# How/why to handle versioning in APIs?

**Question**: ***Why do we need API versions?***

**Answer:** Change in an API is inevitable. Managing the impact of this change can be quite a challenge when it threatens to

break existing client integration, so to handle those impact, we have to add versioning in our app.

**Question**: What are the ways to handle API versioning?

**Answer**: 1. Maintain version number in URI

http://company.com/api/v1.1/api\_name/

2. Add in header of API   
 a.) Use it in header (customer field in header)

Accept-version: v1.1

b.) create custom mimetype for versioning in vnd tree.

Accept: application/vnd.company.myapp-v3+json

Content-Type: application/vnd.company.myapp-v3+json

To register media type : https://www.iana.org/form/media-types

**Question**: what is the version number in API?

**Answer**: *(Version Prefix)(Major Version Number)* ***.*** *(Minor Version Number)*

Ex: ***V2.1***

**Question**: When do we have to up the version our API?

**Answer**: 1. A change in the format of the response data for one or more calls

2. A change in request or response datatype (using float in place of integer)

3. Removing any part of the API.

4. Renaming the API name

**Question**: When do we have not to up the version our API?

**Answer**: 1. Adding new APIs.

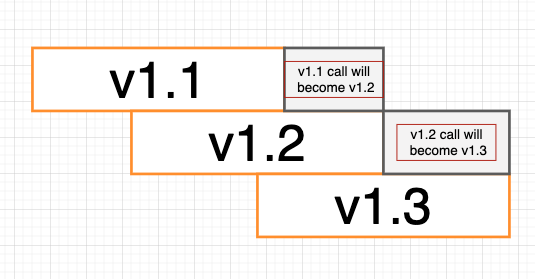
2. Adding new parameter in response (Can up minor version)

**Question**: Do we have to add expiry time in previous version of APIs?

**Answer**: There is a pain for development team to maintain all the version of APIs. So add some time frame (90 days or 180 days) for expiry. and notify all the clients regarding this.

**Question**: What should happen? if some client forgot to implement new API version, and older versions are not available?

**Answer**: In this case It should redirect to latest version of API.



**Question**. What should happen if version is not specified in URI?

**Answer**. It should throw an Error