












## Méthodes HTTP

---

### Sommaire

-  **Méthodes HTTP**
  - **Sommaire**
  -  **Méthodes HTTP : que veut faire le client ?**
  -  **Exemples détaillés**
    -  **GET – Lire une ressource**
    -  **POST – Créer une ressource**
    -  **PUT – Remplacer une ressource**
    -  **PATCH – Modifier partiellement**
    -  **DELETE – Supprimer une ressource**
  -  **À retenir**

## Méthodes HTTP : que veut faire le client ?

Méthode	Action	Exemple	Réponses typiques
GET	Lire une ressource	GET /users/1	200 OK, 404 Not Found
POST	Créer une ressource	POST /users	201 Created, 400 Bad Request
PUT	Remplacer entièrement	PUT /users/1	200 OK, 204 No Content, 404
PATCH	Modifier partiellement	PATCH /users/1	200 OK, 400, 404
DELETE	Supprimer une ressource	DELETE /users/1	204 No Content, 404

### Exemples détaillés

#### ◇ GET – Lire une ressource

```
GET /products/42
```

- ☒ Réponse réussie : 200 OK
- ☒ Produit inexistant : 404 Not Found

#### ◇ POST – Créer une ressource

```
POST /users
{
  "name": "Alice",
  "email": "alice@mail.com"
}
```

- ☒ Création réussie : 201 Created
- ☒ Format incorrect : 400 Bad Request

### ◇ **PUT – Remplacer une ressource**

```
PUT /users/1
{
  "name": "Bob",
  "email": "bob@mail.com"
}
```

- ☒ Succès : 200 OK ou 204 No Content
- ☒ Utilisateur inexistant : 404 Not Found

### ◇ **PATCH – Modifier partiellement**

```
PATCH /users/1
{
  "email": "new@mail.com"
}
```

- ☒ Succès : 200 OK
- ☒ Erreur client : 400 Bad Request
- ☒ Utilisateur absent : 404 Not Found

### ◇ **DELETE – Supprimer une ressource**

```
DELETE /comments/18
```

- ☒ Supprimé : 204 No Content
- ☒ Déjà supprimé / inexistant : 404 Not Found

## **À retenir**

- Chaque **méthode HTTP** correspond à une **intention** (lire, créer, modifier, supprimer)
- Elle est toujours accompagnée d'un **code de retour pour en indiquer le résultat**
- Comprendre et bien choisir ses méthodes rend une API cohérente et prévisible