Académie Lille

Formation Drupal 8:
Cas pratiques

Version : J2
02/10/2018

SMILE
I.T IS OPEN

Table des matières

I MODULES COMMUNAUTAIRES

I.1 MODULES COMMUNAUTAIRE

I.1.a Installer un module

Pour installer un module, la méthode recommandée est composer (exemple essentiel : admin toolbar)

```
composer require admin_toolbar:8.x
```

Pour vérifier qu'un module existe : https://www.drupal.org/project/**{project name}**

I.1.b Activer un module

Pour activer un module, la commande recommandée est drush :

```
# Les commandes drush et drupal console fonctionnent depuis le répertoire
# du fichier index.php
cd /var/www/drupal8
drush -l d8-lille.lxc en admin_toolbar -y
# -l : le chemin du site (pour utiliser drush avec un multisite / env)
# -y : accepter automatiquement l'action
```

- Utiliser Admin Toolbar pour vider le render cache
- Activer les modules field_group, paragraph et field group
- Installer et activer le module examples

I.2 ADMINISTRATION DE BASE (PRÉ-REQUIS)

- Créer une nouvelle entité de type paragraphe
- Ajouter un field group rétractable autour d'un ensemble de champs de l'entité
- Ajouter un champ de article comme référence à l'entité paragraphe nouvellement créée

I.3 DEVEL

Devel est un outil d'assistance au développement

- Activer et installer les la suite de modules Devel, Kint et Devel generate
- Générer du contenu automatiquement
- Accéder à l'onglet « Devel » sur les entités nouvellement générés

I.3.a Dumper des variables

Kint est comparable à var_dump mais plus puissant : il permet d'afficher des informations sur les méthodes de classes, de lire la phpdoc et de formatter efficacement les imbrications de données (tableaux, objets).

```
Kint(‘kint is better than var dump’) ;
# Régler la profondeur des objets pour éviter les dépassements mémoire :
kint_require();
Kint::$maxLevels = 3;
```

I.3.b Exécuter du code

- Retrouver l’entité Paragraph précédente avec Kint, inspecter son contenu.

```
// la valeur de $id se cache dans les données de l’article
Paragraph::load($id) ;
```

I.4 DÉSINSTALLATION

- Supprimer toutes les instances d’entité node et paragraphe précédentes (/admin/modules/uninstall)

I.5 PATCHER UN CODE COMMUNAUTAIRE

- Ajouter **cweagans/composer-patches** avec composer
- Installer le module broken link en version 2.2
- Appliquer le patch https://www.drupal.org/project/broken_link/issues/2938885
- Vérifier que le module fonctionne

II MODULES SPÉCIFIQUES

II.1 STRUCTURE DE BASE

- Faire apparaître un module d8_lille dans le Backoffice, en ajoutant simplement un dossier et un fichier d8_lille.info.yml

II.1.a Ajouter un plugin yml

- Sur la page qui liste les types de contenu, ajouter un item de menu de type « action » qui pointe vers l'édition des paragraphes. Créer un fichier d8_lille**.links.action.yml** dans votre module à partir des informations suivantes :

  route_name : entity.paragraphs_type.collection, **appears_on** : entity.node_type.edit_form.

  Documentation: https://www.drupal.org/docs/8/api/menu-api/providing-module-defined-local-actions

- Ajouter un nouveau type de permission (d8_lille.permissions.yml) nommé 'access images wall' Exemple: https://api.drupal.org/api/drupal/core%21modules%21node%21node.permissions.yml/8.6.x

II.2 ANNOTATIONS (PLUGINS)

II.2.a Les blocs

- Le but de cet exercice est d'afficher un Block au dessus du titre, qui contient le **nom de l'utilisateur** courant, le **nom de la route** visitée et le **nombre de paramètres** dans l'url.

Créer une classe de Block Plugin en suivant la documentation officielle :
https://www.drupal.org/docs/8/creating-custom-modules/create-a-custom-block

Récupérer la route courante :

```
\Drupal:routeMatch() ;
```

Récupérer l'utilisateur courant :

```
\Drupal:currentUser() ;
```

Récupérer les informations de la requête courante (nombre de paramètres) :

```
\Drupal:request() ;
```

Utiliser Devel pour inspecter les informations des objets ci-dessus.

Utiliser les contextes de cache correspondants : user, route & url ...
https://www.drupal.org/docs/8/api/cache-api/cache-contexts

Securité: Utiliser la fonction de traduction **t** avec des caractères d'échappement (**$args)**
https://api.drupal.org/api/drupal/core%21includes%21bootstrap.inc/function/t/8.2.x

II.3 ROUTING

II.3.a Fichier .routing.yml

- Ajouter le fichier d8_lille.routing.yml avec les paramètres suivants :

```
d8_lille.images_wall:
  path: '/images/wall'
  defaults:
    _controller:
'\Drupal\d8_lille\Controller\ImagesWallController::display'
    _title: 'Images Wall'
  requirements:
    _permission: 'access images wall'
```

Retrouver les paramètres de la route à partir de la documentation officielle:
https://www.drupal.org/docs/8/api/routing-system/structure-of-routes,

II.3.b Classe de controlleur

- Ajouter votre classe de controlleur pour la route ajoutée dans le fichier précédent

  Copier le contenu du fichier **web/core/modules/system/src/Controller/Http4xxController.php** en modifiant le namespace et le code de rendu.

III SERVICES

III.1.a Fichier .services.yml

- Ajouter un service **pixabay_requester :**

```
services:
  d8_lille.pixabay_requester:
    class: Drupal\d8_lille\Services\PixabayRequester
```

- Ce service ira charger la classe *Drupal\d8_lille\Services\PixabayRequester*

Si nécessaire, aidez-vous de la documentation officielle:
https://www.drupal.org/docs/8/api/services-and-dependency-injection/structure-of-a-service-file

III.1.b Injection de dépendance

- ***A partir des informations dans les slides :***

  Injecter la classe de service (elle ne fait rien à ce stade) dans votre controlleur ***ImagesWallController***

- Exécuter la route /images/wall, la page doit s'exécuter sans erreurs.

III.2 Guzzle

III.2.a HttpClient

Pour retrouver un service à partir de Drupal console :

```
drupal debug:container |grep Guzzle
```

- Injectez le service Guzzle dans votre service *PixabayRequester*.

```
private $httpClient;
public function __construct(Client $httpClient) {
  $this->httpClient = $httpClient;
}
```

III.2.b Requête avec Guzzle

Guzzle fonctionne dans Drupal 8 comme avec Symfony. Il dispose d'une documentation très bien écrite : http://docs.guzzlephp.org/en/stable/

- Ajouter un service Guzzle qui requête Pixabay, avec la méthode suivante:

```
public function getImages() {
  $url = 'https://pixabay.com/api/';
  $options = [
    'query' => [
      'key' => '10270854-a866a6ef4ccbf9830aa0ab761',
      'q' => 'fox',
    ],
  ];
  try {
    $response = $this->httpClient->request('GET', $url, $options);
    $code = $response->getStatusCode();
    if ($code == 200) {
      $body = $response->getBody()->getContents();
      return json_decode($body)->hits;
    }
  }
  catch (RequestException $e) {
    \Drupal::logger('d8_lille')->error('Pixabay request failed with
message %err', ['%err' => $e→getMessage()]);
```

Vérifier que la requête fonctionne avec Kint, dans votre controlleur appeler la méthode getImages() :

```
class ImagesWallController extends ControllerBase {

  private $pixabayRequester;

  public function __construct() {
    $this->pixabayRequester = \Drupal::service('d8_lille.pixabay_requester');
  }
  /**
   * The wall controller.
   */
  public function display() {
    kint($this→pixabayRequester→getImages());
exit;
  }
```