SDEVEN Software Development & Engineering Methodology

Version: 7.0.5
Release date: 230619

Versioning (SDEVEN.30-RENVER)

Table of Content

Preamble

This section is about versioning scheme practiced by company.

Mainly this scheme is based on *semantic versioning specifications* or in document *Appendix A Semantic versioning* as appendix to this methodology.

Semantic Versioning specifications have been adapted *regarding the qualification part* (NO CHANGE ref major, minor and patch).

Version Structure

The structure of version string is: `M.m[.p][-buildno][.qual]`, where version parts represents:

- `M`, `m` and `p` are not explained here being exactly like accepted practices ref "*Semantic versioning*"
- `buildno` is the build number with internal applicability and uniqueness identify any build regardless of other version elements / parts
- `qual` is a qualifier and can be one of: `alfa | beta | preview | prerelease | release`; ***NOTE***: preview is just a prerelease but just with a "more commercial" name

Defaulting version numbers

Default `patch`, `build` and `release` numbers will be considered latest (ie, ***biggest numbers***) - see also semantic versioning appendix

Examples:

- *1.1.0-548.preview* major 1, minor 1, patch 0, build 058, preview qualifier version
- *2.1.1-621* major 2, minor 1 patch 1, build 621, last qualifier version
- *3.7* major 3, minor 7, last patch, last build, last qualifier version

Versioning and tagging rules

The most desired situation is when tag names follow versioning principles, ie,name of tags are identically with tagged version.

To achieve this, the following rules must happen:

- in branch `development` always close tags / versions with minimum `beta` qualifier
- from branch `development` DO NOT promote tags with qualifier lower then `pre-release` to branch `master`
- in branch master always close tags / versions with minimum `pre-release` qualifier
- NEVER work directly in branches `development` or `master` - these are only for "collecting" work done in other branches
- when you have (or just need) to tag the current work (and the qualifier cannot be set at minimum requirements) then ***just tag an "alpha" on your branch*** and continue your work on the same branch or on another (depends on your context) and somebody which is admin (maintainer) will make a QA (check & quality assurance) branch from it

Versions-by-branches map

This map shows the most desired actions in frequent and current situations regarding branches and tagging.

So it is organized by branches and shows what kind of version **qualifiers** are recommended for each one.

Recommended qualifiers

With bolded text **YES** was marked the **normal** / usual qualifiers practiced for that kind of branch.

Branch	release	pre-release	preview	beta	alpha
master	**YES**	yes	yes		
development		yes	yes	**YES**	
others (see note)					**YES**

Quality assurance branches

"check & quality assurance" branches are temporary created exactly to promote a qualifier () so allow any qualifier.

Last update: July 18, 2023