**Discussion 7.1 – Semantic Versioning**

Patrick Cuauro

Bellevue University

WEB 420-307O RESTFul APIs

Professor Krasso

July 5, 2023

## Semantic Versioning

Semantic versioning (SemVer) is a versioning scheme that helps to communicate the meaning of version numbers. It is a widely used standard for versioning software, and it is especially common in open source projects.

SemVer version numbers are made up of three parts:

* Major version: This number represents significant changes to the API or functionality of the software.
* Minor version: This number represents minor changes to the software, such as bug fixes or new features that are backwards compatible.
* Patch version: This number represents bug fixes that are backwards compatible.

For example, the version number 1.2.3 means that the software has a major version of 1, a minor version of 2, and a patch version of 3. This means that the software has significant changes from version 1.0.0, but it is still backwards compatible with version 1.1.x.

SemVer also defines a set of rules for how version numbers should be incremented. These rules help to ensure that users can understand the meaning of version numbers and make informed decisions about whether or not to upgrade their software.

Here are the rules for SemVer:

* Major version changes should only be made when there are significant changes to the API or functionality of the software.
* Minor version changes should only be made when there are new features or bug fixes that are backwards compatible.
* Patch version changes should only be made when there are bug fixes that are backwards compatible.

Semantic versioning is a valuable tool for software developers and users. It helps to ensure that software is versioned in a way that is meaningful and understandable. This can help to avoid compatibility issues and make it easier for users to upgrade their software.

Here are some of the benefits of using semantic versioning:

* Increased clarity: SemVer version numbers make it clear what changes have been made to the software. This can help users to understand the impact of an upgrade and make informed decisions about whether or not to upgrade.
* Improved compatibility: SemVer helps to ensure that software is backwards compatible with previous versions. This means that users can upgrade their software without having to worry about losing data or breaking their applications.
* Simplified dependency management: SemVer makes it easier to manage dependencies between software projects. This is because SemVer version numbers can be used to ensure that different projects are compatible with each other.

If you are developing software, I encourage you to consider using semantic versioning. It is a simple and effective way to version your software in a way that is meaningful and understandable.

Sources:

<https://www.geeksforgeeks.org/introduction-semantic-versioning/>

<https://blog.axway.com/learning-center/apis/api-management/what-is-semantic-versioning#:~:text=Semantic%20Versioning%20is%20a%20versioning,in%20terms%20of%20backwards%2Dcompatibility>.