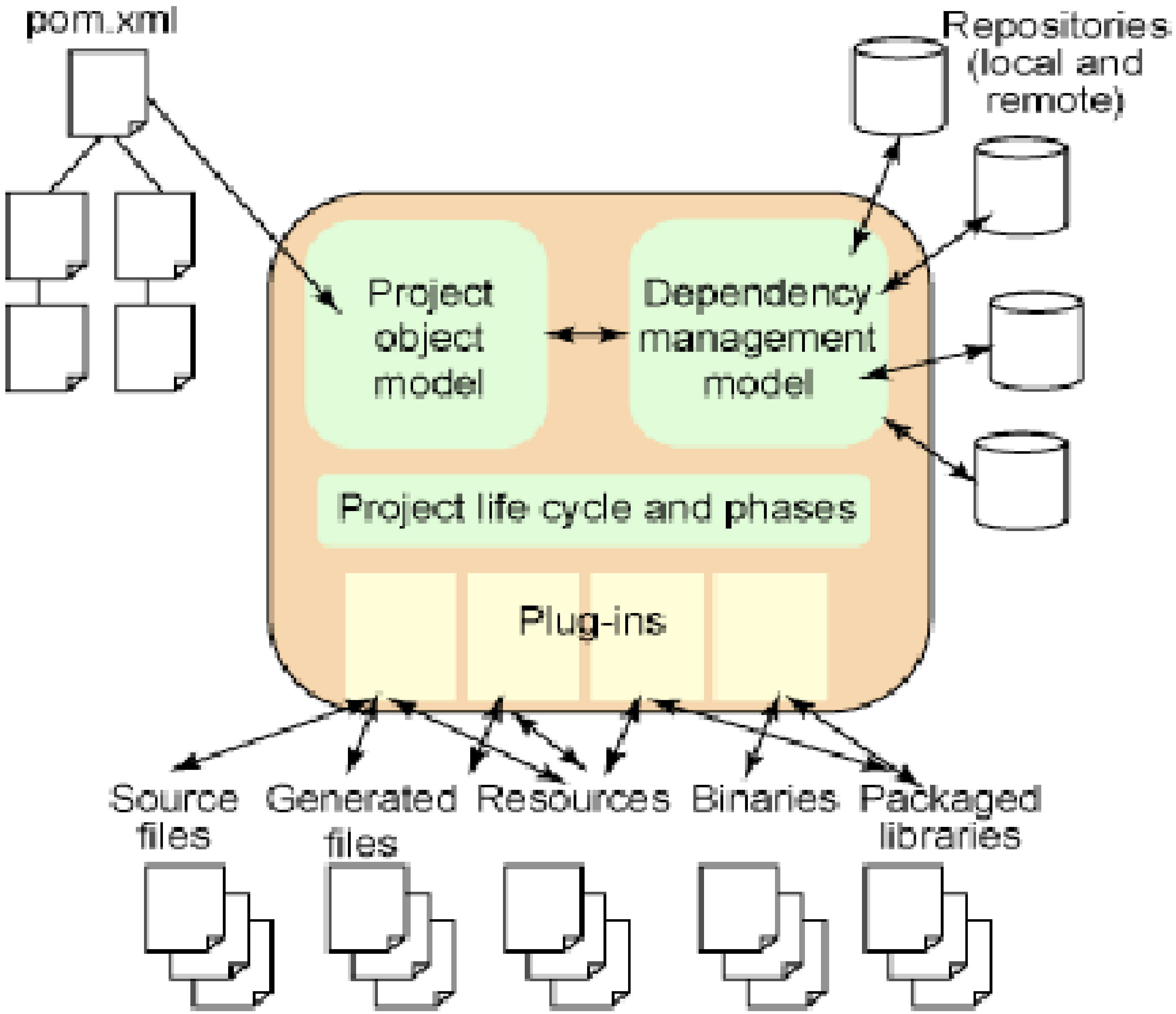
```
repository
└ x
   └ y
      └ z
         └ artifactId
            └ version
               ├ artifactId-version.jar
               ├ artifactId-version.jar.sha1
               ├ artifactId-version.pom
               └ artifactId-version.pom.sha1
```

```
<project>
      .........
 <dependencies>


   <dependency>
     <groupId>com.thoughtworks.xstream</groupId>
     <artifactId>xstream</artifactId>
     <version>1.4.3</version>
   </dependency>

  </dependencies>

</project>
```



```
▼ com
    .DS_Store
  ▼ thoughtworks
      .DS_Store
    ▼ xstream
        .DS_Store
      ▼ xstream
          .DS_Store
        ▶ 1.3
        ▼ 1.4.3
            _maven.repositories
            xstream-1.4.3.jar
            xstream-1.4.3.jar.sha1
            xstream-1.4.3.pom
            xstream-1.4.3.pom.sha1
      ▶ xstream-parent
```

The full picture



Maven™

Plugins and MOJOs

Build Lifecycle

- Maven simplifies and standardizes the project build process. It handles:
  - Compilation
  - Distribution
  - Documentation
  - Team collaboration and
  - Other tasks seamlessly.

http://www.tutorialspoint.com/maven/

Maven Plugins

Maven encapsulates build logic into modules called plugins:

- **Build Plugins:**
  - executed during the build
  - configured in the <build/> element of the POM (if required).
- **Reporting Plugins:**
  - executed during the site generation
  - configured in the <reporting/> element of the POM (if required).

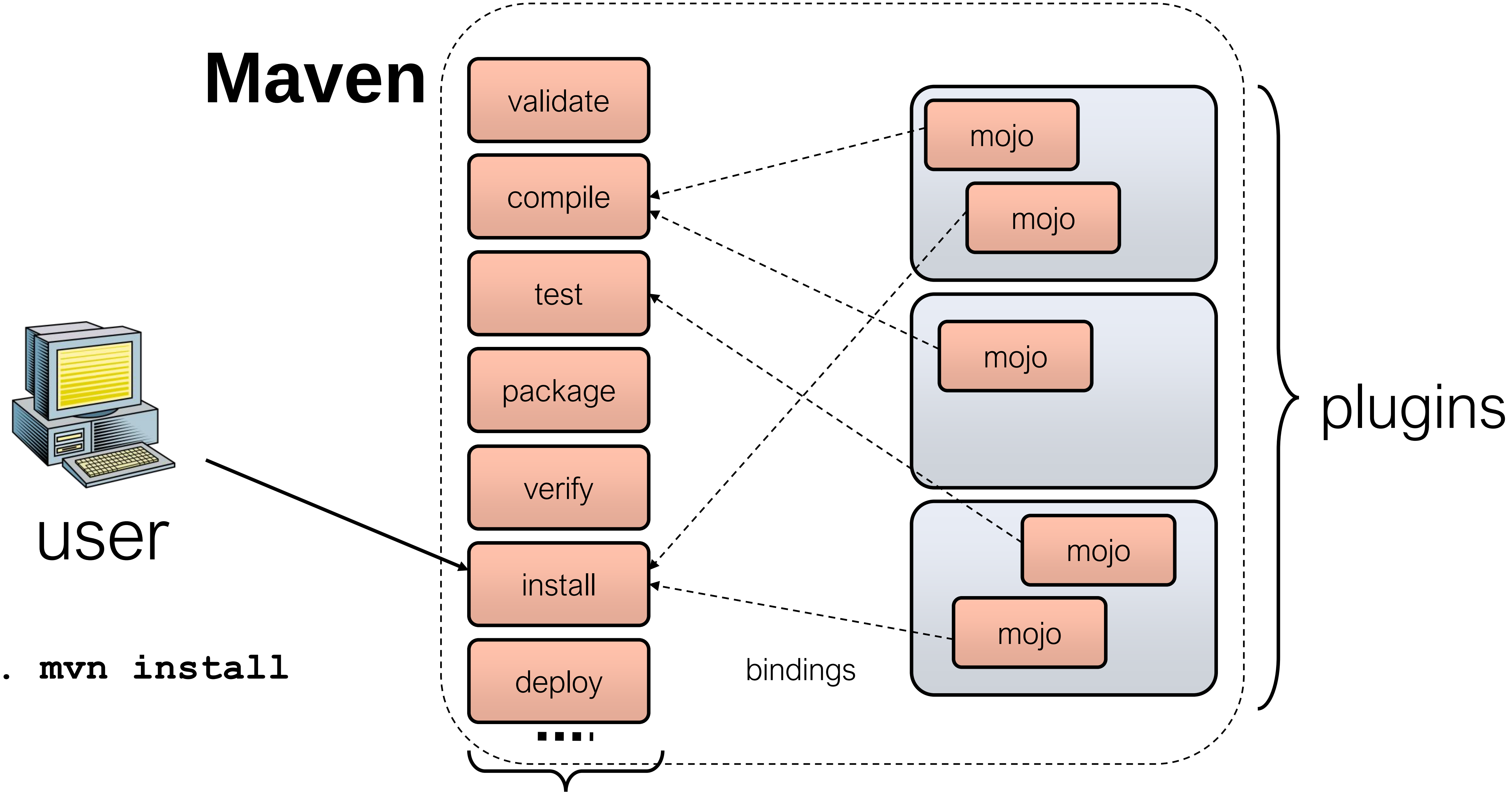
http://maven.apache.org/plugins/index.html

Plugins and MOJOs

- A plugin’s components, called mojos, perform build tasks.
  - MOJO - **M**aven plain **O**ld **J**ava **O**bjects
- Maven acts as a framework which coordinates the execution of plugins in a well defined way.
- Some plugins are standard, others are downloaded on demand.

Plugins and MOJOs

A lifecycle phase invokes the relevant plugins (the mojos) to do the work.



```
e.g. mvn install
```

Build Lifecycle Phases

Build Lifecycle Phases

validate	validate the project is correct and all necessary information is available.
compile	compile the source code of the project.
test	test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed.
package	take the compiled code and package it in its distributable format, such as a JAR.
verify	run any checks on results of integration tests to ensure quality criteria are met.
install	install the package into the local repository, for use as a dependency in other projects locally.
deploy	done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

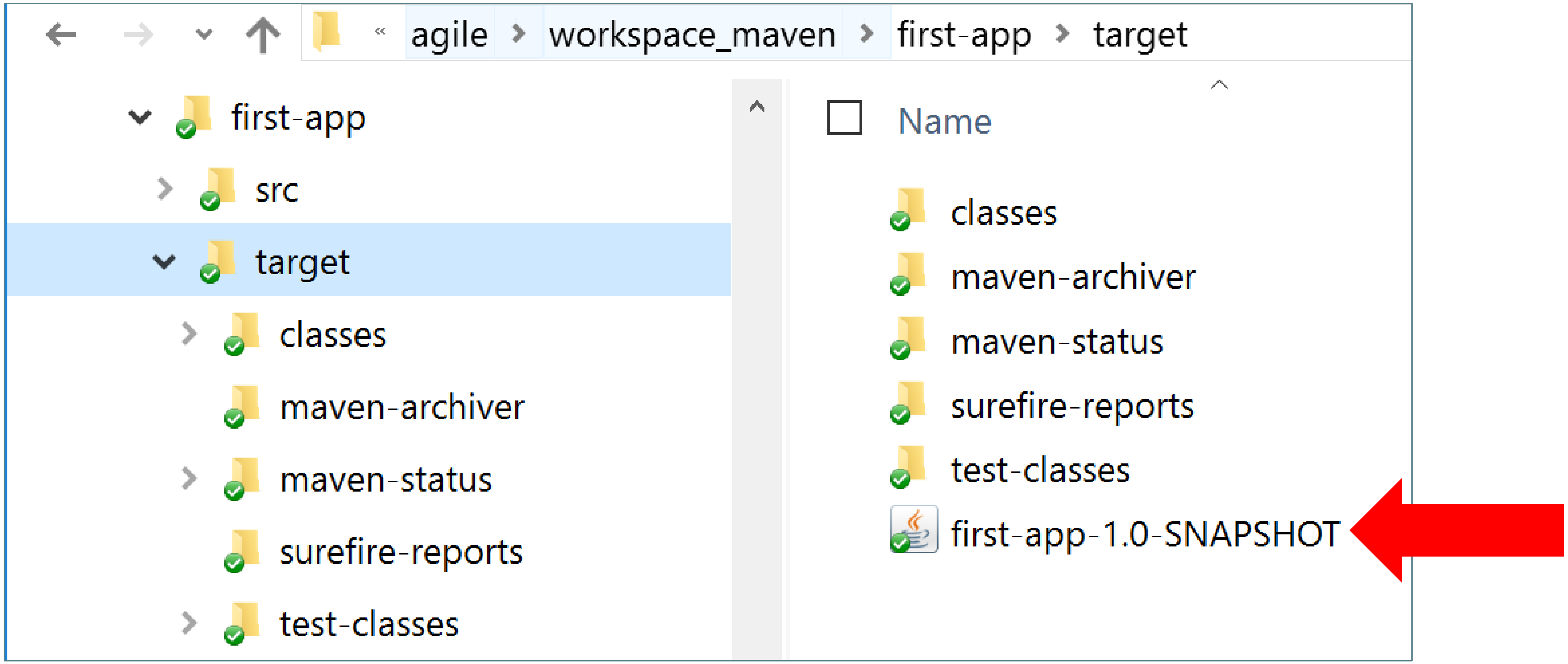
Full lifecycle reference: https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html#Lifecycle_Reference

Declarative Execution

validate
compile
test
package
verify
install
deploy

When user invokes a lifecycle phase, all its predecessors are also executed, if necessary, e.g.

mvn package (invokes validate, compile & test also).



« agile › workspace_maven › first-app › target

- first-app
  - src
  - target
    - classes
    - maven-archiver
    - maven-status
    - surefire-reports
    - test-classes

□ Name

classes
maven-archiver
maven-status
surefire-reports
test-classes
first-app-1.0-SNAPSHOT

Maven™

Archetypes

What are Archetypes?

- Archetype is a Maven project templating toolkit:

  → provides sample maven projects to get users up and running as quickly as possible.

- Using archetypes provides a great way to enable developers quickly in a way consistent with best practices.

https://maven.apache.org/archetype/index.html

Some Archetype Options

maven-archetype-webapp	Web application (WAR) project template
maven-archetype-j2ee-simple	J2EE project (EAR) with directories and subprojects for the EJBs, servlets, etc.
maven-archetype-quickstart (default)	simple Java project (JAR)

We will experiment with this archetype in the next slide deck.

Archetypes

- To create a new project folder structure with the **archetype** plugin, invoke the **generate** goal

Command format: *mvn plugin-name:goal*

Archetypes

- To create a new project folder structure with the **archetype** plugin, invoke the **generate** goal

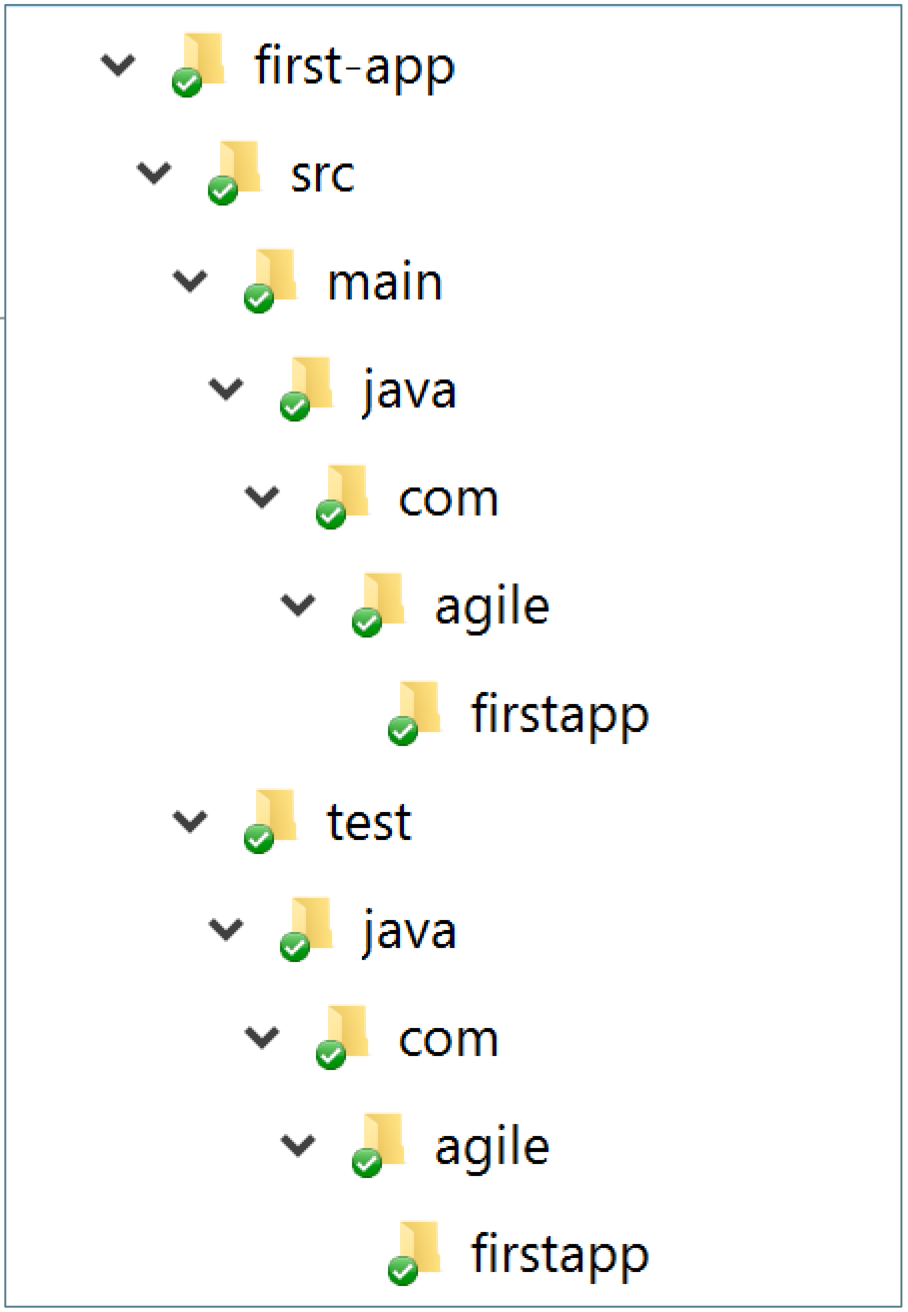
Command format: *mvn plugin-name:goal*

Actual command:

```
mvn archetype:generate
      -DgroupId=[your project's group id]
      -DartifactId=[your project's artifact id]
      -DarchetypeArtifactId=[artifact type]
```

Archetype (Quickstart)

- Folder structure for ‘quickstart’ archetype.
- The base directory name is taken from artifactid.
- A skeleton POM is included in base directory.



```
first-app
  src
    main
      java
        com
          agile
            firstapp
    test
      java
        com
          agile
            firstapp
```

```
mvn archetype:generate
      -DgroupId=com.agile.firstapp
      -DartifactId=first-app
      -DarchetypeArtifactId=maven-archetype-quickstart
```

Archetype (Quickstart)

```
package com.agile.firstapp;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Hello World!" );
    }
}
```

```
mvn archetype:generate
     -DgroupId=com.agile.firstapp
     -DartifactId=first-app
     -DarchetypeArtifactId=maven-archetype-quickstart
```

Maven™

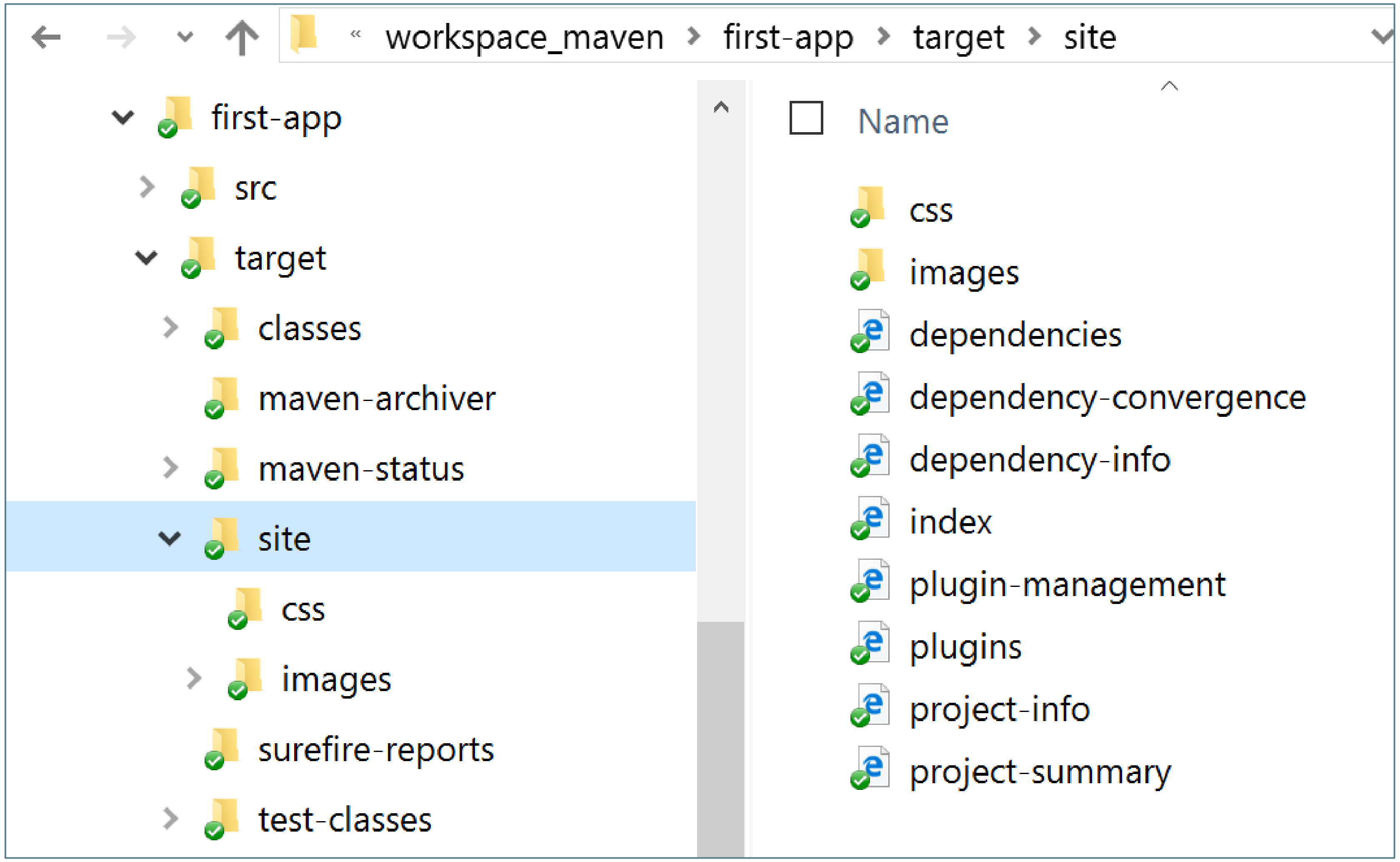
Site Lifecycle

Site Lifecycle

pre-site	execute processes needed prior to the actual project site generation
site	generate the project's site documentation
post-site	execute processes needed to finalize the site generation, and to prepare for site deployment
site-deploy	deploy the generated site documentation to the specified web server

https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html#Lifecycle_Reference

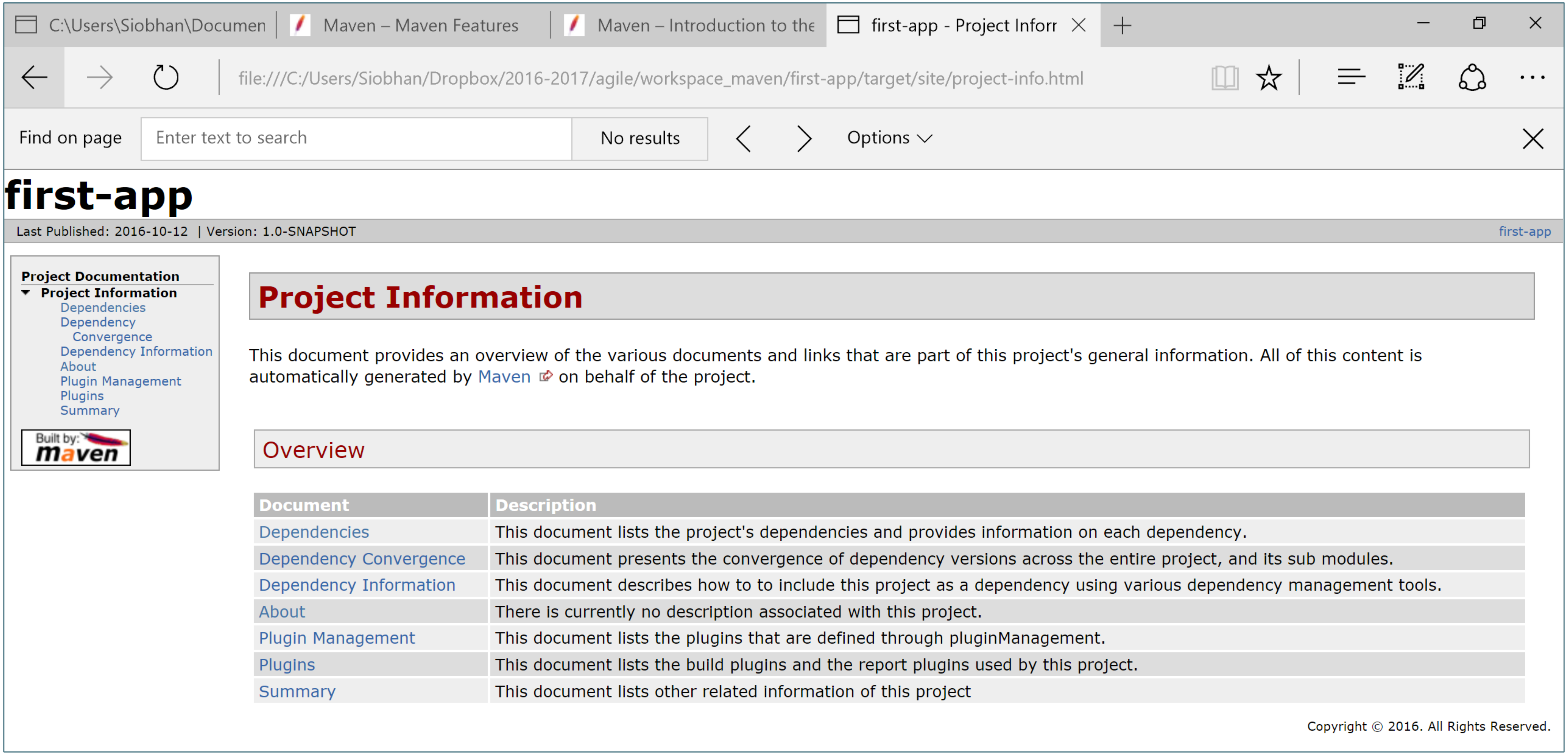
mvn site



```
mvn site
```

generate the project's site documentation

mvn site


C:\Users\Siobhan\Documen
Maven – Maven Features
Maven – Introduction to the
first-app - Project Inforr
file:///C:/Users/Siobhan/Dropbox/2016-2017/agile/workspace_maven/first-app/target/site/project-info.html
Find on page
Enter text to search
No results
Options

first-app

Last Published: 2016-10-12 | Version: 1.0-SNAPSHOT

first-app

Project Documentation

- **Project Information**
  - Dependencies
  - Dependency Convergence
  - Dependency Information
  - About
  - Plugin Management
  - Plugins
  - Summary

Built by: maven

Project Information

This document provides an overview of the various documents and links that are part of this project's general information. All of this content is automatically generated by Maven on behalf of the project.

Overview

Document	Description
Dependencies	This document lists the project's dependencies and provides information on each dependency.
Dependency Convergence	This document presents the convergence of dependency versions across the entire project, and its sub modules.
Dependency Information	This document describes how to to include this project as a dependency using various dependency management tools.
About	There is currently no description associated with this project.
Plugin Management	This document lists the plugins that are defined through pluginManagement.
Plugins	This document lists the build plugins and the report plugins used by this project.
Summary	This document lists other related information of this project

Copyright © 2016. All Rights Reserved.

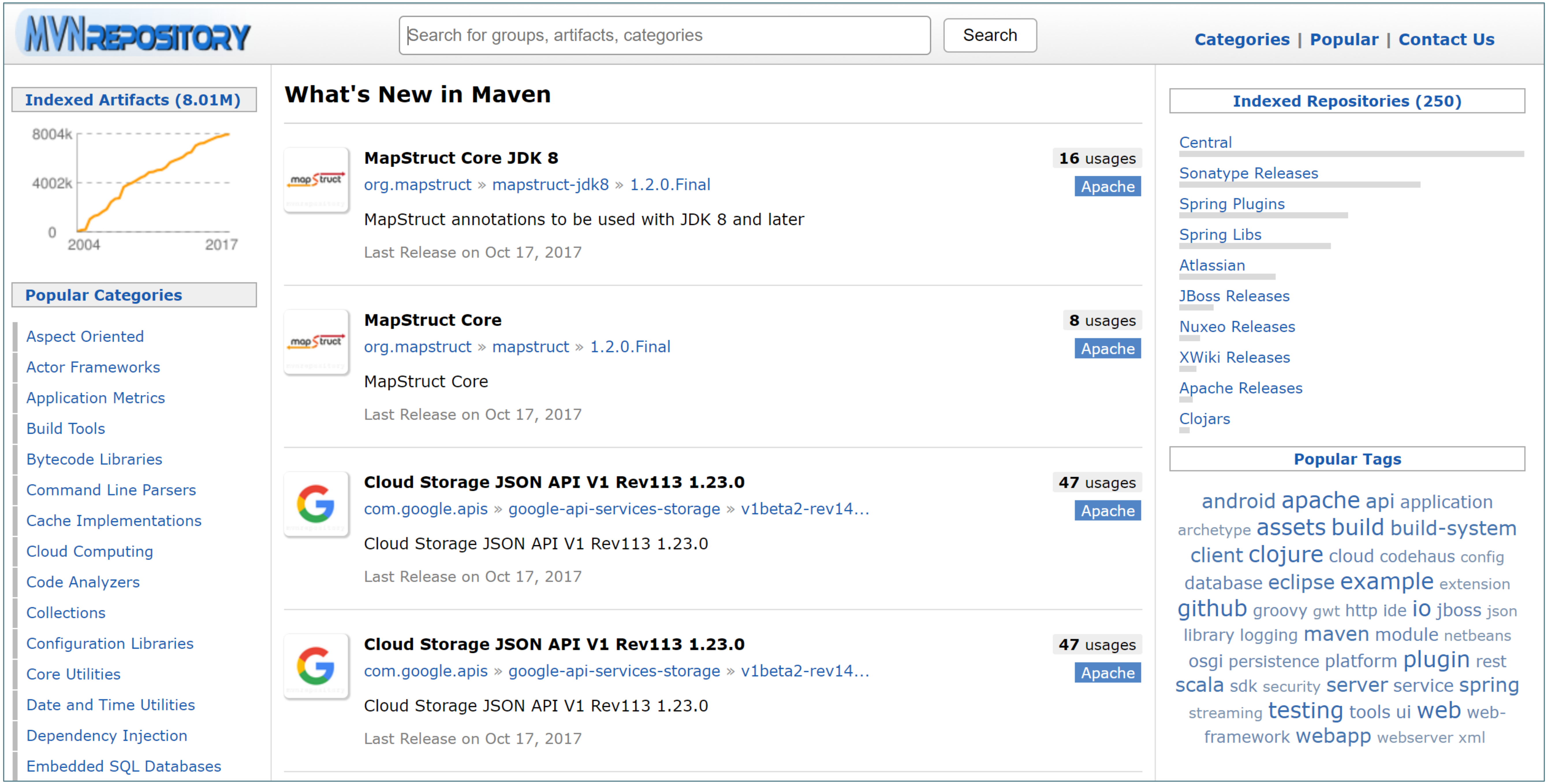
generate the project's site documentation

Maven™

Repository

A repo of our dependencies!

https://mvnrepository.com/



MVNREPOSITORY
Search for groups, artifacts, categories
Search
Categories | Popular | Contact Us
8004k
4002k
0
2004
2017

Indexed Artifacts (8.01M)

Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers
- Collections
- Configuration Libraries
- Core Utilities
- Date and Time Utilities
- Dependency Injection
- Embedded SQL Databases

What's New in Maven

MapStruct Core JDK 8 — 16 usages — Apache
org.mapstruct » mapstruct-jdk8 » 1.2.0.Final
MapStruct annotations to be used with JDK 8 and later
Last Release on Oct 17, 2017

MapStruct Core — 8 usages — Apache
org.mapstruct » mapstruct » 1.2.0.Final
MapStruct Core
Last Release on Oct 17, 2017

Cloud Storage JSON API V1 Rev113 1.23.0 — 47 usages — Apache
com.google.apis » google-api-services-storage » v1beta2-rev14...
Cloud Storage JSON API V1 Rev113 1.23.0
Last Release on Oct 17, 2017

Cloud Storage JSON API V1 Rev113 1.23.0 — 47 usages — Apache
com.google.apis » google-api-services-storage » v1beta2-rev14...
Cloud Storage JSON API V1 Rev113 1.23.0
Last Release on Oct 17, 2017

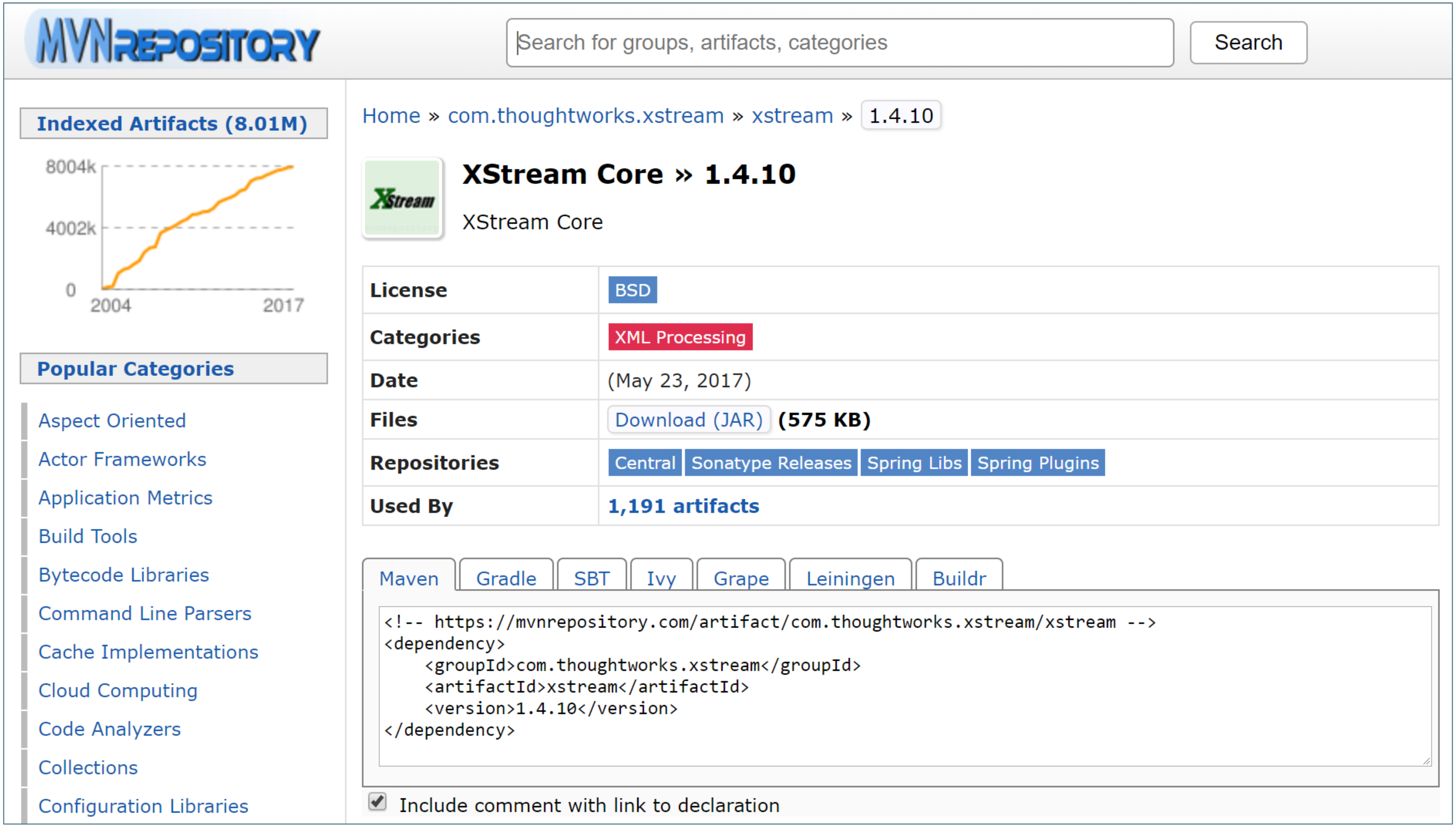
Indexed Repositories (250)

- Central
- Sonatype Releases
- Spring Plugins
- Spring Libs
- Atlassian
- JBoss Releases
- Nuxeo Releases
- XWiki Releases
- Apache Releases
- Clojars

Popular Tags

android apache api application archetype assets build build-system client clojure cloud codehaus config database eclipse example extension github groovy gwt http ide io jboss json library logging maven module netbeans osgi persistence platform plugin rest scala sdk security server service spring streaming testing tools ui web web-framework webapp webserver xml

XStream 1.4.10 (with Maven XML dependency)



Indexed Artifacts (8.01M)

Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers
- Collections
- Configuration Libraries

Home » com.thoughtworks.xstream » xstream » 1.4.10

XStream Core » 1.4.10

XStream Core

License	BSD
Categories	XML Processing
Date	(May 23, 2017)
Files	Download (JAR) **(575 KB)**
Repositories	Central Sonatype Releases Spring Libs Spring Plugins
Used By	**1,191 artifacts**

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
<!-- https://mvnrepository.com/artifact/com.thoughtworks.xstream/xstream -->
<dependency>
    <groupId>com.thoughtworks.xstream</groupId>
    <artifactId>xstream</artifactId>
    <version>1.4.10</version>
</dependency>
```

☑ Include comment with link to declaration

Any questions?