

Versioning

Une API publiée a des clients que vous ne connaissez pas : des applications mobiles qui ne seront pas mises à jour avant des mois, des scripts oubliés sur un serveur, des intégrations de partenaires... Changer l'API peut tous les casser d'un coup.

Ce qui casse, ce qui ne casse pas

□ Compatible (pas de nouvelle version)	□ Rupture (<i>breaking change</i>)
Ajouter une route	Supprimer ou renommer une route
Ajouter un champ dans une réponse	Supprimer ou renommer un champ
Ajouter un paramètre facultatif	Rendre un paramètre obligatoire
Ajouter une valeur possible à un filtre	Changer le type d'un champ ("9.5" → 9.5)
	Changer le sens d'un code de statut

Côté client, la règle inverse : ignorez les champs que vous ne connaissez pas. Un client qui plante parce qu'un champ a été ajouté est un client mal écrit, même si cela existe parfois.

Où mettre la version ?

Dans l'URL

```
GET /v1/cocktails
GET /v2/cocktails
```

La solution la plus répandue : visible, simple à tester avec curl, facile à router. C'est le choix d'API-ritif et la méthode recommandée dans ce cours.

Dans un header

```
GET /cocktails
X-API-Version: 2
```

Les URL restent stables, mais la version est invisible dans un navigateur ou un lien. GitHub (`X-GitHub-API-Version: 2022-11-28`) et Stripe (`Stripe-Version`) utilisent des dates comme numéro de version.

Dans le media type

```
GET /cocktails
Accept: application/vnd.api-ratif.v2+json
```

La plus « REST » selon les puristes, puisque c'est la représentation qui change. La moins pratique au quotidien.

Faire vivre les versions

- On ne change de version que pour une rupture : `v1`, `v2` ... pas de `v1.3.7` dans l'URL.
- On maintient l'ancienne version pendant une période annoncée.
- On prévient les clients, y compris dans les réponses elles-mêmes, avec les headers `Deprecation` (la version est dépréciée) et `Sunset` (date de fin) :

```
HTTP/1.1 200 OK
Deprecation: @1767225600
Sunset: Wed, 30 Jun 2027 23:59:59 GMT
Link: </v2/cocktails>; rel="successor-version"
```

Le meilleur versioning est celui dont on n'a jamais besoin : une API bien conçue dès le départ, et qui n'évolue que par ajouts compatibles, peut vivre des années en `v1`.

Revision #3

Created 2 October 2026 20:00:48 by Thibaud FRICHET

Updated 13 October 2026 20:18:03 by Thibaud FRICHET