Créer une API REST avec API Platform

1. Présentation

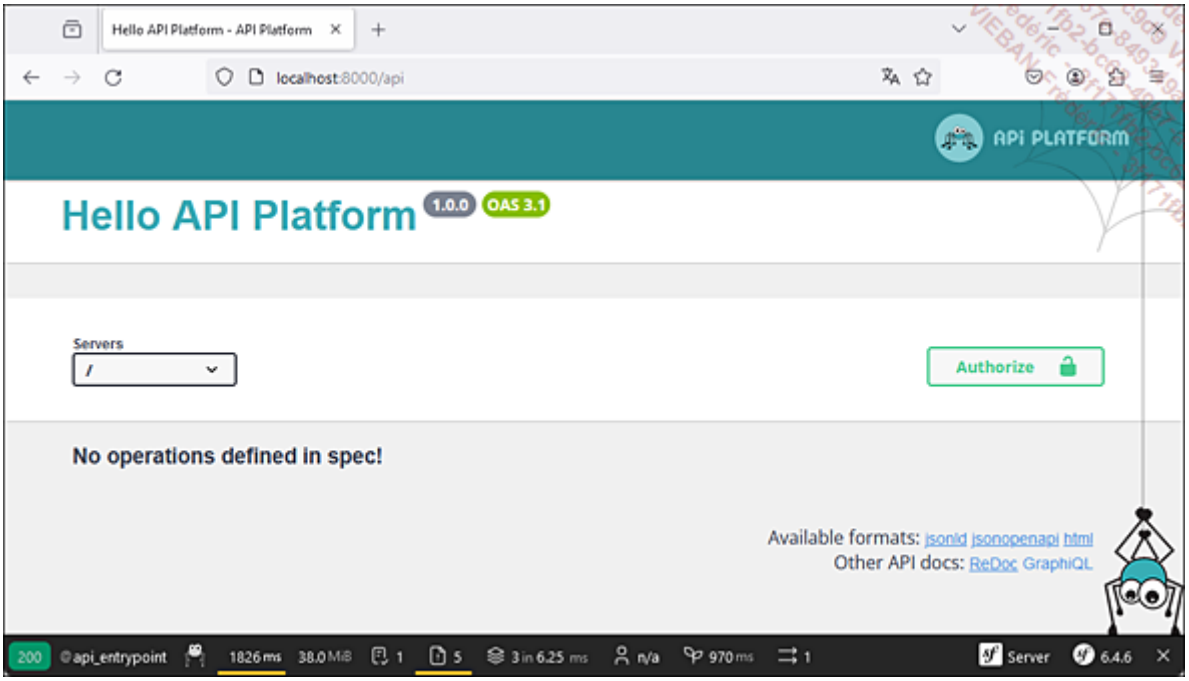
API Platform est une librairie qui permet de créer simplement des API REST. Cette librairie est maintenue par Les Tilleuls.coop (https://les-tilleuls.coop) une entreprise française. Il existe un package permettant d'utiliser toute la souplesse d'API Platform dans Symfony et créer ainsi des API REST en un temps record !

2. Installation

Pour disposer du package API Platform dans son projet, il faut exécuter la recette Symfony correspondante à l'aide de la commande :

composer require api

Une fois la recette exécutée, API Platform est en place. Vous pouvez en valider l'installation en navigant à l'adresse http://<monsite>/**api**.



Pour le moment, aucune opération n'est disponible, mais cela va changer rapidement à l'aide de quelques attributs ou annotations sur les entités Doctrine de l'application.

3. La configuration d'API Platform

Au-delà de la librairie installée dans le dossier **vendor/** du projet Symfony, d'autres fichiers sont apportés par l'installation d'API Platform, notamment :

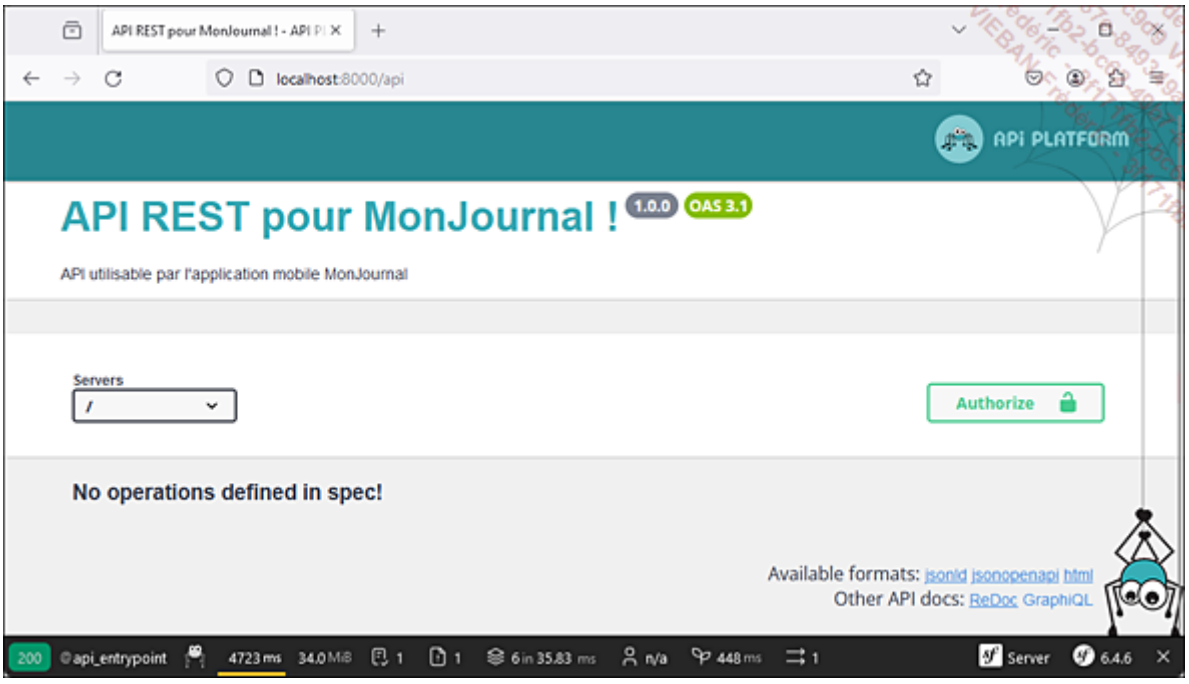
- config/routes/api_platform.yaml : contient le path de la route permettant d'accéder à l'interface de l'API, le path **/api** vient de cette déclaration et peut être modifié.

- config/packages/api_platform.yaml : contient le paramétrage de base d'API Platform, et notamment la référence au répertoire qui contient les entités Doctrine à analyser. Il est également possible d'ajouter des métadonnées pour personnaliser l'interface.

Exemple de personnalisation :

```
api_platform:
  title: "API REST pour MonJournal !"
  description: "API utilisable par l'application mobile MonJournal"
  version: '1.0.0'
  mapping:
    paths: ['%kernel.project_dir%/src/Entity']
  patch_formats:
    json: ['application/merge-patch+json']
  swagger:
    versions: [3]
```

L'interface est ainsi personnalisée :



4. Définition de l'API

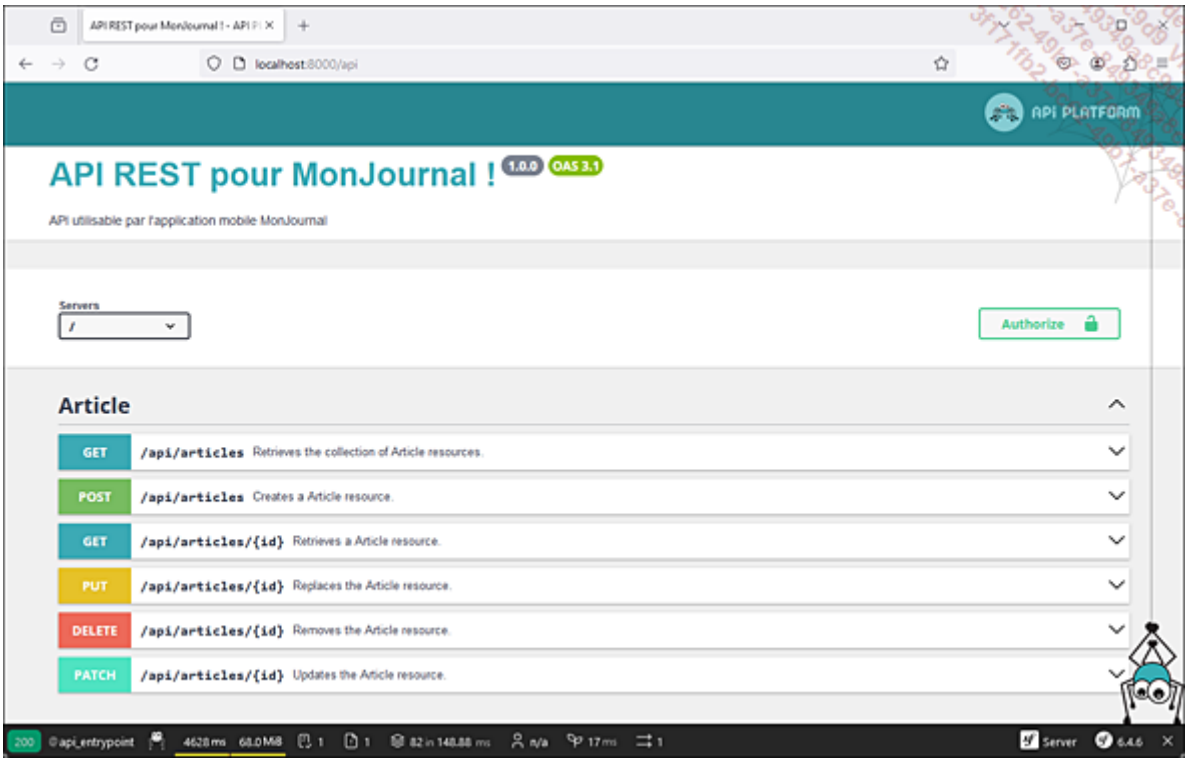
La définition de l'API se fait essentiellement par l'ajout d'attributs ou d'annotation sur les entités Doctrine. En effet, il s'agit de pouvoir implémenter une API qui va permettre de réaliser les opérations CRUD sur les entités :

- Création (**C**reate) : Avec une requête HTTP POST
- Lecture (**R**ead) : Avec une requête HTTP GET
- Mise à jour (**U**pdate) : Avec une requête HTTP PUT
- Suppression (**D**elete) : Avec une requête HTTP DELETE

Pour ce faire, il suffit simplement d'ajouter sur l'entité, l'attribut **#[ApiResource]** de l'espace de noms **ApiPlatform\Metadata** pour exposer une API REST comme dans l'exemple suivant.

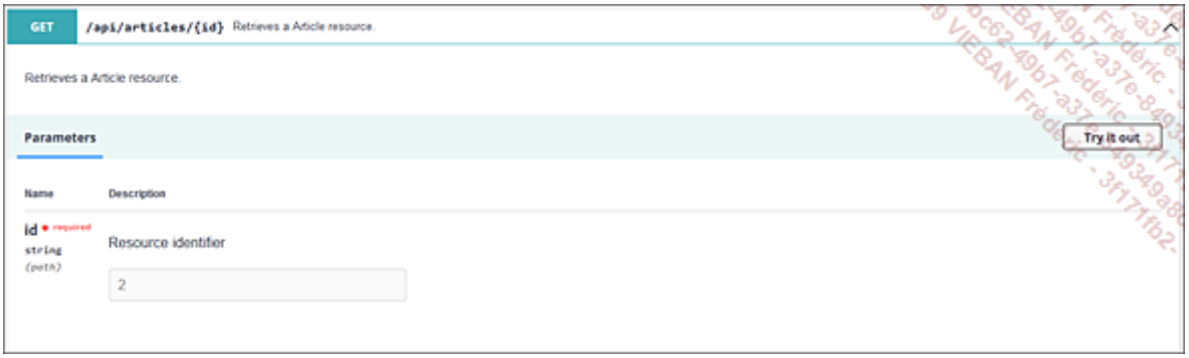
```
<?php


namespace App\Entity;


use Doctrine\ORM\Mapping as ORM;

use ApiPlatform\Metadata\ApiResource;

#[ORM\Table(name: 'article')]

#[ORM\Entity]#[ApiResource]

class Article

{

    // ...

}
```

Et une API REST est automatiquement générée pour la manipulation de cette entité comme l'illustre l'image suivante :

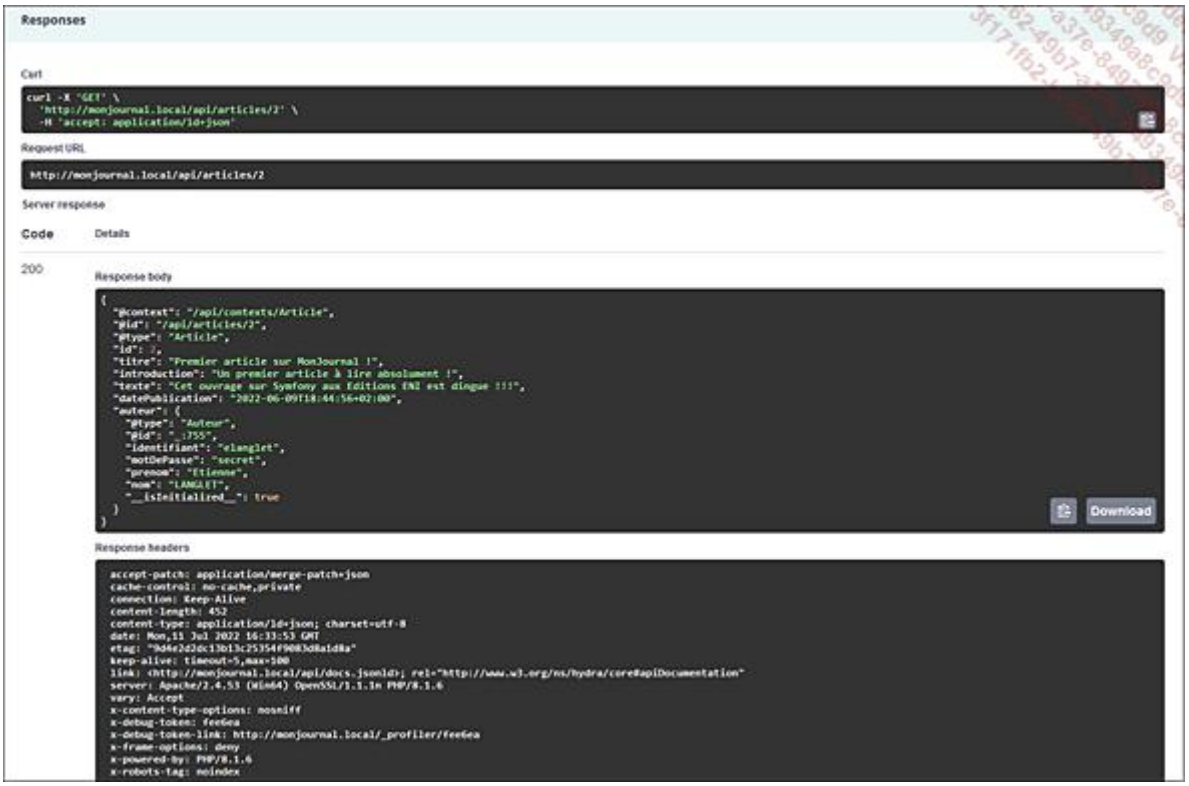


Il est également possible de tester les fonctionnalités directement depuis l'interface en :

- choisissant une opération ;
- en cliquant sur le bouton **Try it out** ;
- en remplissant les éventuels paramètres ;
- et enfin en cliquant sur le bouton **Execute**.



La réponse renvoyée apparaît en dessous :



a. Les opérations de l'API

Une opération est le lien créé entre une ressource, une route et le contrôleur associé. Par défaut, API Platform définit des opérations CRUD sur toutes les ressources, dans l'exemple précédent, on peut créer un article, le modifier, le supprimer et obtenir la liste des articles. API Platform considère deux types d'opérations :

- les opérations sur les collections, appelées **collectionOperations**,
- les opérations sur une entité/une ressource, appelées **itemOperations**.

Ces différentes opérations permettent de distinguer si l'on travaille sur un seul objet ou bien sur un ensemble.

Par défaut, les opérations sur les collections sont au nombre de deux, la première implémentée en HTTP GET permettant d'obtenir la liste des ressources, la seconde en HTTP POST pour ajouter une nouvelle ressource à la liste.

Les opérations sur une seule ressource quant à elles, sont au nombre de quatre :

- Récupération d'une ressource par son identifiant unique (la clé primaire de l'entité), en HTTP GET ;

- Remplacement de la ressource par son identifiant unique en HTTP PUT.
- Mise à jour de la ressource par son identifiant unique en HTTP PATCH.
- Suppression de la ressource par son identifiant unique en HTTP DELETE.

Il est bien entendu possible d'agir sur la génération de ces opérations par défaut et de personnaliser ainsi l'API en fonction des fonctionnalités à fournir.

b. Personnaliser l'API

Pour personnaliser l'API et agir sur les opérations exposées, il faut travailler avec l'attribut **#[ApiPlatform]** et ses deux paramètres : **collectionOperations** et **itemsOperations**. Ces deux paramètres prennent comme valeur la liste des méthodes HTTP dont les opérations doivent être disponibles. Par défaut, comme vu précédemment, les deux opérations de collection et les quatre opérations de ressources sont générées juste avec **#[ApiPlatform]**, la syntaxe complète de déclaration pourrait donc être la suivante :

```
<?php


namespace App\Entity;


use Doctrine\ORM\Mapping as ORM;
use ApiPlatform\Metadata\ApiResource;


#[ORM\Table(name="article")]
#[ORM\Entity]
#[ApiResource(
collectionOperations: ['GET', 'POST'],
itemOperations: ['GET', 'PUT', 'PATCH', 'DELETE']
```

)]

class Article

{

    // ...

}

En agissant sur ces deux paramètres, il est donc possible de contrôler très finement les opérations disponibles dans l'API, ainsi, si par exemple on ne souhaite exposer aucune opération de collection, il suffira de spécifier le paramètre associé avec une liste vide comme dans l'exemple suivant :

#[ApiResource(

collectionOperations: ['GET', 'POST'],

itemOperations: ['GET', 'PUT', 'PATCH', 'DELETE']

)]

Pour plus de possibilités de personnalisation de votre API REST avec API Platform, vous pouvez consulter la documentation officielle (et exhaustive !) d'API Platform à l'adresse https://api-platform.com/docs/distribution/.