

- [Home](#)
- [About](#)
- [Contact](#)
- [Impressum](#)

Just Java

A blog about Java and (related) Open Source tools...

[Maven + Git] Releasing and Branching done right...

Maven is really awful. I hate it.

But it is by far the best tool available... Therefore I am using it for several years now. And since it is a tool one has to struggle with, I have invested a lot of time into optimizing my build environment. So I know Maven quite well now – better than I wanted...

And of course I am using Git. And I really love it. But unfortunately Git is just a second class citizen in Maven land. The basic things work, but other things don't (like [mvn release:branch](#)).

I have created this checklist for doing a release. Just take a look and compare it with the steps you are doing. I'd like to hear some opinions and ideas for improvements.

Small scripts

I use some scripts that make my life easier:

- [git-publish-branch](#)

The setup

The version numbers

The upcoming release will be 1.0.0 (the first release for this project). Therefore no maintenance branches exist yet.

Currently the master branch is set to 1.0.0-SNAPSHOT.

The branches

Branch	Description
	Contains the current development version (that will be released as next major release).
master	(I have an additional <i>pu</i> branch that is merged to master automatically when all tests have been run successfully, but that doesn't matter for this entry)
next	Contains some experimental code (that will be merged to master one day)
maint- [VERSION]	Maintenance branches for every version. Just bugfixes are committed here and merged up.

Release: What has to be done

The master branch contains the latest peace of software that shall be released (that means that all bugfixes have been merged in from the maint* branches).

- Tag for the release
- Creating a new maint-* branch with updated version number
- Updating the version number for master
- Probably updated the version number for next, too

And: Of course release one version of our software...

The Checklist

1: Ensure all bugfixes have been merged into master

```
1 | git log master..maint-[VERSION]
```

?

2: Creating the maint-branch

```
1 | git checkout master
2 | git checkout -b maint-1.0.0
3 | git publish-branch
```

?

3: Releasing on maint

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

?

4: Merging maint to master

```
1 | git checkout master
2 | git merge maint-1.0.0
```

?

5: Updating the version on master

```
1 | mvn versions:set -DnewVersion=1.1.0-SNAPSHOT -DgenerateBackupPoms=false
2 | git commit -a -m "preparing for next major release: updated version in mas
```

?

6: Merging master to next

```
1 | git checkout next
2 | git merge master
```

?

That merge has probably one merge conflict: The version in the pom.xml. Just keep the value of the next branch.

7: Updating the version for next (optional/if necessary)

```
1 | mvn versions:set -DnewVersion=2.0.0-SNAPSHOT -DgenerateBackupPoms=false
2 | git commit -a -m "preparing release: updated version in next"
```

?

8: Pushing the changes

```
1 | git push
```

?

The result


Now there are three branches with different version numbers:

Branch	Version
maint-1.0.0	1.0.1-SNAPSHOT
master	1.1.0-SNAPSHOT
next	2.0.0-SNAPSHOT

Tags: [git](#), [Maven](#), [scm](#), [vcs](#), [workflow](#)

This entry was posted on Saturday, August 7th, 2010 at 23:02 and is filed under Maven. You can follow any responses to this entry through the [RSS 2.0](#) feed. You can [leave a response](#), or [trackback](#) from your own site.

5 Responses to “[Maven + Git] Releasing and Branching done right...”

-  [Andrey Paramonov](#) Says:
[August 23rd, 2010 at 01:11](#)

Hi Johannes,

Thanks for the nice post. Can you show me the SCM section of your POM? I'm wondering what's the standard (or conventional) way of configuring scm-plugin for local Git repository.

Thanks,
Andrey


P.S. You should vote on <http://jira.codehaus.org/browse/MRELEASE-579>. Maybe it will wake up release plugin developers 😊

-  [johannes](#) Says:
[August 26th, 2010 at 09:44](#)

Usually I am using a central Git repository – connected by ssh. But for that example I used a local one.


The URLs are described here:
<http://maven.apache.org/scm/git.html>

So for local repository it might look like that:
scm:git:file:///home/johannes/tmp/...

-  [Simon](#) Says:
[October 1st, 2011 at 11:31](#)

And how do you manage the conflict of versions in step 6 ? Manually editing pom is annoying when there are many modules. Moreover it does not help for automating the release.

Thank you

-  [Ron](#) Says:
[October 16th, 2011 at 02:12](#)

What is the naming convention for your version #'s? For example: [major release].[minor release].[patch #]. Do you track this with each specific app, or globally?

I'm new to the whole 'release' business, and it seems a bit confusing to keep track of.

I have a parent-pom.xml and many child-pom.xml's for specific applications. Are you releasing each

application here, or globally for the whole thing.



- **Johannes Schneider** Says:

[October 31st, 2011 at 13:29](#)

I don't have a "nice" solution for this step. I merge it manually every time. This isn't too difficult – just the version numbers...

But of course that step should be automated...

Leave a Reply

Name (required)

Mail (will not be published) (required)

Website

-

• Pages

- [About](#)
- [Contact](#)
- [Impressum](#)

• Archives

- [January 2012](#)
- [December 2011](#)
- [October 2011](#)
- [November 2010](#)
- [October 2010](#)
- [August 2010](#)
- [July 2010](#)
- [June 2010](#)
- [May 2010](#)
- [April 2010](#)
- [January 2010](#)
- [December 2009](#)
- [January 2008](#)
- [September 2007](#)

• Categories

- [Java](#) (57)
 - [db4o](#) (1)
 - [Generics](#) (3)
 - [Guice](#) (2)
 - [JavaFX](#) (22)
 - [JFXtras](#) (3)
 - [Maven](#) (6)
- [Linux](#) (3)

[Subscribe RSS](#)

Copyright © 2011 Just Java

[WordPress](#) Theme by [Michael Tyson](#).