Zahava Gopin

WEB 420: discussion 7.1

4/24/2023

Semantic Versioning

“Semantic versioning is a formal convention for determining the version number of new software releases. The standard helps software users to understand the severity of changes in each new distribution” (\_). Semantic versioning consists of three numbers, each of which stand for something else. An example of semantic versioning is 1.2.3. 1 shows a major version, 2 shows a minor version, and 3 shows a patch. By using semantic versioning, the versioning process remains consistent for all software packages.

The major number shows the current version of the package. This number should be changed every time the package is updated, and the user needs to update their work. “The minor number describes the current functional release of your software. This is incremented whenever you add a new feature but do not otherwise alter your package’s interface” (\_). It shows people that a change was made but the package is still fully usable with the previous minor number. The patch number changes when the update does not affect interface or usability of the package. The patch number is used the most often for bug fixes. Projects begin at 1.0.0. Even if your project has a new version, its okay to go back and publish a new feature on the first version.

Semantic Versioning allows developers to add optional pre-release to the version number, to indicate that a release is not yet stable or to provide additional information about the release.

The use of Semantic Versioning helps to ensure that different versions of a software project can be used together, and that developers and users can easily understand what has changed between different versions.

Resources:

Walker, J. (n.d.). *What is Semantic Versioning?* How-to Geek. Retrieved April 24, 2023, from https://www.howtogeek.com/devops/what-is-semantic-versioning/

‌