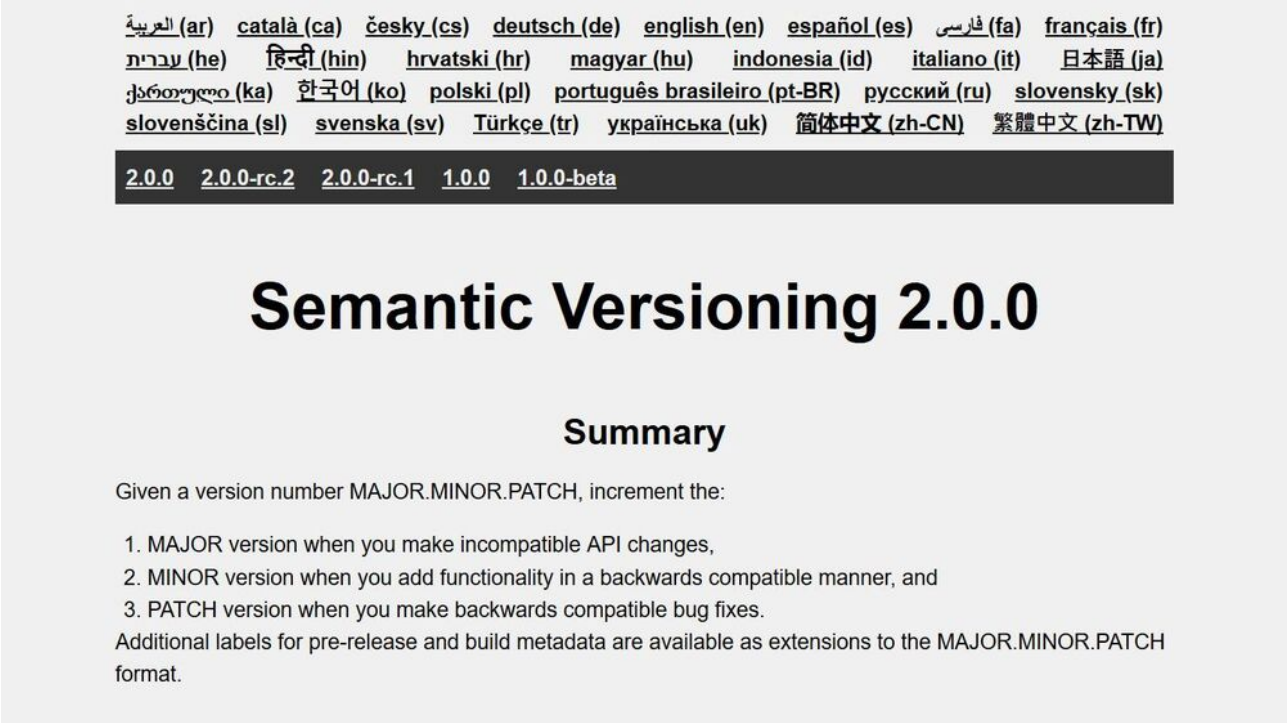
Semantic Versioning for Paperwork: v2.4.1

Prof. Dr. Ulrich Anders, CBS International Business School

Version 2.4.1.en · 2020-06-02



Indispensible in the technical worlds

A whole IT-industry could not work without proper versioning. The versioning system *git* has even become the basis for on of the most central hubs for software development: GitHub. It was recently acquired from Microsoft for around 7.5 billion USD.

Versions show us that things have changed and point us into a direction of how relevant the change is. Was a bug removed, was a new functionality introduced, or has even the interface (so called API) of the package changed? For example, have a look at the software package React. Millions of people are downloading and using this package per week and build their apps based on this technology. For them, it is important to know, if the package has changed, and if yes, in which way.

To immediately see how significant the change was, *Semantic Versioning* was introduced. It is a convention that people use to give some transparency to version changes. Semantic versioning is simple: it uses three digits that represent MAJOR.MINOR.PATCH changes.

Completely ignored for paperwork

Even though semantic versioning is absolutely indispensible in the technical world it has not made its way into the non-technical word. However, semantic versioning would be of similarly help for paperwork. Instead of naming a file like 'MyDocumentFinalFinal2CorrectedOk.pdf' or even worse 'MyDocument.pdf' we could give our document a proper name that defines its state properly: 'MyDocument_v2.4.1.pdf'. It would be easy to just use the convention from semantic versioning:

- MAJOR: larger content changes, restructuring of document, significant re-work, major overhaul of appearance or style.
- MINOR: smaller content changes, additions or deletions of content, improvements of appearance or style.
- PATCH: corrections of grammar or misprints, small formatting improvements, no content changes.

We should copy our document and update its version number each time before we start working on it. At the end of the working session we should review whether the version number reflects the changes properly. In this way we also generate a version history and can always go back to a previous version. At the latest we have to update the version number whenever a document is going to leave the harddisk and is shared with others. Wouldn't it be nice if we would know for every document that we use or receive that we have a specific or the latest version?

A better naming convention

Every document that leaves my harddisk will have a version. But since we are discussing how to make the naming of files more useful we can as well improve it even further. Wouldn't is be useful if we immediately had the following information as well?

- Date or period (if relevant) and in the international accepted format, e.g. 2019-07-31 or 2019-07 or 2019-Q2 or similar.
- Name(s) of the author(s), e.g. Prof-Dr-Ulrich-Anders
- Context, e.g. Blog
- Title, e.g. Semantic-Versioning
- Version, e.g. v2.4.1
- Additions to version, e.g. -wip (work in progress), -alpha (in advanced development), -beta (in testing / review), -rc (release candidate), or -xy for some initials (edited by this person)
- Language, e.g. en
- Format, e.g. pdf

Avoid spaces in these elements and replace them with - hyphens. Separate the sections above with _ underscores.

So here is an example based on this naming convention for files:

2020-06-02_Prof-Dr-Ulrich-Anders_Blog_Semantic-Versioning_v2.4.1.en.pdf

In addition to achieve extra transparency on the content and facilitate the communication with respect to paperwork you are also putting your name on your work. Isn't that a good thing?