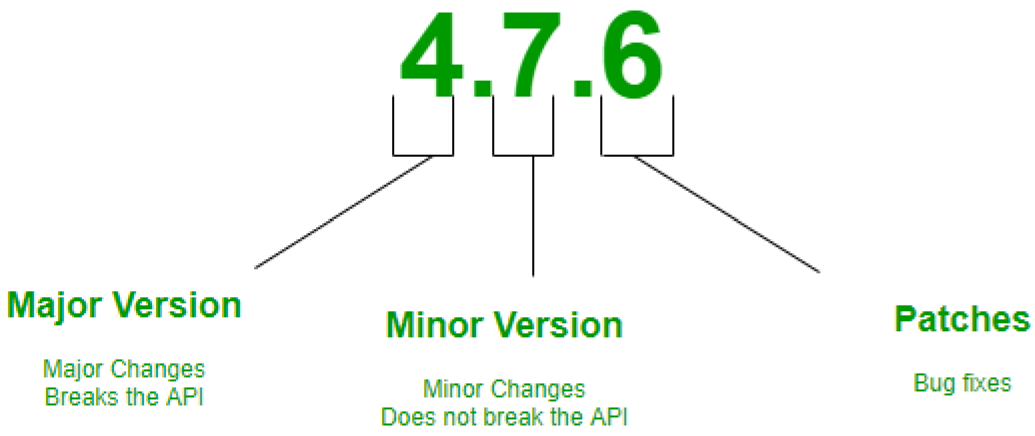
Semantic Software Versioning

Semantic versioning (also known as **SemVer**) is a versioning system that has been on the rise over the last few years. It has always been a problem for software developers, release managers and consumers. Having a universal way of versioning the software development projects is the best way to track what is going on with the software as new plugins, addons, libraries and extensions are being built almost every day. This problem can be solved by Semantic Versioning. In brief, it’s a way for numbering the software releases.

So, SemVer is in the form of **Major.Minor.Patch**.



Semantic Versioning is a 3-component number in the format of X.Y.Z, where :

- **X** stands for a **major version**. The leftmost number denotes a major version. When you increase the major version number, you increase it by one but you reset both patch version and minor versions to zero. If the current version is 2.6.9 then the next upgrade for a major version will be 3.0.0. Increase the value of X when breaking the existing API.
- **Y** stands for a **minor version**. It is used for the release of new functionality in the system. When you increase the minor version, you increase it by one but you must reset the patch version to zero. If the current version is 2.6.9 then the next upgrade for a minor version will be 2.7.0. Increase the value of Y when implementing new features in a backward-compatible way.
- **Z** stands for a **Patch Versions:** Versions for patches are used for bug fixes. There are no functionality changes in the patch version upgrades. If the current version is 2.6.9 then the next version for a patch upgrade will be 2.6.10. There is no limit to these numbers. Increase the value of Z when fixing bugs