semantic versioning

برای تغییر نسخه اپلیکیشن از روش semantic استفاده می‌کنیم که درواقع هر نسخه به شکل major.minor.patch می‌باشد.

> MAJOR version when you make incompatible API changes
>
> MINOR version when you add functionality in a backward compatible manner
>
> PATCH version when you make backward compatible bug fixes

بنابراین هروقت تفاوت نسخه قبلی با نسخه جدید در حد رفع باگ و ریفکتورینگ و بهبود کد است، patch اضافه می‌شود. اگر اضافه کردن قابلیت های جدید در برنامه مثل اضافه کردن یک endpoint جدید باشد، minor اضافه می‌شود و اگر تغییر خیلی بزرگ مثل تغییر در معماری برنامه داشته باشیم و تغییراتی داشته باشیم که باعث شده نسخه جدید با نسخه قبلی backward compatibleنباشد، major اضافه می‌شود.

لازم به ذکر است که اگر هرکدام از این اعداد افزایش یابند، اعداد رو به رویشان به صفر ریست می شوند. مثلا اگر داشته باشیم 2.1.3 و قرار باشد major اضافه شود، نسخه حاصل می شود: 3.0.0 و اگر قرار بوده باشد که minor اضافه شود، نسخه حاصل می شود: 2.2.0.

همچنین برای نسخه های پیش از انتشار از alpha و beta در انتهای نسخه استفاده می‌کنیم. Alpha برای زمانیست که نسخه آماده شده قرار است به دست گروهی برای تست برسد، و beta برای وقتیست که تعدادی از مشتری ها قرار است از این نسخه استقاده کنند و بازخورد بدهند .

این مسائل قابل کاستوم کردن بنابر عرف شرکت می‌باشند.

و نکته آخر در باره ارتباط conventional commits با semantic versioning می‌باشد:

> fix type commits should be translated to PATCH releases. feat type commits should be translated to MINOR releases. Commits with BREAKING CHANGE in the commits, regardless of type, should be translated to MAJOR releases.