

HAL et JSON:API

REST dit qu'il faut des liens, mais pas comment les écrire. Plusieurs formats se sont imposés. Voici la même commande dans les deux plus répandus.

HAL

HAL (*Hypertext Application Language*, type `application/hal+json`) est le plus simple : on garde son JSON habituel et on ajoute deux clés réservées.

- `_links` : les liens, indexés par relation ;
- `_embedded` : des ressources liées incluses directement, pour éviter des requêtes supplémentaires.

```
{
  "id": 42,
  "status": "pending",
  "price": 9.5,
  "_links": {
    "self": {"href": "/v1/orders/42"},
    "cocktail": {"href": "/v1/cocktails/3"}
  },
  "_embedded": {
    "cocktail": {
      "id": 3,
      "name": "Segfault Sunrise",
      "_links": {"self": {"href": "/v1/cocktails/3"}}
    }
  }
}
```

HAL permet aussi des liens **modèles** (*templated*), à compléter par le client :

```
"search": {"href": "/v1/cocktails/{?tag,max_price}", "templated": true}
```

JSON:API

JSON:API (type `application/vnd.api+json`) va beaucoup plus loin : il impose la structure complète du document, la pagination, les filtres, le tri, les erreurs et les relations.

```
{
  "data": {
    "type": "orders",
    "id": "42",
    "attributes": {
      "status": "pending",
      "price": 9.5
    },
    "relationships": {
      "cocktail": {
        "data": {"type": "cocktails", "id": "3"},
        "links": {"related": "/v1/cocktails/3"}
      }
    },
    "links": {"self": "/v1/orders/42"}
  },
  "included": [
    {
      "type": "cocktails",
      "id": "3",
      "attributes": {"name": "Segfault Sunrise"}
    }
  ]
}
```

La syntaxe de tri `?sort=-price,name` utilisée dans ce cours vient de JSON:API.

Comparatif

	HAL	JSON:API
Philosophie	Ajouter des liens à son JSON	Un format complet et strict
Données	À la racine, comme d'habitude	Dans <code>data.attributes</code>
Ressources incluses	<code>_embedded</code>	<code>included</code>
Actions (méthode, champs)	Non prévu	Non prévu

	HAL	JSON:API
Prise en main	Immédiate	Il faut lire la spécification
Outillage	Correct	Bibliothèques dans la plupart des langages

Les autres

- **Siren** décrit aussi les **actions** possibles : méthode, URL et champs à envoyer, comme un formulaire HTML.
- **JSON-LD** (et son extension Hydra) vient du web sémantique : il relie les données à des vocabulaires partagés. Google l'utilise pour les données structurées des pages web.

Et API-ritif ?

API-ritif utilise un style **inspiré de HAL** : des `_links` indexés par relation, avec un champ `method` en plus pour les actions. Ce n'est pas du HAL strict, et c'est très courant : beaucoup d'API prennent ce qui les arrange dans HAL.

Pour vos propres API, aucun format n'est imposé. Ce qui compte : des liens cohérents, partout pareils, et documentés dans le contrat.

Revision #3

Created 2 October 2026 20:00:59 by Thibaud FRICHET

Updated 13 October 2026 22:31:40 by Thibaud FRICHET