

A black and white aerial photograph of the Paris skyline at night. The Eiffel Tower is prominent in the center-left, its lights reflected in the Seine River below. The city lights of Paris are scattered across the horizon, with the Louvre and other landmarks visible. The sky above is filled with dramatic, swirling clouds.

DÉCOUVRIR DRUPAL

LE THEMING

QUI SUIS-JE ?



Une photo de moi.

FLORIS MORICEAU

- ▶ Développeur Front-End chez L'équipe.tech
(HTML / CSS / UI / UX / A11y)
- ▶ Secrétaire de l'association Drupal France
- ▶ Lead dev Front-End de drupal.fr
- ▶ Bientôt 10 ans de Front...

Sur les Internets mondiales :

Twitter : @FlorisMoriceau

Drupal : FlorisMoriceau

SOMMAIRE

- ▶ #1 - Anatomie d'un thème
- ▶ #2 - Créer son thème
- ▶ #3 - Ajouter des assets (images, fonts, css, js...)
- ▶ #4 - Les fichiers *template*
- ▶ #5 - TWIG
- ▶ #6 - Les Régions
- ▶ #7 - Rendu du contenu
- ▶ #8 - Preprocess
- ▶ #9 - Javascript
- ▶ #10 - Questions / Réponses (Peut-être, si on a le temps, et c'est pas gagné...)

 zurb_foundation.theme

▼  custom

▼  drupal_france

►  config

►  css

►  images

►  node_modules | library root

►  scss

►  templates

 .gitattributes

 .sass-lint.yml

 config.js

 drupal_france.info.yml

 drupal_france.libraries.yml

 drupal_france.theme

 favicon.ico

 gulpfile.js

 logo.svg

 package.json

 package-lock.json

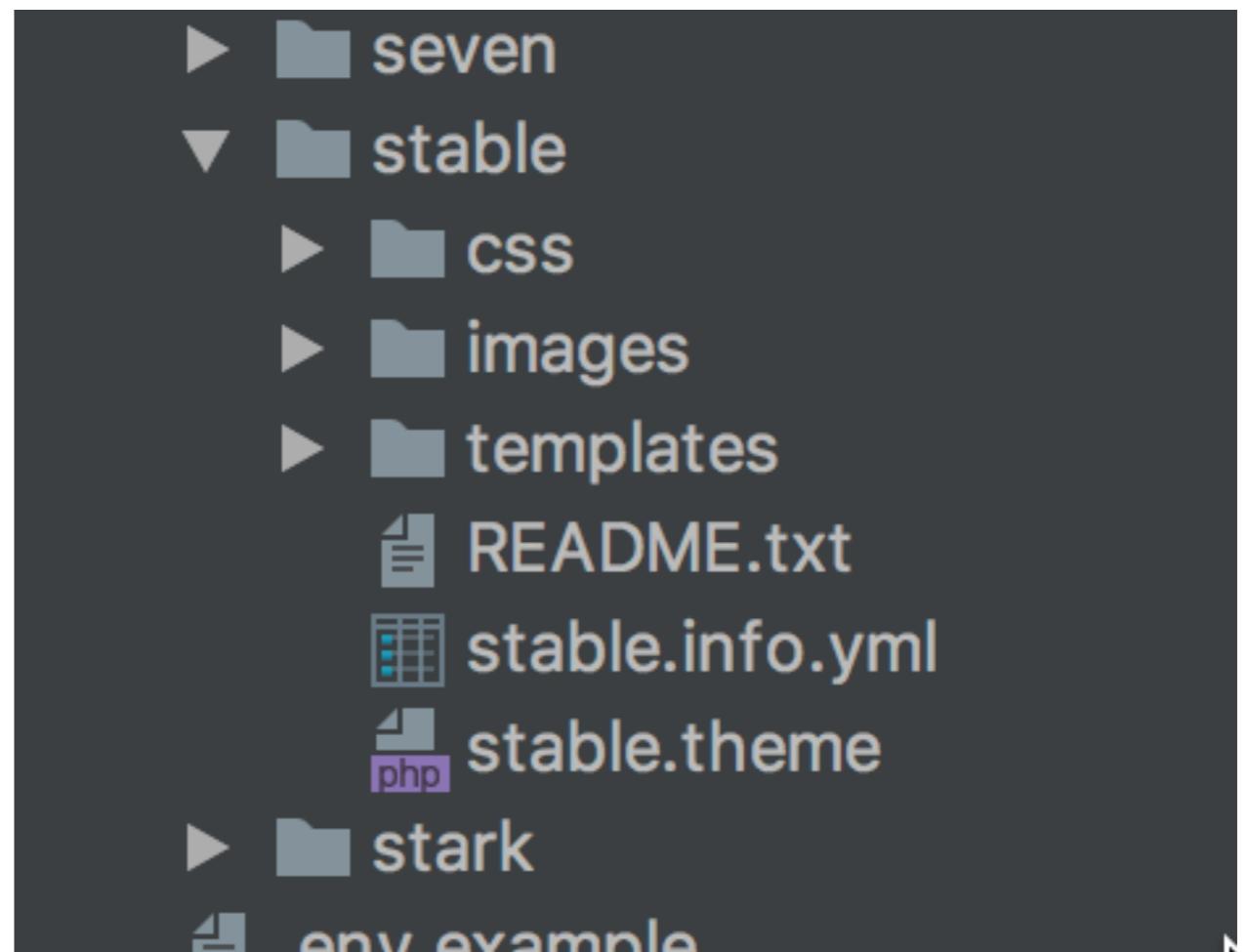
 translations

#1

ANATOMIE D'UN THÈME

LES DOSSIERS & FICHIERS

```
-THEMENAME.breakpoints.yml  
-THEMENAME.info.yml  
-THEMENAME.libraries.yml  
-THEMENAME.theme  
  
-config  
  |-install  
  |  |-THEMENAME.settings.yml  
  |-schema  
  |  |-THEMENAME.schema.yml  
  
-css  
  |-style.css  
  
-js  
  |-script.js  
  
-images  
  |-buttons.png  
  
-logo.svg  
  
-screenshot.png  
  
-favicon.ico  
  
-templates  
  |-maintenance-page.html.twig  
  |-node.html.twig
```



THEMENAME.INFO.YML

Informations du thème

Librairies chargées avec le thème

Régions du thème

```
1  name: Drupal France
2  description: "Theme inheriting from the Foundation base theme.<br>
3  type: theme
4  core: 8.x
5  package: 'Drupalfr'
6  base theme: zurb_foundation
7
8  libraries:
9    - 'drupal_france/global'
10
11 regions:
12   meta_header: 'Meta Header'
13   main_menu: 'Main Menu'
14   main_menu_extra: 'Main Menu Extra Elements'
15   header: Header
16   help: Help
17   highlighted: Highlighted
18   content: Content
19   sidebar_first: 'Left Sidebar'
20   sidebar_second: 'Right Sidebar'
21   footer_first: 'Footer first column'
22   footer_middle: 'Footer middle column'
23   footer_last: 'Footer last column'
24   left_off_canvas: 'Left off canvas'
25   right_off_canvas: 'Right off canvas'
```

CETTE RÉGION EST OBLIGATOIRE

Remarque : Lorsque vous travaillez avec ce fichier, un `drush cr` s'impose à chaque modification

LES FICHIERS .CSS

- ▶ Les fichiers **.css** contiennent les feuilles de styles du thème.
- ▶ C'est à l'auteur du thème de créer ses propres fichiers **.css** et de les référencer en les déclarant comme librairies dans le fichier **/THEMENAME.libraries.yml**.
- ▶ Les fichiers **.css** peuvent être nommés librement, mais il est recommandé de respecter les conventions en vigueur :
 - ▶ Mettre tous les fichiers **.css** dans un répertoire **css/**.
 - ▶ Utiliser **style.css** pour les styles généraux, **layout.css** pour les styles liés à la structure de page... (Méthodologie SMACSS)

LES FICHIERS .JS

- ▶ Les fichiers **.js** contiennent les scripts (javascript) du thème.
- ▶ C'est à l'auteur du thème de créer ses propres fichiers **.js** et de les référencer en les déclarant comme librairies dans le fichier **/THEMENAME.libraries.yml**.
- ▶ Les fichiers **.js** peuvent être nommés librement.
- ▶ Par défaut **aucune librairie javascript n'est chargée pour les utilisateurs anonymes**, pas même *jQuery*.

FICHIERS DE TEMPLATE

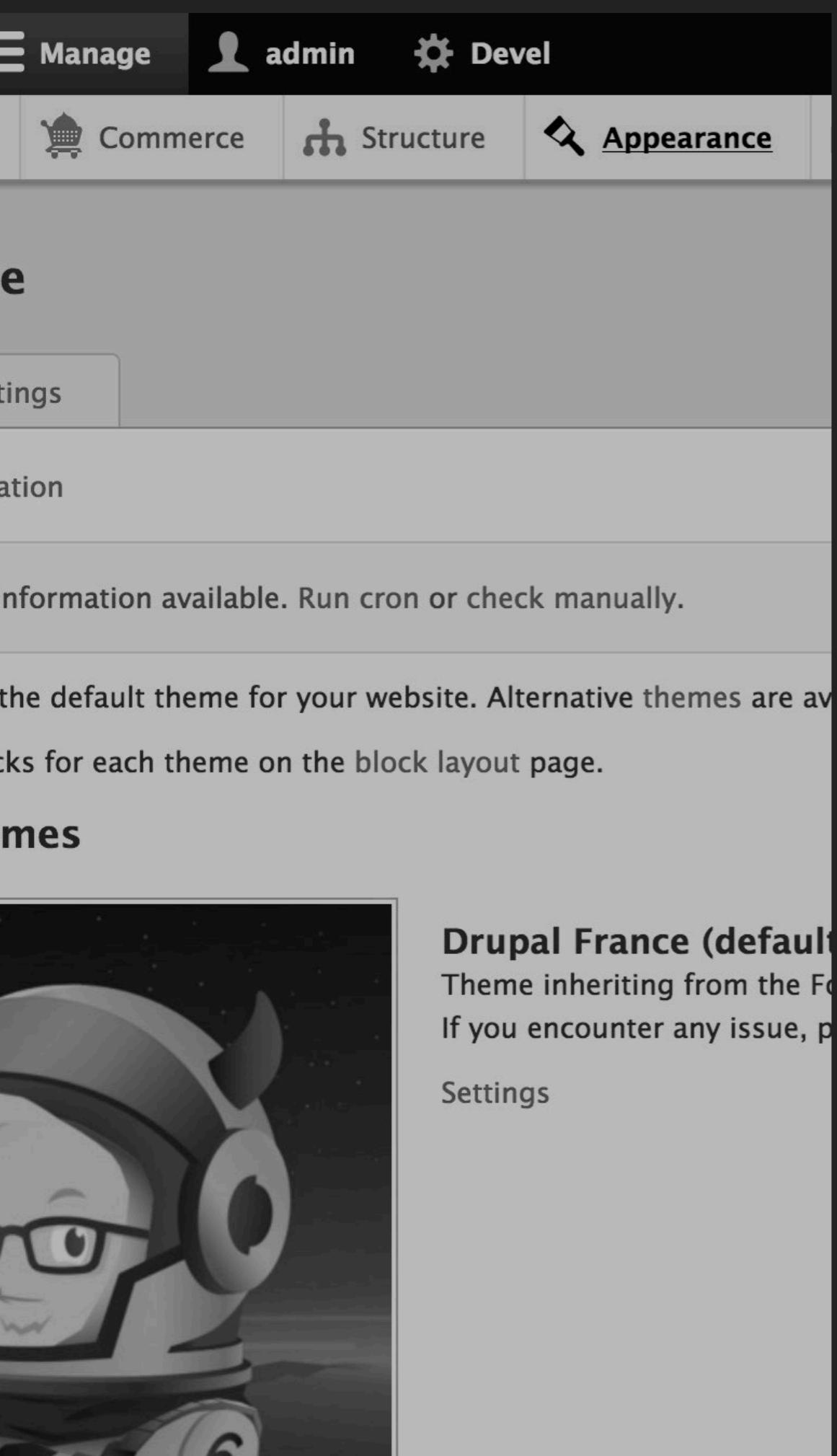
- ▶ Les fichiers avec l'extension **.html.twig** sont les **templates**.
- ▶ Un template contient un mélange de **balises HTML** et de code **Twig**.
- ▶ Tout thème Drupal contient généralement ces 3 templates de base :
 - ▶ **page.html.twig** : définit la structure générale de toutes les pages (emplacement des régions).
 - ▶ **node.html.twig** : définit la structure des noeuds.
 - ▶ **block.html.twig** : définit la structure des blocs.
- ▶ Par défaut, la template `page.html.twig` sert à afficher toutes les pages, `node.html.twig` sert à afficher tous les noeuds... Une modification dans un de ces templates impacte donc toutes les pages, tous les noeuds... Nous verrons plus tard comment créer des templates plus spécifiques.
- ▶ Ces fichiers sont à placer dans le répertoire **/templates**.

THEMENAME.THEME

- ▶ Le fichier **montheme.theme** permet d'ajouter du code PHP additionnel spécifique au thème.
- ▶ Ce fichier est facultatif, mais il doit être obligatoirement placé à la racine du thème s'il existe.
- ▶ Il contient des fonctions de **preprocess** que l'on souhaite étendre (voir plus loin), ainsi qu'un certain nombre de fonctions de hook.
- ▶ Ce fichier est automatiquement reconnu et chargé par Drupal.

FICHIERS IMAGE

- ▶ Drupal n'utilise pas de répertoire spécifique pour stocker les images d'un thème.
- ▶ La bonne pratique est de créer un sous-dossier **/images** (ou **/img...**) dans le répertoire du thème, mais on peut s'organiser autrement si nécessaire.
- ▶ Dans tous les cas, veiller à utiliser les bons chemins pour référencer les images.



#2

CRÉER SON THÈME (À PARTIR D'UN THÈME EXISTANT)

OPTIONS DE PERSONNALISATION

- ▶ Dans la section **Apparence** du back-office les personnalisations sont très limitées, on peut seulement :
 - ▶ Changer le logo.
 - ▶ Afficher/Masquer certains éléments comme :
 - ▶ Le Nom du site qui apparaît dans la région "en-tête".
 - ▶ Les portraits utilisateur.
 - ▶ Le slogan du site.
 - ▶ Changer l'icône de raccourcis.



Appearance settings

[List](#)[Settings](#)[Global settings](#)[Seven](#)[ZURB Foundation 6](#)[Drupal France](#)

[Home](#) » [Administration](#) » [Appearance](#)

Control default display settings for your site, across all themes. Use theme-specific settings to override these defaults.

▼ PAGE ELEMENT DISPLAY

- User pictures in posts
- User pictures in comments
- User verification status in comments
- Shortcut icon

▼ LOGO IMAGE

- Use the logo supplied by the theme

▼ FAVICON

Your shortcut icon, or favicon, is displayed in the address bar and bookmarks of most browsers.

- Use the favicon supplied by the theme

SI ON DÉSIRE PLUS D'OUTILS DE PERSONNALISATION, IL FAUT LES CODER.

ET DONC, CRÉER SON PROPRE THÈME.

POUR CRÉER SON THÈME, PLUSIEURS POSSIBILITÉS

▶ Partir de zéro

Possible mais fastidieux

(*Ne faites pas ça. Ou alors seulement sur un side project personnel qui n'a pas de deadline, pour la beauté du geste, et pour apprendre tous les rouages du templating*)

▶ Partir d'un thème de base

Un thème de base est un thème que l'on va étendre en héritant de tout son code. On dispose d'un système de thèmes parent/enfant. Exemples : Omega, Zen, Bootstrap... ZURB Foundation

▶ Partir d'un thème "prêt-à-l'emploi"

Par exemple un thème acheté sur themeforest.com.

L'idée ici est de limiter au maximum les personnalisations afin de gagner du temps.

Attention: Sortir des fonctionnalités offertes par ce genre de thème peut se révéler très coûteux.

MÉTHODE 1 : VIA UN THÈME PRÉSENT DANS LE CORE (BARTIK)

- ▶ On fait une **copie** du thème par défaut **Bartik** (c'est-à-dire du répertoire `/core/themes/bartik`).
- ▶ On **colle** le répertoire copié dans `/themes/custom` et on le renomme comme on le souhaite.
Attention cependant : Le thème doit avoir un « nom machine » composé exclusivement de caractères alphanumériques et de '`_`' (underscore)
- ▶ Ici, on choisira '**THEMENAME**'
- ▶ On renommez le fichier `/themes/custom/THEMENAME/bartik.info.yml` en `/themes/custom/THEMENAME/THEMENAME.info.yml` (on n'oublie pas d'adapter également le contenu du fichier .info.yml).
- ▶ On renomme tous les fichiers dont le nom comporte bartik par THEMENAME.
- ▶ Dans tous les fichiers on recherche les occurrences de la chaîne 'bartik' et on les remplace par 'THEMENAME'.
- ▶ **On vide TOUS les caches !**
- ▶ Enfin, on va dans Admin > Apparence, on active et défini par défaut notre nouveau thème THEMENAME.
- ▶ On vérifie qu'il n'y a pas d'erreur en visitant par exemple la page d'accueil.

MÉTHODE 2 : VIA UN THÈME CONTRIB AYANT UN STARTER (ZF)

- ▶ Certains thèmes contrib proposent un « STARTER KIT » comme par exemple Bootstrap ou ZURB Foundation
- ▶ Dossier prêt à être copié collé pour faire un thème enfant héritant de ce dernier
- ▶ On fait une **copie** du STARTER KIT.
- ▶ On **colle** le répertoire copié dans `/themes/custom` et on le renomme comme on le souhaite.
- ▶ Etc.

REMARQUE : DANS LES DEUX CAS, NE PAS OUBLIER DE PRÉCISER LE 'BASE THEME' DANS LE .INFO.YML

REMARQUE : LE THÈME PARENT DOIT ÊTRE ACTIVÉ SI VOUS VOULEZ ACTIVER UN DE SES THÈMES ENFANT.

JOB'S DONE!

Warcraft Human peon

MAIS. . .
REVENONS SUR NOTRE
FICHIER .INFO.YML

THEMENAME.INFO.YML

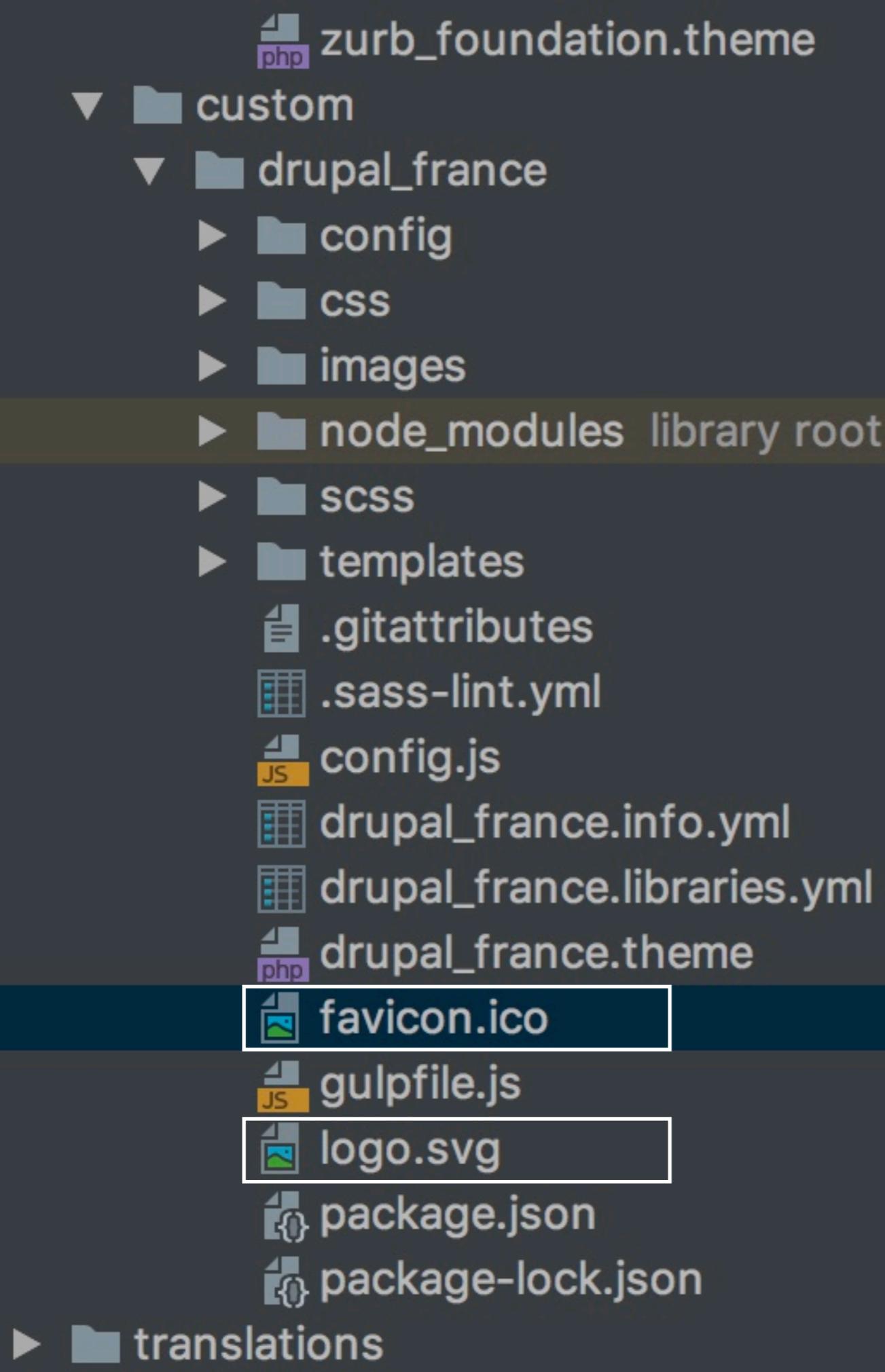
Informations du thème

Librairies chargées avec le thème

Régions du thème

```
1  name: Drupal France
2  description: "Theme inheriting from the Foundation base theme.<br>
3  type: theme
4  core: 8.x
5  package: 'Drupalfr'
6  base theme: zurb_foundation
7
8  libraries:
9    - 'drupal_france/global'
10
11 regions:
12   meta_header: 'Meta Header'
13   main_menu: 'Main Menu'
14   main_menu_extra: 'Main Menu Extra Elements'
15   header: Header
16   help: Help
17   highlighted: Highlighted
18   content: Content
19   sidebar_first: 'Left Sidebar'
20   sidebar_second: 'Right Sidebar'
21   footer_first: 'Footer first column'
22   footer_middle: 'Footer middle column'
23   footer_last: 'Footer last column'
24   left_off_canvas: 'Left off canvas'
25   right_off_canvas: 'Right off canvas'
26   page_top: 'Page top'
27   page_bottom: 'Page bottom'
28
```

ET SUR LES FICHIERS DANS NOTRE
DOSSIER
'THEMENAME'



CONCRÈTEMENT

The screenshot shows the Drupal 8 administration interface. The top navigation bar includes links for Back to site, Manage, admin, and Devel. Below this is a secondary navigation bar with icons for Content, Commerce, Structure, Appearance (which is underlined, indicating it's the active section), Extend, Configuration, People, and Statistics.

The main content area is titled "Appearance" and contains two tabs: "List" and "Settings". The "List" tab is selected. Below the tabs, the breadcrumb navigation shows Home » Administration.

A message box displays a warning: "⚠ No update information available. Run cron or check manually.".

The main content area is titled "Set and configure the default theme for your website. Alternative themes are available." It shows a list of installed themes:

- NAME**: **Drupal France (default theme)**
Theme inheriting from the Foundation base theme.
If you encounter any issue, please send an email to floris.moriceau@komuneco.fr
- DESCRIPTION**: [Settings](#)

Annotations highlight specific parts of the interface:

- A blue box labeled "SCREENSHOT.PNG" has a blue arrow pointing to the thumbnail image of the "Drupal France" theme.
- A blue box labeled "NAME" has a blue arrow pointing to the theme name "Drupal France".
- A blue box labeled "DESCRIPTION" has a blue arrow pointing to the "Settings" link under the theme description.

CONCRÈTEMENT





DOCUMENTATION OFFICIELLE

<https://www.drupal.org/docs/8/theming-drupal-8/defining-a-theme-with-an-infoyml-file>

LES POINTS À RETENIR :

Dans un fichier .yml

- ▶ Tabs are NOT allowed. Use spaces ONLY.
- ▶ Properties and lists MUST be indented by two (2) spaces.

Mais surtout :

- ▶ libraries-override (optional)
- ▶ **base theme (recommended)**
- ▶ hidden (optional)
- ▶ engine (optional)
- ▶ logo (optional)
- ▶ screenshot (optional)
- ▶ regions (optional)
- ▶ regions_hidden (optional)
- ▶ features (optional)
- ▶ **stylesheets-remove (deprecated)**
- ▶ ckeditor_stylesheets (optional)
- ▶ **name (required)**
- ▶ **type (required)**
- ▶ description (optional)
- ▶ package (optional)
- ▶ **core (required)**
- ▶ php (optional)
- ▶ version (optional)
- ▶ libraries (optional)

VERSION: VERSION

css:

base:

```
css/foundation.min.css:  
css/motion-ui.min.css: {  
css/zurb_foundation.css:
```

js:

```
js/foundation.min.js: {}  
js/foundation_init.js: {}  
js/top_bar_active.js: {}  
js/motion-ui.min.js: {}
```

dependencies:

- core/drupal
- core/jquery
- core/jquery.once

status_in_reveal:

version: VERSION

js:

```
js/status_in_reveal.js: {}
```

dependencies:

- core/drupal
- core/jquery
- core/jquery.once

#3

AJOUTER DES ASSETS (CSS/JS)

CHARGER DES FICHIERS CSS & JS

Ce que dit la documentation :

<https://www.drupal.org/docs/8/theming/adding-stylesheets-css-and-javascript-js-to-a-drupal-8-theme>

1. Save the CSS or JS to a file using the [proper naming conventions and file structure](#).
2. [Define a "library", which registers these CSS/JS files with your theme.](#)
3. "Attach" the library to *all* pages, to [specific Twig templates](#), or [target specific pages via a render element in a preprocess function](#).

EN PRATIQUE

- ▶ L'ajout de fichier de **CSS** et de **JS** se fait via un système de librairie (bibliothèque).
- ▶ Il faut tout d'abord déclarer une librairie (ou en utiliser une existante) dans le fichier **THEMENAME.libraries.yml**, puis demander au système de la charger.
Ceci peut être fait depuis le fichier **THEMENAME.info.yml**.

```
name: ZURB Foundation 6
description: The most advanced responsive theme for Drupal 8
# core: 8.x
type: theme

# Image appears on admin/appearance
screenshot: images/screenshot.jpg

libraries:
- zurb_foundation/global

# Uncompressed, uncomment for development
# - zurb_foundation/development

# Regions are output in the page.html.twig file
regions:
header: Header
meta_header: 'Meta header'
help: Help
page_top: 'Page top'
page_bottom: 'Page bottom'
highlighted: Highlighted
content: Content
sidebar_first: 'Left Sidebar'
sidebar_second: 'Right Sidebar'
footer_first: 'Footer first column'
footer_middle: 'Footer middle column'
footer_last: 'Footer last column'
```

```
global:
version: VERSION
css:
base:
css/foundation.min.css: {}
css/motion-ui.min.css: {}
css/zurb_foundation.css: {}
js:
js/foundation.min.js: {}
js/foundation_init.js: {}
js/top_bar_active.js: {}
js/motion-ui.min.js: {}
dependencies:
- core/drupal
- core/jquery
- core/jquery.once
```

```
status_in_reveal:
version: VERSION
js:
js/status_in_reveal.js: {}
dependencies:
- core/drupal
- core/jquery
- core/jquery.once

alert_close:
version: VERSION
js:
js/alert_close.js: {}
dependencies:
- core/drupal
- core/jquery
- core/jquery.once
```

CHARGER UNE LIBRAIRIE

- ▶ Lorsqu'une librairie est appelée depuis le fichier **THEMENAME.info.yml**, alors elle est **chargée sur toutes les pages** dès lors que le thème est utilisé.
- ▶ Il est possible de charger **sous condition** une bibliothèque. Pour cela plusieurs techniques sont disponibles :
 - ▶ depuis un fichier de template.
 - ▶ en utilisant le **hook_page_attachments_alter()**.
 - ▶ en utilisant une fonction de **preprocess**.

CHARGER UNE LIBRAIRIE

- ▶ Directement depuis un fichier de template directement, par exemple dans le fichier gérant l'affichage des blocs :

```
{% attach_library('drupal_france/blocks') %}
```

CHARGER UNE LIBRAIRIE

- ▶ Chargement d'une librairie depuis le fichier
`/themes/THEMENAME/THEMENAME.theme`
en utilisant la fonction de hook
`hook_page_attachments_alter()`.

```
/**  
 * Implements hook_page_attachments_alter().  
 */  
function drupal_france_page_attachments_alter(array &$attachments) {  
  $attachments['#attached']['library'][] = 'drupal_france/global';  
}
```

CHARGER UNE LIBRAIRIE

- ▶ Chargement d'une librairie depuis le fichier
/themes/THEMENAME/THEMENAME.theme
en utilisant une fonction de **preprocess**.

```
/**  
 * Implements hook_preprocess_HOOK() for page templates.  
 */  
function drupal_france_preprocess_page(array &$amp;variables) {  
  $variables['#attached']['library'][] = 'drupal_france/global';  
}
```

```
↳ drupalfr-newsletter-subscription-t  
  ↳ node  
    ↳ node.html.twig  
    ↳ node--event--full.html.twig  
    ↳ node--event--teaser.html.twig  
    ↳ node--local-group--full.html.twig  
    ↳ node--local-group--teaser.html.twig  
    ↳ node--news--full.html.twig  
    ↳ node--news--listing.html.twig  
    ↳ node--news--teaser.html.twig  
    ↳ node--organization--teaser.html.twig  
    ↳ node--planet--full.html.twig  
    ↳ node--planet--teaser.html.twig  
    ↳ node--release--listing.html.twig  
    ↳ node--search-result.html.twig  
    ↳ node--showcase--teaser.html.twig  
    ↳ node--view--forum--block-last-to  
  ↳ paragraph  
    ↳ paragraph.html.twig  
    ↳ paragraph--banner--default.html.twig  
    ↳ paragraph--faq.html.twig  
    ↳ paragraph--faq-element.html.twig  
    ↳ paragraph--html--default.html.twig  
    ↳ paragraph--layout-twocol--default.html.twig  
    ↳ paragraph--title-text-media--default.html.twig  
    ↳ paragraph--trombinoscope--default.html.twig  
  ↳ system  
    ↳ maintenance-page.html.twig  
    ↳ page.html.twig  
    ↳ region.html.twig  
    ↳ taxonomy-term--tags.html.twig  
  ↳ user  
    ↳ user--compact.html.twig  
    ↳ user--full.html.twig  
  ↳ views  
    ↳ views-infinite-scroll-pager.html.twig
```

```
64   ↳ <div c  
65     ↳ <div  
66       ↳ {  
67         ↳ %  
68       ↳ {  
69         ↳ %  
70       ↳ {  
71         ↳ %  
72       ↳ {  
73         ↳ %  
74       ↳ {  
75         ↳ %  
76       ↳ {  
77         ↳ %  
78       ↳ {  
79         ↳ %  
80       ↳ {  
81         ↳ %  
82       ↳ {  
83         ↳ %  
84       ↳ {  
85         ↳ %  
86       ↳ <d  
87         ↳ {  
88         ↳ %  
89       ↳ {  
90         ↳ %  
91       ↳ {  
92         ↳ %  
93       ↳ {  
94         ↳ %  
95       ↳ {  
96         ↳ %  
97       ↳ {  
98         ↳ %  
99       ↳ {  
100        ↳ %  
101        ↳ %  
102        ↳ %  
103        ↳ %
```

#4

LES FICHIERS TEMPLATES

À QUOI ÇA SERT ?

- ▶ Les **templates** constituent la structure des pages du sites. En fait, de chacun des éléments qui la composent.
- ▶ Ce sont des fichiers **html** (balises **<div>**, **<h1>**, **<a>**...) avec du code **Twig** permettant d'afficher les éléments (sous forme de variables) créés par Drupal.
- ▶ Chaque type d'élément du site possède son propre template.

#4 LES TEMPLATES

PAR EXEMPLE (C'EST DU VRAI CODE)

```
<div class="l-main row expanded">
  <main id="main" class="{{ main_grid }}" role="main">
    {%
      if page.highlighted %
        {%
          spaceless %
        }
        {{ page.highlighted }}
        {%
          endspaceless %
        }
      {%
        endif %
      }
      <a id="main-content"></a>
      {%
        if breadcrumb %
          {{ breadcrumb }}
        {%
          endif %
      }
      <section>
        {{ page.content }}
      </section>
    </main>
    {%
      if page.sidebar_first %
        <div id="sidebar-first" class="{{ sidebar_first_grid }}" column sidebar>
          {{ page.sidebar_first }}
        </div>
    {%
      endif %
    }
    {%
      if page.sidebar_second %
        <div id="sidebar-second" class="{{ sidebar_sec_grid }}" column sidebar>
          {{ page.sidebar_second }}
        </div>
    {%
      endif %
    }
  </div>
```

IMBRICATION DES TEMPLATES

- ▶ Imaginons l'appel d'une page de billet de blog
- ▶ Et maintenant, imaginons le parcours d'imbrication de templates pour depuis la balise `<html>` jusqu'à la `<div>` contenant un commentaire sous ce billet de blog.

#4 LES TEMPLATES

FROM <HTML> TO <DIV CLASS='COMMENT'>

html.html.twig

page.html.twig

region.html.twig

block.html.twig

node.html.twig

node.html.twig

field.html.twig

comment-wrapper.html.twig

comment.html.twig

LES TEMPLATES DÉRIVÉES

- ▶ Les **templates dérivées** sont des templates utilisées dans un **contexte précis**.
- ▶ C'est le **nom** de la template qui indique à Drupal quand l'utiliser.
- ▶ Toutes ces templates sont optionnelles et reconnues automatiquement par Drupal en fonction de leurs noms.
- ▶ Si des templates ne sont pas présentes dans le thème, alors Drupal utilise ceux par défaut (/core/themes/*).
- ▶ On parle de "template dérivée", de "suggestion de template" ou de "**templates suggérées**".
- ▶ **Il faut vider le cache de Drupal lorsque l'on ajoute des templates au thème.**
- ▶ Documentation :
<https://www.drupal.org/docs/8/theming/twig/twig-template-naming-conventions>

EXEMPLE DE SUGGESTIONS DE TEMPLATES

- ▶ **Template de page page.html.twig :**
`page--chemin--interne.html.twig` sera utilisé lorsque l'on appelle **chemin/interne**
 - ▶ Ex. : `page--front.html.twig` cible la page d'accueil.
 - ▶ Ex. : `page--chemin.html.twig` cible toutes les pages dont le chemin commence par chemin.
 - ▶ Ex. : `page--node--240.html.twig` cible la page du noeud portant l'id 240.
- ▶ **Template de noeud node.html.twig :**
`node--type-de-noeud.html.twig` cible les noeuds de type **type-de-noeud**.
 - ▶ Ex. : `node--article.html.twig` cible tous les noeuds de type article.

TROUVER LES SUGGESTIONS

- ▶ Dans le fichier `www/sites/development.services.yml`

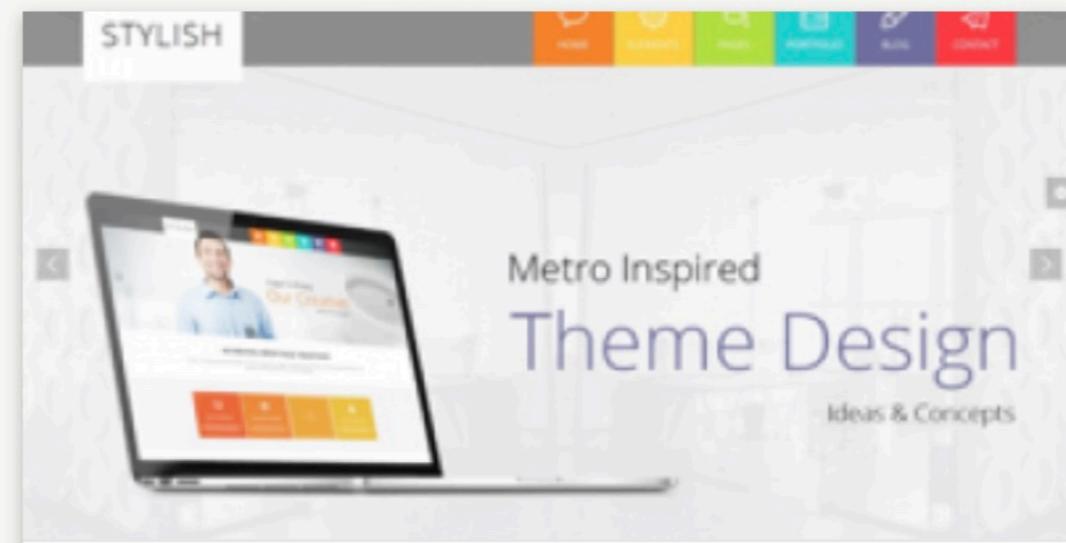
```
# Local development services.

parameters:
  http.response.debug_cacheability_headers: true
  twig.config:
    debug: true
services:
  cache.backend.null:
    class: Drupal\Core\Cache\NullBackendFactory
```



Test news

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...



Test d'une news avec un titre super long, mais alors vraiment très long

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...



Test news

Lorem ipsum dolor elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...

```
s Console Sources Network Performance Memory Application Security Audits axe HTML validator
<!-- BEGIN OUTPUT from 'themes/custom/drupal_trance/templates/views/views--view--unformatted--news--block--homepage.html.twig' -->


::before
  <div class="views-row small-12 medium-4 column">
    <!-- THEME DEBUG -->
    <!-- THEME HOOK: 'node' -->
    <!-- FILE NAME SUGGESTIONS:
      * node--view--news--block--homepage.html.twig
      * node--view--news.html.twig
      * node--59--teaser.html.twig
      * node--59.html.twig
      x node--news--teaser.html.twig
      * node--news.html.twig
      * node--teaser.html.twig
      * node.html.twig
    -->
    <!-- BEGIN OUTPUT from 'themes/custom/drupal_france/templates/node/node--news--teaser.html.twig' -->
    <article role="article" about="/association/actualites/test-news" class="news teaser clearfix news-element" data-equal-height="439px;">


```

```
div class="off-canvas-wrapper">
<div class="inner-wrap off-canvas-wrapper-inner"
  id="inner-wrap"
  data-off-canvas-wrapper>
  {% if page.left_off_canvas %}
    <aside id="left-off-canvas-menu"
      class="off-canvas left-off-canvas-menu"
      role="complementary"
      data-off-canvas>
      {{ page.left_off_canvas }}
    </aside>
  {% endif %}

  {% if page.right_off_canvas %}
    <aside id="right-off-canvas-menu"
      class="off-canvas right-off-canvas-men
      role="complementary"
      data-off-canvas>
      {{ page.right_off_canvas }}
    </aside>
  {% endif %}

<div class="off-canvas-content meta_header"
  data-off-canvas-content>

  {% if page.meta_header %}
    {% if meta_header_grid %}
      <div class="row expanded"><div class="row
    {% endif %}

    {{ page.meta_header }}

    {% if meta_header_grid %}
      </div></div>
    {% endif %}
  {% endif %}

<header class="row expanded l-header" role="b
  <div class="row">
    {# Menu toggle button #}

```

#5

TWIG

UN MOTEUR DE TEMPLATE

- ▶ **Twig** est capable d'afficher le contenu de n'importe quel type de variable (chaine, tableau ou objet).
- ▶ Il est possible de faire des **boucles**, des **conditions**...
- ▶ Chaque variable peut être **filtrée** avant d'être affichée.
- ▶ Les chaines peuvent être rendues traduites (la traduction est alors faite via le back-office de Drupal).

LES BASES

- ▶ Affiche le contenu de la variable var : `{{ var }}`.
- ▶ Affiche le contenu de la propriété title de la variable node : `{{ node.title }}`.
- ▶ Conditions : `{% if var %} {% else %} {% endif %}`
- ▶ Commentaires : `{# commentaire #}`
- ▶ Traduction : `{% trans %} chaine {% endtrans %}`.
- ▶ Fonctions spécifiques :
 - ▶ `{{ url('route_name') }}`.
 - ▶ `{{ path('entity.node.canonical', { 'node': node.id }) }}`.
- ▶ Les modules [Twig tweak](#) ou encore [Bamboo Twig](#) proposent encore davantage de fonctions.

LES FILTRES

- ▶ Date : {{ date|format_date('medium') }}
- ▶ Chaine :
 - ▶ Echappement : {{ content }} (par défaut)
 - ▶ Interprétation du contenu de la variable : {{ texte|raw }} (**attention !**)
 - ▶ Placeholder : {{ text|placeholder }}
- ▶ Exclusion : {{ content|without('links') }}
- ▶ {{ string|lower }} (upper également)
- ▶ {{ class_name|clean_class }}
- ▶ {{ id_name|clean_id }}
- ▶ Documentation :
<https://www.drupal.org/docs/8/theming/twig/filters-modifying-variables-in-twig-templates>
<https://twig.symfony.com/doc/2.x/>

CONCRÈTEMENT

```
{% if page.footer_first or page.footer_middle or page.footer_last %}
<div class="l-footer row expanded">
  <footer class="row footer medium-up-2 large-up-3">
    {% if page.footer_first %}
      <div id="footer-first" class="column">
        {{ page.footer_first }}
      </div>
    {% endif %}
    {% if page.footer_middle %}
      <div id="footer-middle" class="column">
        {{ page.footer_middle }}
      </div>
    {% endif %}
    {% if page.footer_last %}
      <div id="footer-last" class="column">
        {{ page.footer_last }}
      </div>
    {% endif %}
  </footer>
</div>
{% endif %}
<div class="bottom-bar panel hidden">
  <div class="row">
    <div class="large-12 column">
      &copy; {{ "now"|date('Y') }} {{ site_name }} {{ 'All rights reserved.'|t }}
    </div>
  </div>
</div>
```

```
div class="off-canvas-wrapper">
<div class="inner-wrap off-canvas-wrapper-inner"
  id="inner-wrap"
  data-off-canvas-wrapper>
  {% if page.left_off_canvas %}
    <aside id="left-off-canvas-menu"
      class="off-canvas left-off-canvas-menu"
      role="complementary"
      data-off-canvas>
      {{ page.left_off_canvas }}
    </aside>
  {% endif %}

  {% if page.right_off_canvas %}
    <aside id="right-off-canvas-menu"
      class="off-canvas right-off-canvas-men
      role="complementary"
      data-off-canvas>
      {{ page.right_off_canvas }}
    </aside>
  {% endif %}

<div class="off-canvas-content meta_header"
  data-off-canvas-content>

  {% if page.meta_header %}
    {% if meta_header_grid %}
      <div class="row expanded"><div class="row
    {% endif %}

    {{ page.meta_header }}

    {% if meta_header_grid %}
      </div></div>
    {% endif %}
  {% endif %}

<header class="row expanded l-header" role="b
  <div class="row">
    {# Menu toggle button #}

```

#6

LES RÉGIONS

STRUCTURE DE LA PAGE

► [/admin/structure/block > Demonstrate block regions](/admin/structure/block)



DÉFINITION ET RENDU DES RÉGIONS

- ▶ Les régions doivent être définies dans le fichier
THEMENAME.info.yml
- ▶ Elles doivent être appelées obligatoirement dans
page.html.twig

THEMENAME.INFO.YML

```
regions:
  header: Header
  primary_menu: 'Primary menu'
  secondary_menu: 'Secondary menu'
  page_top: 'Page top'
  page_bottom: 'Page bottom'
  highlighted: Highlighted
  featured_top: 'Featured top'
```

PAGE.HTML.TWIG

```
{% if page.highlighted %}
  <div class="highlighted">
    <aside class="layout-container section clearfix" role="complementary">
      {{ page.highlighted }}
    </aside>
  </div>
{% endif %}
```

nce.theme

node--news--teaser.html.twig

```
/*...#}

set classes = [
  node.bundle|clean_class,
  node.isPromoted() ? 'is-promoted',
  node.isSticky() ? 'is-sticky',
  not node.isPublished() ? 'is-unpubl
  view_mode ? view_mode|clean_class,
  'clearfix',
  'news-element',
```

```
article{{ attributes.addClass(classes)
```

```
<div class="news-element--feature-ima
  {{ content.field_media_teaser }}>
</div>
```

```
  {{ title_prefix }}
  <h3{{ title_attributes.addClass('news
    <a href="{{ url }}" rel="bookmark">
  </h3>
  {{ title_suffix }}
```

```
<div{{ content_attributes.addClass('c
  {{ content.body }}>
</div>
```

```
articles>
```

#7

LE RENDU DU CONTENU

L'AFFICHAGE DES CHAMPS

- ▶ Pour chaque type de contenu on peut régler l'affichage des champs sur [admin/structure/types/manage/content-type/display](#) où **content-type** est le nom machine du type de contenu.
- ▶ Il est possible de modifier l'ordre des champs, l'apparence des étiquettes, le format du champ...
- ▶ Les réglages disponibles varient en fonction du type de champ :
 - ▶ Les champs **Texte** peuvent être coupés.
 - ▶ Les champs **Image** peuvent être affichés sous forme d'image, d'URL, de fichier...
 - ▶ Les champs **Date** peuvent utiliser un format différent.
- ▶ Ces réglages peuvent être **differents** selon le **mode d'affichage** (modes "Par défaut" et "Accroche"). => Cela permet par exemple de masquer un champ en mode "Accroche" et de l'afficher en mode "Par défaut" (noeud intégral).

L'AFFICHAGE DES IMAGES

- ▶ Les images transmises par les utilisateurs ne sont **jamais** aux bonnes dimensions.
- ▶ Le module **Image** (core) va nous permettre de prédéfinir des dimensions, sous forme de **styles d'image**, par exemple :
 - ▶ Vignette : 70px x 110px (mise à l'échelle).
 - ▶ Normal : 200px x 300px (mise à l'échelle et recadrage).
- ▶ Ces dimensions pourront ensuite être appliquées automatiquement aux champs **Image** depuis l'écran "gérer l'affichage" vu précédemment.

#7 RENDU DU CONTENU

STYLE D'IMAGE

Edit style *Rectangulaire (400x200)*

Edit

Devel

Home » Administration » Configuration » Media » Image styles

Preview

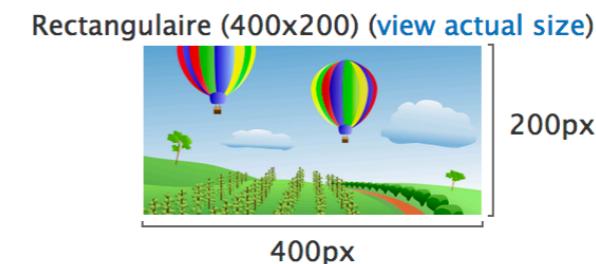
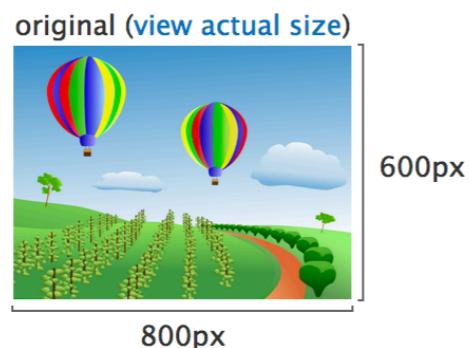


Image style name *

Rectangulaire (400x200)

Machine name: rect_400_200 [[Edit](#)]

[Show row weights](#)

EFFECT	ACTIONS
❖ Focal Point Scale and Crop 400x200	Edit ▾
❖ Select a new effect ▾ Add	

[Save](#)

[Delete](#)

STYLE D'IMAGE

Les actus de l'asso



Test news

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...



Test d'une news avec un titre super long, mais alors vraiment très long

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...



Test news

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores...

[TOUTES LES ACTUALITÉS](#)

ALLER PLUS LOIN

- ▶ La gestion et la manipulation des champs est cependant **limitée** dans l'interface d'administration.
- ▶ On est parfois **obligés** de **modifier** le **code** du template correspondant à notre champ ou type de contenu.
- ▶ Les réglages de l'admin sont toujours pris en compte. Par exemple, les styles d'image choisis seront ceux utilisés même si l'on modifie le template.

OÙ SONT LES CHAMPS ?

- ▶ Dans le fichier de template des noeuds (`node.html.twig`), on n'a pas directement accès aux champs.
- ▶ C'est la **variable content** qui contient tous les champs que l'on affiche, et en particulier ceux ajoutés via **Field**.
- ▶ Comment faire pour récupérer les champs un par un afin de les présenter différemment ?

LA VARIABLE 'CONTENT'

- ▶ Tout le contenu d'un noeud est affiché grâce à la ligne :
`{{ content|without('comment', 'links') }}`
- ▶ Pour connaître le contenu de content on utilise la fonction **kint()** :
`{{ kint(content) }}`
- ▶ On découvre alors l'ensemble des champs et leurs propriétés qui sont affichées en haut de page sous forme de **var_dump()**.
- ▶ On peut afficher séparément chacun d'entre eux avec le code suivant :
`{{ content.nom_du_champ }}`
`{{ node.createdtime }}`

#7 RENDU DU CONTENU

NODE DE TYPE 'NEWS' ET FORMAT 'TEASER'

The screenshot shows a code editor interface with a sidebar labeled "Project" containing a tree view of files. The main area displays a Twig template file named "node--news--teaser.html.twig". The code defines a class variable "classes" based on node properties like bundle, promoted status, sticky status, and publish status. It then uses this class variable in an "article" element's "addClass" method. The template also includes logic for displaying media teasers, titles, URLs, and content bodies.

```
1  {#/**...#}
68 {%
69   set classes = [
70     node.bundle|clean_class,
71     node.isPromoted() ? 'is-promoted',
72     node.isSticky() ? 'is-sticky',
73     not node.isPublished() ? 'is-unpublished',
74     view_mode ? view_mode|clean_class,
75     'clearfix',
76     'news-element',
77   ]
78 %}
79 <article{{ attributes.addClass(classes).setAttribute('data-equalizer-watch', true) }}>
80
81   <div class="news-element--feature-image">
82     {{ content.field_media_teaser }}
83   </div>
84
85   {{ title_prefix }}
86   <h3{{ title_attributes.addClass('news-element--title') }}>
87     <a href="{{ url }}" rel="bookmark">{{ label }}</a>
88   </h3>
89   {{ title_suffix }}
90
91   <div{{ content_attributes.addClass('content', 'news-element--content') }}>
92     {{ content.body }}
93   </div>
94
95 </article>
96
```

```
nce.theme × node--news--teaser.html.twig ×
/****#}

set classes = [
  node.bundle|clean_class,
  node.isPromoted() ? 'is-promoted',
  node.isSticky() ? 'is-sticky',
  not node.isPublished() ? 'is-unpubl
  view_mode ? view_mode|clean_class,
  'clearfix',
  'news-element',
]

article{{ attributes.addClass(classes)

<div class="news-element--feature-im
  {{ content.field_media_teaser }}>
</div>

{{ title_prefix }}
<h3{{ title_attributes.addClass('news
  <a href="{{ url }}" rel="bookmark">
</h3>
{{ title_suffix }}

<div{{ content_attributes.addClass('c
  {{ content.body }}>
</div>

articles>
```

#8

PREPROCESS

FONCTIONS DE PREPROCESS

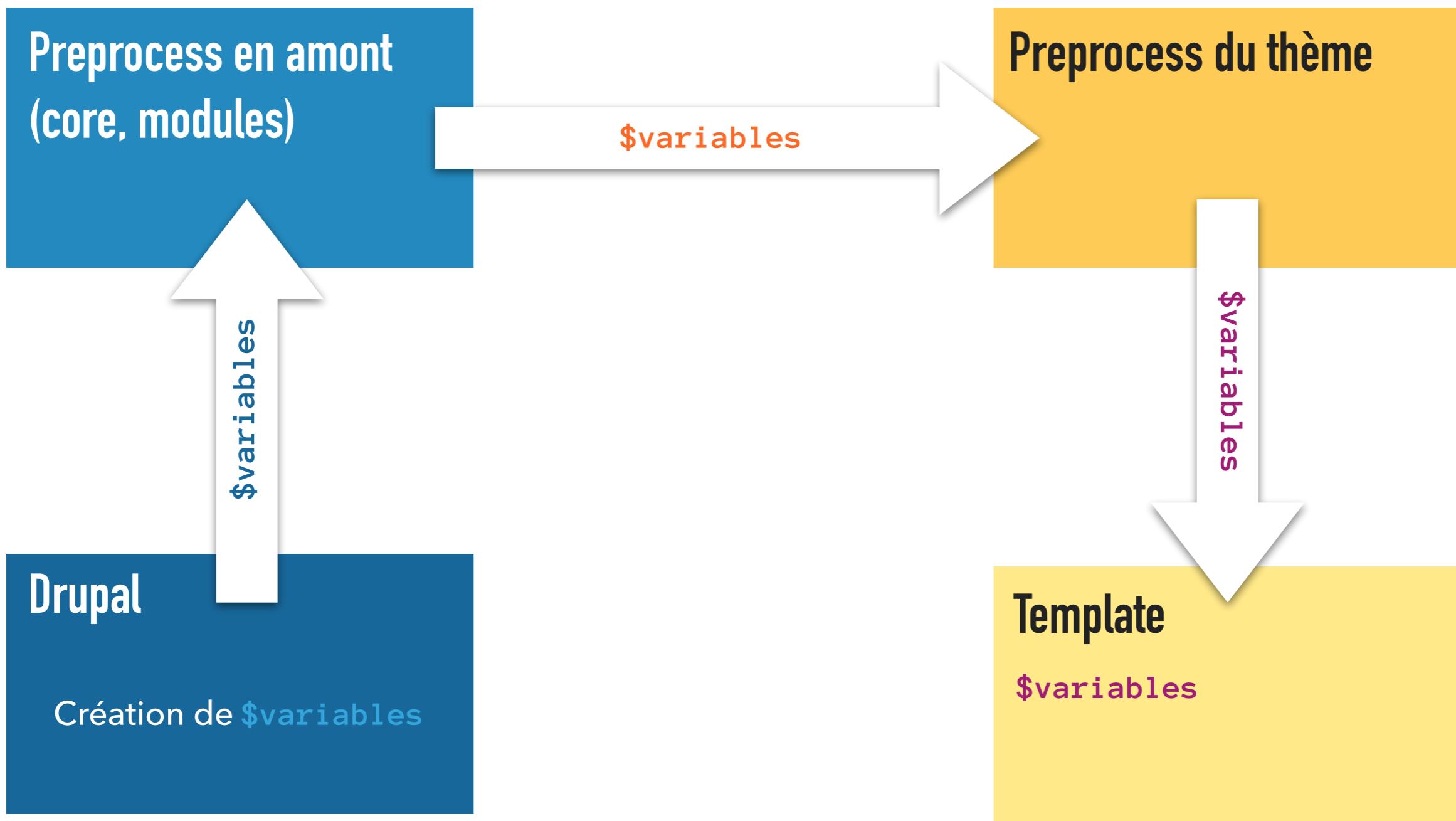
- ▶ Pour chaque template on dispose de **variables** que l'on affiche avec le formatage voulu.
- ▶ Ces **variables** sont créées en amont des **templates** dans lesquelles elles sont utilisées.
- ▶ On dispose de **fonctions** dites de **preprocess** qui permettent de fabriquer les variables qui sont utilisées dans les **templates**.
- ▶ **Il est possible d'étendre ces fonctions pour modifier les variables envoyées aux templates ou en créer de nouvelles.**
- ▶ Pour ajouter une fonction de preprocess il faut la **nommer comme suit :**
THEMENAME_preprocess_HOOK().
Par exemple **drupal_france_preprocess_block()**.

FONCTIONS DE PREPROCESS

- ▶ Pour chaque template on dispose de **variables** que l'on affiche avec le formatage voulu.
- ▶ Ces **variables** sont créées en amont des **templates** dans lesquelles elles sont utilisées.
- ▶ On dispose de **fonctions** dites de **preprocess** qui permettent de fabriquer les variables qui sont utilisées dans les **templates**.
- ▶ **Il est possible d'étendre ces fonctions pour modifier les variables envoyées aux templates ou en créer de nouvelles.**
- ▶ Pour ajouter une fonction de preprocess il faut la **nommer comme suit :**
THEMENAME_preprocess_HOOK().
Par exemple **drupal_france_preprocess_block()**.

LA VARIABLE '\$VARIABLES'

En mode super simplifié



APPELLEUR UN PREPROCESS

- ▶ Différentes fonctions de **preprocess** peuvent être appelées permettant à tous les modules et thèmes de modifier les variables passées aux templates.

Ces fonctions sont les suivantes pour le hook de thème HOOK :

- ▶ **template_preprocess_HOOK(array &\$variables)**
- ▶ **MODULENAME preprocess(array &\$variables, \$hook)**
- ▶ **MODULENAME preprocess_HOOK(array &\$variables)**
- ▶ **THEMENAME preprocess(array &\$variables, \$hook)**
- ▶ **THEMENAME preprocess_HOOK(array &\$variables)**

CONCRÈTEMENT

```
/**  
 * Implements hook preprocess menu account().  
 */  
function drupal_france preprocess menu account(&$variables) {  
  if (isset($variables['items']['drupalfr_user.account.compte'])) {  
    $variables['items']['drupalfr_user.account.compte']['attributes']->addClass([  
      'button',  
      'primary',  
      'button--user-account',  
    ]);  
  
    $variables['items']['drupalfr_user.account.compte']['title'] = new FormattableMarkup(  
      '<i class="fa fa-user-circle-o" aria-hidden="true"></i><span class="sr-  
only">@original_title</span>',  
      [  
        '@original_title' => $variables['items']['drupalfr_user.account.compte']['title'],  
      ]  
    );  
  }  
}
```

```
/**  
 * @file  
 * Shows status messages in a modal in a Foundation-based theme.  
 */  
  
(function ($, Drupal) {  
  
  /**  
   * Displays status messages in a Foundation-based modal.  
   */  
  Drupal.behaviors.foundationStatusInRegion = {  
    attach: function (context, settings) {  
      $('#status-messages').once('foundation-status-messages')  
        // Move the status messages out of the region.  
        .empty()  
        .appendTo('body')  
        .foundation('open');  
  
      // This is required as this region's  
      // messages are removed.  
      $region.html($region.html().replace(/<div/ig, '  
    });  
  };  
});(jQuery, Drupal);
```

#9

JAVASCRIPT

DRUPAL & JAVASCRIPT

- ▶ Drupal intègre de nombreuses libraries javascript ([jQuery](#), [Modernizr...](#)) et propose également des libraries d'utilitaires ([core.drupal](#), [core.drupalSettings](#), [drupal.ajax](#), [drupal.form...](#)).
- ▶ Par exemple, Drupal 8 embarque :
 - ▶ [jQuery 3.2.1](#)
 - ▶ [jQuery UI 1.12.1](#)
- ▶ Chaque librairie définit ses [dépendances](#) : [jQuery UI dépend](#) de [jQuery](#) par exemple.
- ▶ Lorsque l'on utilise une librairie depuis un script custom, il ne faut pas oublier de déclarer cette dépendance.

Pensez à tester en anonyme, car aucune librairie n'est chargée par défaut.
- ▶ Vous disposez de toutes les librairies définies dans le fichier [/core/core.librairies.yml](#).

CONCRÈTEMENT

Pour pouvoir utiliser du code JS dans vos scripts avec Drupal,
la bonne pratique est d'encadrer votre code comme suit :

```
/**  
 * @file  
 * Shows status messages in a modal instead of inline.  
 */  
  
(function ($, Drupal) {  
  
  /**  
   * Displays status messages in a Foundation reveal modal.  
   */  
  Drupal.behaviors.foundationStatusInReveal = {  
    attach: function (context, settings) {  
      $('#status-messages').once('foundation-reveal').each(function () {  
        // Move the status messages out of the highlighted region.  
        var $messages = $(this);  
        var $region = $messages.parent();  
        $messages.appendTo('body');  
        $messages.foundation('open');  
  
        // This is required as this region will likely be empty after status  
        // messages are removed.  
        $region.html($region.html().replace(/\n/g, ''));  
      });  
    };  
  };  
})(jQuery, Drupal);
```

DRUPAL BEHAVIORS

- ▶ Les **scripts sont exécutés** après le chargement complet du **DOM**
`$(document).ready() {};`
Mais on a aussi parfois besoin de les ré-exécuter lorsque l'on interagit avec la page (appel AJAX, overlay...).
- ▶ Pour ce faire il suffit d'attacher le script JS à l'objet **Drupal.behaviors** (librairie **core.drupal**).

La fonction exécutée reçoit 2 variables :

- ▶ **context** : le DOM (le HTML chargé ou ajouté).
- ▶ **settings** : toutes les variables injectées côté serveur.

RAPPEL

- ▶ Comme pour les fichiers de style, les scripts JS doivent être déclarés sous forme de bibliothèque via les fichiers **THEMENAME.info.yml** et **THEMENAME.libraries.yml**.



#10

QUESTIONS ?

RTFM :

<https://www.drupal.org/docs/8/theming>

<https://twig.symfony.com/doc/2.x/>



MERCI À VOUS

@FlorisMoriceau