RFC 9745 : The Deprecation HTTP Header Field

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 17 mars 2025

Date de publication du RFC : Mars 2025

https://www.bortzmeyer.org/9745.html

Ce nouveau champ de l'en-tête HTTP sert à indiquer que la ressource demande va être (ou a été) abandonné et qu'il faut penser à migrer. Il sert surtout pour les API HTTP.

(Et si vous ne savez pas ce qu'est une ressource, dans le monde HTTP, voyez la section 3.1 du RFC 9110[1].)

L'idée de ce champ est de permettre aux clients HTTP d'être informés, de préférence à l'avance, du fait qu'une ressource est considérée comme abandonnée et qu'ils devraient donc aller voir ailleurs. Par exemple, si on a une API avec une fonction d'URL `https://api.example.com/v1/dosomething` et que cette fonction est finalement jugée inutile, on pourra faire en sorte que tout appel à cet URL renvoie un champ `Deprecation:` dans l'en-tête, que les clients HTTP comprendront. (Ce champ a été ajouté dans le registre des champs de l'en-tête HTTP <https://www.iana.org/assignments/http-fields/http-fields.xml#field-names>.) `Deprecation:` est juste une information, la ressource continue à fonctionner.

Un exemple est disponible sur ce blog (je n'en ai pas trouvé <https://mailarchive.ietf.org/arch/msg/httpapi/RY2JfY2zihS6utky6O3_rqhyl1M/> de « vraies »), qui a quelques API :

```
% curl -i https://www.bortzmeyer.org/apps/limit
HTTP/1.1 200 OK
Date: Sat, 22 Feb 2025 15:50:23 GMT
Server: Apache/2.4.62 (Debian)
Deprecation: @2147483648
Link: <https://www.bortzmeyer.org/9745.html>; rel="deprecation"; type="text/html"
Content-Type: text/plain

Will be deprecated on 2038-01-19T03:14:08Z.
```

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc9110.txt`

Que veut dire « @2147483648 » ? Le champ `Deprecation:` indique le nombre de secondes depuis l'"*epoch*" Unix. La commande GNU date vous permet de voir facilement cet instant :

```
% date --date=@2147483648
mar. 19 janv. 2038 03:14:08 UTC
```

C'est, dans cet exemple, au moment de la bogue de 2038 que cette ressource ne sera plus maintenue. La syntaxe utilisée est celle des objets de type Date des champs HTTP structurés (RFC 9651, section 3.3.7). D'ailleurs, cette question de syntaxe avait fait l'objet d'une intéressante discussion à l'IETF <https://github.com/ietf-wg-httpapi/deprecation-header/issues/11>, d'autres champs HTTP, comme `Sunset:`, mentionné plus loin, utilisent une autre syntaxe pour les dates. Et vous trouverez, par exemple sur Stack Overflow, d'innombrables articles qui utilisent pour le champ `Deprecation:`, la syntaxe qui avait été initialement proposée (et qui était la même que celle de `Sunset:`).

Cette indication d'un futur abandon ne s'applique qu'à la ressource demandée. Si vous retirez un ensemble de ressources (par exemple votre API avait plusieurs ressources commençant par `https://api.example.co` et désormais vous êtes passé à une nouvelle version, avec le préfixe `https://api.example.com/v2/`), vous devrez le spécifier dans la documentation associée (continuez la lecture, on va parler de cette documentation).

Vous avez sans doute remarqué dans l'exemple plus haut le champ `Link:`. Il sert à donner des détails, sous forme d'un URL où vous trouverez davantage d'informations. Au passage, le moment indiqué dans le champ `Deprecation:` peut être dans le passé indiquant que, quoi que la ressource fonctionne encore, elle est déjà considérée comme abandonnée (et donc sans doute plus maintenue). Les liens sont normalisés dans le RFC 8288 et le type de lien `deprecation` est désormais dans le registre des types de liens <https://www.iana.org/assignments/link-relations/link-relations.xml#link-relations-1>.

Arrivés à ce stade, si vous connaissez bien HTTP, vous vous demandez sans doute « Et `Sunset:` (RFC 8594), alors, il sert à quoi ? » Je dois dire qu'il n'est pas évident d'expliquer pourquoi les deux existent, mais notez quand même que `Deprecation:` est **toujours** plus tôt que `Sunset:`, qui représente donc la fin effective (`Deprecation:` pouvant être simplement une reclassification de la ressource, sans qu'elle cesse de fonctionner). Un exemple donné par le RFC :

```
Deprecation: @1688169599
Sunset: Sun, 30 Jun 2024 23:59:59 UTC
```

Dans cet exemple, la ressource cesse d'être officiellement disponible le 1 juillet 2023 mais n'est réellement retirée que le 30 juin 2024. Après cette deuxième date, on peut supposer qu'à cet URL, on récupèrera un code de retour 404 ou, mieux, 410 (RFC 9110, section 15.5.11).

Quelques articles qui peuvent être utiles lors d'une stratégie de retrait d'une ressource :

— Je vous recommande la lecture de « "*API Lifecycle, Versioning, and Deprecation*" <https://zapier.com/engineering/api-geriatrics/> ». Notez que la solution qu'il propose pour indiquer le futur retrait d'une API n'utilisait pas encore la syntaxe de ce RFC, qui est venu après, mais les concepts sont les mêmes.
— « "*How to deprecate an entire API in OpenAPI ?*" <https://stackoverflow.com/questions/79227185/how-to-deprecate-an-entire-api-in-openapi> » sur Stack Overflow,

— « *"The right way to turn off your old APIs"* <https://httptoolkit.com/blog/how-to-turn-off-your-old-> » (attention, il utilise l'ancienne syntaxe pour exprimer la date d'abandon).

Et le champ `Expires:` (RFC 9111, section 5.3) ? Rien à voir, `Expires:` concerne le contenu, mais l'URL reste stable et fonctionnel.

Un petit mot sur la sécurité (section 7 du RFC). Le champ `Deprecation:` n'est qu'une indication et il ne faut sans doute pas agir automatiquement sur la base de ce champ. Toujours vérifier la documentation !

Pour traiter ce champ, ainsi que le `Sunset:` du RFC 8594, regardez ce script (en ligne sur `https://www.bortzmeyer.org/files/test-deprecation-sunset.py`) Python. Quelques exemples d'utilisation dans le monde :

— L'entreprise de mode Zalando a un guide de la conception d'API qui recommande ce champ <https://opensource.zalando.com/restful-api-guidelines/#189>.
— La mise en œuvre dans la bibliothèque Requests <https://requests.readthedocs.io/> a été discutée <https://github.com/psf/requests/issues/5724> mais le consensus a plutôt été que c'était à l'application utilisatrice de gérer cela.
— Mon client pour l'API RIPE Atlas <https://atlas.ripe.net/>, Blaeu <https://framagit.org/bortzmeyer/blaeu>, a ce code <https://framagit.org/bortzmeyer/blaeu/-/commit/6e546a422c7521b90d4e4c01a4ba3245b4cec775> pour tester si l'API utilisée ne va pas être abandonnée, tiré de l'exemple Python cité plus haut.
— Et bien plus encore qui ont le concept mais avec une autre syntaxe, choisie avant ce RFC.