2 minutes

Exception avec AbstractController

La méthode `createNotFoundException()` de la classe abstraite `AbstractController`

Voici la méthode :

```
<?php

protected function createNotFoundException(string $message = 'Not Found', \Throwable $previous = null): NotFoundHttpException
{
  return new NotFoundHttpException($message, $previous);
}
```

Copier

Cette méthode permet de retourner une réponse `HTTP` avec le code de statut `404` pour `Not Found` (non trouvé).

Elle permet de retourner une page d'erreur lorsque l'élément demandé n'est pas trouvé.

Par défaut le message est simplement `'Not Found'`. Il suffit de passer une chaîne de caractères en premier argument pour le modifier.

Par exemple :

```
<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

class DefaultController extends AbstractController
{
  #[Route('/', name: 'index')]
  public function index(): Response
  {
    throw $this->createNotFoundException('Rien à cette adresse');
  }
}
```

Copier

La méthode `getParameter()` de la classe abstraite `AbstractController`

Voici la méthode :

```
<?php
protected function getParameter(string $name)
{
  if (!$this->container->has('parameter_bag')) {
    throw new ServiceNotFoundException('parameter_bag.', null, null, [], sprintf('The "%s::getParameter()" method is missing a parameter bag to work p
  }

  return $this->container->get('parameter_bag')->get($name);
}
```

Copier

Nous reverrons en détail cette méthode dans le chapitre dédié au conteneur de services.

Nous pouvons déjà dire qu'elle permet de récupérer un paramètre déclaré dans le fichier `config/services.yaml` et plus précisément dans l'option `parameters`.

C'est l'endroit où il faut déclarer des variables globales à l'application et qui ne changent pas suivant la ou les machine(s) où sera exécutée l'application.