

[Home](#) › [Interview](#) › [Spring, Hibernate, & Maven Interview Q&A](#) › [Git & SVN](#) › ♥ [Git & Maven for releasing software artifacts Q&A](#)

# ♥ Git & Maven for releasing software artifacts Q&A

Posted on [March 27, 2015](#) by [Arulkumaran Kumaraswamipillai](#)

**Q1.** What are the general steps involved in making a software release?

**A1.**

**#1:** Checking out the software artifacts to you build server or local box from a source control management (i.e. SCM) system.

**#2:** Giving it a version so it can be uniquely identified

**#3:** Building, testing and packaging the artifacts.

**#4:** Deploying the built artifacts a repository like Nexus where the artifacts can be picked up for for actual roll out on target servers by the release team.

**#5:** Tagging this state in SCM so it can be associated with the matching artifacts.

## 600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

[open all](#) | [close all](#)

☒ [Ice Breaker Interview](#)

☒ [Core Java Interview C](#)

☒ [JEE Interview Q&A \(3](#)

☒ [Pressed for time? Jav](#)

☒ [SQL, XML, UML, JSC](#)

☒ [Hadoop & BigData Int](#)

☒ [Java Architecture Inte](#)

☒ [Scala Interview Q&As](#)

☒ [Spring, Hibernate, & I](#)

☒ [Spring \(18\)](#)

☒ [Hibernate \(13\)](#)

☒ [AngularJS \(2\)](#)

☒ [Git & SVN \(6\)](#)

☒ [♥ Git & Maven fc](#)

☒ [♥ Merging Vs rel](#)

☒ [♥ Understanding](#)

☒ [6 more Git interv](#)

☒ [8 Git Source cor](#)

☒ [Setting up Cygw](#)

☒ [JMeter \(2\)](#)

☒ [JSF \(2\)](#)

☒ [Maven \(3\)](#)

**Q2.** How will you go about using Git and Maven to perform software releases?

**A2.**

**Step #1:** Checking the software out from Git and checkout if you want to release from a branch

```
1 git clone ssh://user@server:/GitRepos/myproject.g
```

```
1 git checkout my-current-branch
```

**Step #2:** Giving it a version. "0.0.1-SNAPSHOT" in pom.xml becomes "0.0.2-RC1" (that is Release Candidate 1)

```
1 mvn versions:set -DnewVersion=0.0.2-RC1
```

You should not have any "SNAPSHOT" in your multi-module pom files.

```
1 find . -maxdepth 2 -type f -name pom.xml -exec g
```

If you find any "SNAPSHOT", manually update them with something like





```
1 find . -maxdepth 2 -type f -name pom.xml -exec g
```

**Step #3:** Building, testing, packaging and deploying it to an artifact repository like Nexus.

```
1 mvn deploy
```

















You need to have the path to the Nexus configured in your pom.xml with "**distributionManagement**"

```
1 <distributionManagement>
2   <snapshotRepository>
3     <id>myapp-snapshots</id>
```

-  [Testing & Profiling/Sa](#)
-  [Other Interview Q&A 1](#)
-   [Free Java Interview](#)

## 16 Technical Key Areas

[open all](#) | [close all](#)

-  [Best Practice \(6\)](#)
-  [Coding \(26\)](#)
-  [Concurrency \(6\)](#)
-  [Design Concepts \(7\)](#)
-  [Design Patterns \(11\)](#)
-  [Exception Handling \(3\)](#)
-  [Java Debugging \(21\)](#)
-  [Judging Experience I](#)
-  [Low Latency \(7\)](#)
-  [Memory Managemen](#)
-  [Performance \(13\)](#)
-  [QoS \(8\)](#)
-  [Scalability \(4\)](#)
-  [SDLC \(6\)](#)
-  [Security \(13\)](#)
-  [Transaction Managen](#)

## 80+ step by step Java Tutorials

[open all](#) | [close all](#)

-  [Setting up Tutorial \(6\)](#)
-  [Tutorial - Diagnosis \(2\)](#)
-  [Akka Tutorial \(9\)](#)
-  [Core Java Tutorials \(2\)](#)
-  [Hadoop & Spark Tuto](#)
-  [JEE Tutorials \(19\)](#)
-  [Scala Tutorials \(1\)](#)
-  [Spring & Hibernate Ti](#)
-  [Tools Tutorials \(19\)](#)

```

4         <name>MyApp Snapshots</name>
5         <url>http://sdlc/nexus/content/repos
6         <uniqueVersion>false</uniqueVersion>
7     </snapshotRepository>
8     <repository>
9         <id>myapp-releases</id>
10        <name>MyApp Releases</name>
11        <url>http://sdlc/nexus/content/repos
12    </repository>
13 </distributionManagement>
14

```

and the Maven settings.xml configured with user/password to connect to the Nexus repo

```

1 <settings>
2     ...
3     <servers>
4         <server>
5             <id>bfs-releases</id>
6             <username>user</username>
7             <password>password</password>
8         </server>
9         <server>
10            <id>bfs-snapshots</id>
11            <username>user</username>
12            <password>password</password>
13        </server>
14    </servers>
15    ....
16
17    <proxies>
18        <proxy>
19            <id>myproxy</id>
20            <active>true</active>
21            <protocol>http</protocol>
22            <host>proxy.mya pp</host>
23            <port>8080</port>
24            <username>user</username>
25            <password>password</password>
26        </proxy>
27    </proxies>
28 </settings>
29

```

**Step #4:** Tagging this state in SCM.

```
1 mvn scm:tag -Dtag="0.0.2-RC1"
```

You need to have the “SCM” configured in your pom.xml file

```

1 <scm>
2     <url>ssh://git@stash.internal.myhost.com/mypr
3     <connection>scm:git:ssh://git@stash.internal.
4     <developerConnection>scm:git:ssh://git@stash.

```

[Other Tutorials \(45\)](#)

## 100+ Java pre-interview coding tests

[open all](#) | [close all](#)

- [Can you write code? \(](#)
- [♦ Complete the given](#)
- [Converting from A to I](#)
- [Designing your classe](#)
- [Java Data Structures](#)
- [Passing the unit tests](#)
- [What is wrong with th](#)
- [Writing Code Home A](#)
- [Written Test Core Jav](#)
- [Written Test JEE \(1\)](#)

## How good are your .....?

[open all](#) | [close all](#)

- [Career Making Know-](#)
- [Job Hunting & Resum](#)

```
5 </scm>
6
```

**Step #5** Advance with the maven pom.xml tags to 0.0.3-SNAPSHOT to continue with the development work.

```
1 mvn versions:set -DnewVersion=0.0.3-SNAPSHOT
```

**Note:** in your pom.xml file

```
1 <build>
2     <pluginManagement>
3         <plugins>
4             <plugin>
5                 <groupId>org.codehaus.mojo</groupId>
6                 <artifactId>versions-maven-p</artifactId>
7                 <version>1.3.1</version>
8                 <configuration>
9                     <generateBackupPoms>false</generateBackupPoms>
10                </configuration>
11            </plugin>
12        </plugins>
13    </pluginManagement>
14 </build>
15
```

if you have set the “generateBackupPoms” to “true” then you need to run “mvn versions:commit” or “mvn versions:rollback”.

The “scm” plugin needs the following config in the pom.xml file

```
1 ...
2     <scm>
3         <url>ssh://git@stash.internal.mydomain.com</url>
4         <connection>scm:git:ssh://git@stash.internal.mydomain.com</connection>
5         <developerConnection>scm:git:ssh://git@stash.internal.mydomain.com</developerConnection>
6     </scm>
7     ...
8     <build>
9         <pluginManagement>
10             <plugins>
11                 <plugin>
12                     <groupId>org.apache.maven.plugins</groupId>
13                     <artifactId>maven-scm-plugin</artifactId>
14                     <version>1.4</version>
15                     <configuration>
16                         <connectionType>development</connectionType>
17                     </configuration>
18                 </plugin>
19             </plugins>

```

```
20     </pluginManagement>
21 </build>
22
```

The “**versions**” and “**scm**” plugins give you better control in releasing your software artifacts. There is another approach where Maven does most of the hard work for you if your release is very straight forward. This is with the help of the maven “**release**” plugin. In the pom.xml file you need to have

```
1 <plugin>
2   <groupId>org.apache.maven.plugins</groupId>
3   <artifactId>maven-release-plugin</artifactId>
4   <version>2.2.1</version>
5   <configuration>
6     <autoVersionSubmodules>true</autoVersionSubmodules>
7   </configuration>
8 </plugin>
9
```

and then use it in the command-line as:

```
1 mvn release:prepare
2 mvn release:perform
3
```

**You may also like:** [Git source control Q&A | 12 Maven interview Questions & Answers](#) | [7 More Maven interview Questions & Answers](#)

## Popular Member Posts

♦ [11 Spring boot interview questions & answers](#)

904 views

♦ [Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers](#)

816 views

001A: ♦ [7+ Java integration styles & patterns interview questions & answers](#)

427 views

[18 Java scenarios based interview Questions and Answers](#)

408 views

♦ [7 Java debugging interview questions & answers](#)

324 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

311 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

304 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

301 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

251 views

♦ Object equals Vs == and pass by reference Vs value

234 views

8

Like

Share

Tweet

submit

reddit

1

G+1

2

Share

Bio

Latest Posts



## Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via [Amazon.com](http://Amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)



About [Arulkumaran Kumaraswamipillai](#)



Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via [Amazon.com](https://www.amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)

◀ Logging in Java with Logback and MDC Tutorial

10 Java web services written test questions and answers ▶

**Posted in** Git & SVN, Maven, SDLC

**Tags:** Free Content

## Empowers you to open more doors, and fast-track

### Technical Know Hows

☀ [Java generics in no time](#) ☀ [Top 6 tips to transforming your thinking from OOP to FP](#) ☀ [How does a HashMap internally work? What is a hashing function?](#)  
☀ [10+ Java String class interview Q&As](#) ☀ [Java auto un/boxing benefits & caveats](#) ☀ [Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect](#)

### Non-Technical Know Hows

☀ [6 Aspects that can motivate you to fast-track your career & go places](#) ☀ [Are you reinventing yourself as a Java developer?](#) ☀ [8 tips to safeguard your Java career against offshoring](#) ☀ [My top 5 career mistakes](#)

## Prepare to succeed

☀ [Turn readers of your Java CV go from “Blah blah” to “Wow”?](#) ☀ [How to prepare for Java job interviews?](#) ☀ [16 Technical Key Areas](#) ☀ [How to choose from multiple Java job offers?](#)

Select Category ▼

## © Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.