Software versioning

by George Frias https://bit.ly/3ccqLfr
https://awwshop.wikidot.com/operations:versioning/

2020-10-21 | 2020-10-22

Used to create WPF software installation with WiX Toolset (For Windows)

Overview

The purpose of this document is to outline information about how I version software in a software development life cycle.

I use a four part version number, it's Major.Minor.Build.Revision. An example of this would be 1.0.0.0 or 2.1.3.132.

The creation of software needs stepping stones or milestones that can be marked by releases. These releases are versioned in a way that shows what part of the development life cycle the software was in. It's only one part of the version number, and it's called the build.

The following Wikipedia links have helped me to define a software versioning system.

- https://en.wikipedia.org/wiki/Software_deployment
- https://en.wikipedia.org/wiki/Software_release_life_cycle
- https://en.wikipedia.org/wiki/Release_management
- https://en.wikipedia.org/wiki/Software_versioning

Version string

I follow a version of Major.Minor.Build.Revision. Out of all of the positions, Build is the only position that will always mean the same thing for all software releases, provided I don't change the meanings for build. At this point if you've been reading from the top of the page with no knowledge of this stuff it might seem confusing, just read on and it should start to make more sense once you know exactly what each position is.

Valid versions	Invalid versions
• 1.0.0.0 • 1.0.0.100 • 2.10.0.0 • 10.0.0.0 • 1.1.1.1 • 10.10.3.10 • 1.0.0.10030	• 1.0.0.00 • 1.0.0.01 • 1.0.01.0 • 1 • 1.0 • 1.0.0 • 0.1.1.1

Valid versions contain all positions, that's Major.Minor.Build.Revision. All positions except for major can be zero to whatever, but cannot start with zero unless it is zero. Major needs to be one or greater.

Version major

The major version is directly related to the product name. Some names include the version in it, others do not. In the case of a no version reference, you need to look it up yourself. From what I've researched, it's standard practice to increase the Major version in the release of a new product. What that means is that the old product is not replaced with the software as an upgrade, it's an entirely new product.

The following software release names are examples for inspection.

Password Wizard 2015

This is a perfect example of a name that does not need the version in it, but it still shows what the more recent release is. For example, Password Wizard 2020 would be a newer version than the 2015 version, and it does not include a version string in the title.

GUID Maker 1

The major version is in the name, so you will always know which version is newer.

Version minor

It's funny that minor is called minor because it gets changed when a big change in the software occurs, or any foundation is changed.

I change the minor version to software if there is a big change in the license. Other than that I haven't increased the minor version. It could also change if the supported operating system versions changed, or any other underlying tech like that was changed.

Version build

Build is the only version whose meaning is always known. For more information about each build, visit the Wikipedia link above.

Here's my build definition:

Value	Build name	Version	Full name
0	Pre-alpha	1.0.**0**.0	GUID Maker 1.0.0.0 Pre-alpha
1	Alpha	1.0.**1**.0	GUID Maker 1.0.1.0 Alpha
2	Beta	1.0.**2**.0	GUID Maker 1.0.2.0 Beta
3	RC (Release Candidate)	1.0.**3**.0	GUID Maker 1.0.3.0 RC
4	RTM (Release to Marketing)	1.0.**4**.0	GUID Maker 1.0.4.0
5	GA (General Availability)	1.0.**5**.0	GUID Maker 1.0.5.0
6	Gold	1.0.**6**.0	GUID Maker 1.0.6.0

Version revision

Every release receives an increase in revision no matter what, and the first release is zero. By the final releases of a product this number will more than likely be high.