

Programowanie

REST API w Symfony bez API Platform – serializer i walidacja

 Jan Kowalski  06.10.2026  211 wyświetleń  0 komentarzy

 Dostępne w językach:

 PL

 EN

[🏠 Start](#) / [Blog](#) / [Programowanie](#)

/ REST API w Symfony bez API Platform – serializer i walidacja

[← Wróć do bloga](#)



Pobierz PDF

Kiedy nie używać API Platform?

API Platform to potężne narzędzie, ale przy prostych projektach bywa przerostem formy nad treścią. Masz prosty CRUD z 3 encjami? Zrób to ręcznie – mniej magii, więcej kontroli.

Struktura kontrolera

```
<?php
```

```
namespace App\Api\Controller;
```

```

use App\Entity\BlogArticle;
use App\Repository\BlogArticleRepository;
use Doctrine\ORM\EntityManagerInterface;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;
use Symfony\Component\Validator\Validator\ValidatorInterface;

#[Route('/api/v1', name: 'api_v1_', format: 'json')]
class ArticleApiController extends AbstractController
{
    public function __construct(
        private readonly BlogArticleRepository $repo,
        private readonly EntityManagerInterface $em,
        private readonly ValidatorInterface $va
    ) {}

    #[Route('/articles', name: 'articles_index', methods: ['GET'])]
    public function index(): JsonResponse
    {
        return $this->json(
            $this->repo->findPublished(),
            200,
            [],
            ['groups' => 'article:list']
        );
    }

    #[Route('/articles/{id}', name: 'articles_show', methods: ['GET'])]
    public function show(BlogArticle $article): JsonResponse
    {
        return $this->json($article, 200, [], ['groups' => 'article:show']);
    }
}

```

Grupy serializacji

Dzięki grupom kontrolujesz które pola są zwracane — inaczej w liście, inaczej w szczegółach:

```
use Symfony\Component\Serializer\Annotation\Groups,

class BlogArticle
{
    #[Groups(['article:list', 'article:show'])]
    private ?int $id = null;

    #[Groups(['article:list', 'article:show'])]
    private ?string $title = null;

    #[Groups(['article:show'])] // tylko w szczeg
    private ?string $content = null;

    #[Groups(['article:list', 'article:show'])]
    private ?\DateTimeImmutable $publishedAt = null;
}
```

Walidacja danych wejściowych

```
use Symfony\Component\Validator\Constraints as Assert

class BlogArticle
{
    #[Assert\NotBlank(message: 'Tytuł nie może być pusty')]
    #[Assert\Length(min: 5, max: 255)]
    private ?string $title = null;

    #[Assert\NotBlank]
    #[Assert\Length(min: 50)]
    private ?string $content = null;
}
```

Uruchamianie walidacji w kontrolerze:

```
$violations = $this->validator->validate($article),

if (count($violations) > 0) {
    $errors = [];
    foreach ($violations as $v) {
        $errors[] = [
            'field'    => $v->getPropertyPath(),
            'message' => $v->getMessage(),
        ];
    }
    return $this->json(['violations' => $errors], '
}
```

Spójny format błędów

Zawsze zwracaj ten sam format — ułatwia debugowanie po stronie frontendu:

```
{
    "status": "error",
    "code": 422,
    "message": "Walidacja nie powiodła się.",
    "violations": [
        { "field": "title", "message": "Tytuł nie może"
        { "field": "content", "message": "Ta wartość n:
    ]
}
```

Kody HTTP — ściągawka

Kod	Znaczenie	Kiedy
200	OK	GET sukces
201	Created	POST sukces
204	No Content	DELETE sukces
400	Bad Request	błędny format JSON
401	Unauthorized	brak tokenu
403	Forbidden	brak uprawnień
404	Not Found	zasób nie istnieje
422	Unprocessable	błędy walidacji
500	Server Error	błąd aplikacji

Tagi: #PHP #Symfony #REST API #Security

Udostępnij: [Facebook](#) [X](#) [LinkedIn](#) [Kopiuj link](#)

Komentarze (0)

Bądź pierwszy — zostaw komentarz!

[Zaloguj się](#), aby dodać komentarz.

 **FactoryCode**

Projekt demonstracyjny zbudowany na Symfony 8, PHP 8.4 i PostgreSQL 16.
Tłumaczenia zarządzane przez panel admina z AI.

NAWIGACJA

 [Start](#)

 [Blog](#)

 [Projekty](#)

 [Kontakt](#)

 [Polityka prywatności](#)

KONTAKT

 kontakt@factorycode.pl

 [GitHub](#)