
Yopa Documentation

Rilis 2.0

Putu Hendra McHen, Agus Seputra, Artha Pico, Hendy Irawan

Sep 27, 2017

| | | |
|----------|---|-----------|
| 1 | Sejarah | 3 |
| 2 | Instalasi | 5 |
| 2.1 | Prasyarat | 5 |
| 2.2 | Membuat Situs Baru | 5 |
| 2.3 | Mengembangkan Situs yang Sudah Ada | 6 |
| 2.4 | Mengembangkan Menggunakan Cloud IDE | 7 |
| 3 | Kustomisasi Dasar | 9 |
| 3.1 | Install Theme | 9 |
| 3.2 | Pathauto | 9 |
| 3.3 | Realname | 10 |
| 3.4 | REST API | 10 |
| 3.5 | Install Display Suite | 10 |
| 3.6 | Drupal 8 Crash Course | 10 |
| 4 | Perawatan | 13 |
| 4.1 | Update (Pemutakhiran) | 13 |
| 4.2 | Backup Berkala | 13 |
| 5 | Development | 15 |
| 5.1 | Kustomisasi Template | 15 |
| 5.1.1 | Nodes | 15 |
| 5.1.2 | Blocks | 19 |
| 5.1.3 | Related topics | 19 |
| 5.2 | Persiapan | 19 |
| 6 | Indices and tables | 21 |

Yoopa adalah *content management system* berbasis Drupal untuk laboratorium penelitian, subproyek dari [Vidyanusa](#) dari [LSKK ITB](#). Drupal 8 sendiri adalah CMS berbasis Symfony 2 framework. Sebenarnya Yoopa diimplementasikan sebagai seperangkat *Drupal modules* dan konfigurasi Drupal, jadi bukan CMS baru yang terpisah.

Yoopa diprakarsai oleh [Putu Hendra McHen \(LinkedIn\)](#), [Agus Seputra](#), dan [Artha Pico](#). Versi 2.0 dilanjutkan oleh [Hendy Irawan](#).

Yoopa digunakan sebagai engine untuk website [LSKK ITB](#).

Yoopa terdiri dari 5 komponen yaitu:

1. **Yoopa Community** adalah produk yang menyimpan informasi profil pengguna dan riwayat hidup pengguna, dimana produk ini membutuhkan proses authentication untuk dapat diakses.
2. **Crayonpedia** adalah produk sejenis Wikipedia yang memiliki banyak konten pendidikan. Produk ini bisa diakses tanpa harus login. Namun untuk melakukan posting artikel pengguna yang telah terdaftar diwajibkan untuk log in.
3. **Yoopa Produk** adalah produk yang menyimpan produk-produk pengguna yang pernah dibuat. Produk ini bisa diakses tanpa harus login. Namun untuk melakukan posting artikel pengguna yang telah terdaftar diwajibkan untuk log in.
4. **Course LSKK** adalah produk yang dibuat menggunakan CMS Moodle untuk manajemen informasi perkuliahan. Produk ini mengharuskan pengguna untuk login.
5. **E-Learning LSKK** adalah produk yang memiliki konten pembelajaran dimana produk ini membutuhkan proses login terlebih dahulu untuk bisa diakses.

Dokumentasi Yoopa ada di <http://yoopa.readthedocs.io/id/latest/>, yang merupakan hasil *build* otomatis dari repository GitHub dokumentasi kolaboratif di <https://github.com/lस्क/yoopa>.

Daftar Isi:

BAB 1

Sejarah

Yoopa 1.0 berbasis CodeIgniter.

Yoopa 2.0 berbasis Drupal 8 dan dikelola dengan [Composer](#).

Untuk menginstall Drupal dengan modul-modul Yoopa, dapat dilakukan untuk membuat situs baru, maupun untuk mengembangkan situs yang sudah ada.

Prasyarat

Baik untuk situs baru maupun situs yang sudah ada, membutuhkan XAMPP dan [Composer](#).

1. Install XAMPP.
2. Recommended: Nyalakan OPCache, caranya sebagai berikut. Edit `php.ini` lalu pastikan:

```
zend_extension=php_opcache.dll
[opcache]
opcache.enable=1
```

3. Unduh dan jalankan [Composer Installer](#). (Di lingkungan jaringan ITB, Anda akan memerlukan setting proxy saat menjalankan *Composer Installer*).

Membuat Situs Baru

Untuk membuat situs baru, langkahnya sama dengan [installing Drupal](#).

TODO: Gunakan <https://github.com/drupal-composer/drupal-project>, sepertinya ini lebih baik karena: Composer by default, Drupal Core dapat diupdate via Composer, support drush, support DrupalConsole.

1. Unduh [Drupal 8.x recommended release](#) terbaru
2. Extract ke folder sementara, misalnya `D:\tmp`. Folder tersebut akan berisi subfolder `drupal-8.x`. Buat folder tujuan sesuai nama web, misalnya `C:\xampp\htdocs\lskk.org`, lalu pindahkan semua isi folder `drupal-8.x` tadi ke folder tersebut.

3. Rename file `example.gitignore` menjadi `.gitignore`. (tambahkan `."` di akhir, yang nanti akan hilang dengan sendirinya, jadi nama file akhirnya adalah `.gitignore`).
4. Buka *Command Prompt*, lalu masuk ke folder website Yoopa/Drupal Anda, contoh:

```
C:
cd \xampp\htdocs\lskk.org
```

5. Set konfigurasi Composer untuk Drupal, caranya:

```
composer config repositories.drupal composer https://packages.drupal.org/8
```

6. Tentukan bagaimana ingin mengakses website tersebut di local, misalnya `lskk.org.amanahwin`. Maka jalankan Notepad/Notepad++ sebagai admin, buka file `C:\Windows\System32\drivers\etc\hosts`, lalu tambahkan di bagian bawah:

```
127.0.0.1    lskk.org.amanahwin
```

7. Edit file `C:\xampp\apache\conf\extra\httpd-vhosts.conf` lalu tambahkan:

```
NameVirtualHost *:80
<VirtualHost *:80>
    ServerName lskk.org.amanahwin
    DocumentRoot "C:/xampp/htdocs/lskk.org"
    SetEnv APPLICATION_ENV "development"
    <Directory "C:/xampp/htdocs/lskk.org">
        DirectoryIndex index.php
        AllowOverride All
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

8. Jalankan *XAMPP Control Panel* dan re-Start *Apache*. Start *MySQL*.
9. Cek website Drupal dapat diakses di <http://lskk.org.amanahwin/> Harusnya menampilkan halaman instalasi. Language: pilih English Profile: pilih Standard Database name: `lskkorg` Database user: `root` Database password: (kosong/sesuai konfigurasi XAMPP)

Mengembangkan Situs yang Sudah Ada

Bila *source code* situs sudah ada, contoh *source code* website <http://www.lskk.org> ada di <https://github.com/lskk/lskk.org>, Anda dapat meng-clone di local PC lalu melanjutkan pengembangannya.

1. Gunakan **TortoiseGit** untuk meng-clone repository website Yoopa yang diinginkan. Misal, buka <https://github.com/lskk/lskk.org>, dari situ Anda akan mendapatkan Clone URL-nya yaitu <https://github.com/lskk/lskk.org.git>
2. Clone ke folder di dalam `C:\xampp\htdocs`, misalnya `C:\xampp\htdocs\lskk.org`, lalu pindahkan semua isi folder `drupal-8.x` tadi ke folder tersebut.
4. Buka *Command Prompt*, lalu masuk ke folder website Yoopa/Drupal Anda, contoh:

```
C:
cd \xampp\htdocs\lskk.org
```

5. Kembalikan *dependencies* dari konfigurasi Composer untuk Drupal, caranya:

```
composer -vvv install
```

6. Tentukan bagaimana ingin mengakses website tersebut di local, misalnya `lskk.org.amanahwin`. Maka jalankan Notepad/Notepad++ sebagai admin, buka file `C:\Windows\System32\drivers\etc\hosts`, lalu tambahkan di bagian bawah:

```
127.0.0.1    lskk.org.amanahwin
```

7. Edit file `C:\xampp\apache\conf\extra\httpd-vhosts.conf` lalu tambahkan:

```
NameVirtualHost *:80
<VirtualHost *:80>
    ServerName lskk.org.amanahwin
    DocumentRoot "C:/xampp/htdocs/lskk.org"
    SetEnv APPLICATION_ENV "development"
    <Directory "C:/xampp/htdocs/lskk.org">
        DirectoryIndex index.php
        AllowOverride All
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

8. **Jalankan XAMPP Control Panel dan re-Start Apache.** Start *MySQL*.
9. **Cek website Drupal dapat diakses di <http://lskk.org.amanahwin/>** Harusnya menampilkan halaman instalasi. Language: pilih English Profile: pilih Standard Database name: `lskkorg` Database user: `root` Database password: (kosong/sesuai konfigurasi XAMPP)

Mengembangkan Menggunakan Cloud IDE

TODO

Kustomisasi Dasar

Setelah instalasi Drupal, maka ada beberapa hal yang perlu dikustomisasi agar website sesuai dengan yang diinginkan.

Install Theme

Yoopa menggunakan [theme Bootstrap](#). Cara installnya:

1. Buka *Command Prompt*, lalu masuk ke folder website Yoopa/Drupal Anda, contoh:

```
C :  
cd \xampp\htdocs\lskk.org
```

2. Install module `drupal/bootstrap` menggunakan Composer:

```
composer -vvv require drupal/bootstrap
```

3. Di website, klik **Manage > Appearance**. Akan muncul theme Bootstrap, klik *Install and set as default*.
4. Di Bootstrap > Settings, Anda dapat mengatur lebih jauh. Bootstrap sendiri mendukung Bootswatch yaitu “sub-theme” dari Bootstrap. Klik **Advanced > Theme > pilih Bootswatch sub-theme pilihan Anda**.

Pathauto

Dibutuhkan agar URI untuk masing-masing *content* bersifat *SEO-friendly*.

1. Install module `drupal/pathauto` menggunakan Composer:

```
composer -vvv require drupal/pathauto
```

2. Di web, klik **Manage > Extend**
3. Centang **Pathauto**, klik *Install*.

4. Manage > Configuration > Development > Performance > Clear all caches
5. Manage > Configuration > Search and metadata > Bulk generate > Content

Realname

TODO: Menunggu kesiapan dari modul [Realname](#) untuk Drupal 8. Terutama, [bug ini](#) masih pending.

REST API

1. Install module [drupal/restui](#) menggunakan Composer:

```
composer -vvv require drupal/restui
```

2. Manage > Extend. Enable modules:

1. HAL
2. HTTP Basic Authentication
3. REST UI
4. RESTful Web Services
5. Serialization

3. Manage > People > Permissions. Set permissions:

- GET: Anonymous, Authenticated
- POST/PATCH/DELETE: Authenticated

4. Manage > Configuration > REST.

Secara *default*, *resource* yang *enabled* adalah Content dengan format `hal_json`. Pastikan hal ini benar.

5. Test menggunakan browser atau Postman: http://lskk.org/amanahwin/node/1?_format=hal_json

Install Display Suite

Display Suite digunakan untuk mengatur layout content secara GUI di website Drupal, tanpa membuat template.

1. Require module [drupal/ds](#) menggunakan Composer:

```
composer -vvv require drupal/ds
```

2. Di web, klik Manage > Extend
3. Centang **Display Suite** dan **Display Suite Extras**, klik *Install*.

Drupal 8 Crash Course

1. [Understanding Drupal 8](#)
2. [Los Angeles 2015: Drupal 8 The Crash Course](#)

3. Drupal 8 The Crash Course presentation
4. Creating Drupal 8 modules
5. How to build a Drupal 8 module - SitePoint

Update (Pemutakhiran)

Update diperlukan untuk memperbaiki bugs dan menambah features, tapi yang paling penting adalah mendapatkan *security fixes* yang melindungi dari malware dan hacker.

Caranya:

1. Buka Command Prompt dan jalankan Composer untuk update:

```
composer -vvv update
```

2. Clear all cache

Backup Berkala

TODO

Kustomisasi Template

Referensi cara kustomisasi template adalah di [Twig Template naming conventions - Drupal](#).

Nodes

Pattern: `node--[type|nodeid]--[viewmode].html.twig` Base template: `node.html.twig` (base location: `core/modules/node/templates/node.html.twig`)

Theme hook suggestions are made based on these factors, listed from the most specific template to the least. Drupal will use the most specific template it finds:

1. `node--nodeid--viewmode.html.twig`
2. `node--nodeid.html.twig`
3. `node--type--viewmode.html.twig`
4. `node--type.html.twig`
5. `node--viewmode.html.twig`
6. `node.html.twig`

Note that underscores in a content type's machine name are replaced by hyphens.

See the [node.html.twig API documentation](#).

node.html.twig

1. 8.2.x `core/themes/bartik/templates/node.html.twig`
2. 8.2.x `core/themes/classy/templates/content/node.html.twig`
3. 8.2.x `core/modules/statistics/tests/themes/statistics_test_attached/node.html.twig`

4. 8.2.x core/modules/system/tests/themes/test_theme/templates/node.html.twig
5. 8.2.x core/modules/node/templates/node.html.twig
6. 8.2.x core/themes/stable/templates/content/node.html.twig

Default theme implementation to display a node.

Available variables:

- **node:** The node entity with limited access to object properties and methods. Only method names starting with “get”, “has”
 - node.getCreatedTime() will return the node creation timestamp.
 - node.hasField(‘field_example’) returns TRUE if the node bundle includes field_example. (This does not indicate the presence of a value in this field.)
 - node.isPublished() will return whether the node is published or not.

Calling other methods, such as node.delete(), will result in an exception. See `Drupal\Node\EntityNode` for a full list of public properties and methods for the node object.

- label: The title of the node.
- content: All node items. Use `{{ content }}` to print them all, or print a subset such as `{{ content.field_example }}`. Use `{{ content|without(‘field_example’) }}` to temporarily suppress the printing of a given child element.
- author_picture: The node author user entity, rendered using the “compact” view mode.
- metadata: Metadata for this node.
- date: Themed creation date field.
- author_name: Themed author name field.
- url: Direct URL of the current node.
- display_submitted: Whether submission information should be displayed.
- **attributes:** HTML attributes for the containing element. The `attributes.class` element may contain one or more of the following
 - node: The current template type (also known as a “theming hook”).
 - node-type-[type]: The current node type. For example, if the node is an “Article” it would result in “node-type-article”. Note that the machine name will often be in a short form of the human readable label.
 - node-view-mode-[view_mode]: The View Mode of the node; for example, a teaser would result in: “node-view-mode-teaser”, and full: “node-view-mode-full”.

The following are controlled through the node publishing options.

- node-promoted: Appears on nodes promoted to the front page.
- node-sticky: Appears on nodes ordered above other non-sticky nodes in teaser listings.
- node-unpublished: Appears on unpublished nodes visible only to site admins.
- title_attributes: Same as attributes, except applied to the main title tag that appears in the template.
- content_attributes: Same as attributes, except applied to the main content tag that appears in the template.
- author_attributes: Same as attributes, except applied to the author of the node tag that appears in the template.
- title_prefix: Additional output populated by modules, intended to be displayed in front of the main title tag that appears in the template.

- `title_suffix`: Additional output populated by modules, intended to be displayed after the main title tag that appears in the template.
- `view_mode`: View mode; for example, “teaser” or “full”.
- `teaser`: Flag for the teaser state. Will be true if `view_mode` is ‘teaser’.
- `page`: Flag for the full page state. Will be true if `view_mode` is ‘full’.
- `readmore`: Flag for more state. Will be true if the teaser content of the node cannot hold the main body content.
- `logged_in`: Flag for authenticated user status. Will be true when the current user is a logged-in member.
- `is_admin`: Flag for admin user status. Will be true when the current user is an administrator.

@todo Remove the id attribute (or make it a class), because if that gets rendered twice on a page this is invalid CSS for example: two lists in different view modes.

File

core/modules/node/templates/node.html.twig

```
{ #
/**
 * @file
 * Default theme implementation to display a node.
 *
 * Available variables:
 * - node: The node entity with limited access to object properties and methods.
 *   Only method names starting with "get", "has", or "is" and a few common
 *   methods such as "id", "label", and "bundle" are available. For example:
 *   - node.getCreatedTime() will return the node creation timestamp.
 *   - node.hasField('field_example') returns TRUE if the node bundle includes
 *     field_example. (This does not indicate the presence of a value in this
 *     field.)
 *   - node.isPublished() will return whether the node is published or not.
 *   Calling other methods, such as node.delete(), will result in an exception.
 *   See \Drupal\node\Entity\Node for a full list of public properties and
 *   methods for the node object.
 * - label: The title of the node.
 * - content: All node items. Use {{ content }} to print them all,
 *   or print a subset such as {{ content.field_example }}. Use
 *   {{ content|without('field_example') }} to temporarily suppress the printing
 *   of a given child element.
 * - author_picture: The node author user entity, rendered using the "compact"
 *   view mode.
 * - metadata: Metadata for this node.
 * - date: Themed creation date field.
 * - author_name: Themed author name field.
 * - url: Direct URL of the current node.
 * - display_submitted: Whether submission information should be displayed.
 * - attributes: HTML attributes for the containing element.
 *   The attributes.class element may contain one or more of the following
 *   classes:
 *   - node: The current template type (also known as a "theming hook").
 *   - node--type-[type]: The current node type. For example, if the node is an
 *     "Article" it would result in "node--type-article". Note that the machine
 *     name will often be in a short form of the human readable label.
 *   - node--view-mode-[view_mode]: The View Mode of the node; for example, a
 *     teaser would result in: "node--view-mode-teaser", and
```

```

*      full: "node--view-mode-full".
*      The following are controlled through the node publishing options.
*      - node--promoted: Appears on nodes promoted to the front page.
*      - node--sticky: Appears on nodes ordered above other non-sticky nodes in
*        teaser listings.
*      - node--unpublished: Appears on unpublished nodes visible only to site
*        admins.
*      - title_attributes: Same as attributes, except applied to the main title
*        tag that appears in the template.
*      - content_attributes: Same as attributes, except applied to the main
*        content tag that appears in the template.
*      - author_attributes: Same as attributes, except applied to the author of
*        the node tag that appears in the template.
*      - title_prefix: Additional output populated by modules, intended to be
*        displayed in front of the main title tag that appears in the template.
*      - title_suffix: Additional output populated by modules, intended to be
*        displayed after the main title tag that appears in the template.
*      - view_mode: View mode; for example, "teaser" or "full".
*      - teaser: Flag for the teaser state. Will be true if view_mode is 'teaser'.
*      - page: Flag for the full page state. Will be true if view_mode is 'full'.
*      - readmore: Flag for more state. Will be true if the teaser content of the
*        node cannot hold the main body content.
*      - logged_in: Flag for authenticated user status. Will be true when the
*        current user is a logged-in member.
*      - is_admin: Flag for admin user status. Will be true when the current user
*        is an administrator.
*
* @see template_preprocess_node()
*
* @todo Remove the id attribute (or make it a class), because if that gets
*   rendered twice on a page this is invalid CSS for example: two lists
*   in different view modes.
*
* @ingroup themeable
*/
#}
<article{{ attributes }}>

{{ title_prefix }}
{% if not page %}
  <h2{{ title_attributes }}>
    <a href="{{ url }}" rel="bookmark">{{ label }}</a>
  </h2>
{% endif %}
{{ title_suffix }}

{% if display_submitted %}
  <footer>
    {{ author_picture }}
    <div{{ author_attributes }}>
      {% trans %}Submitted by {{ author_name }} on {{ date }}{% endtrans %}
      {{ metadata }}
    </div>
  </footer>
{% endif %}

<div{{ content_attributes }}>
  {{ content }}

```

```
</div>

</article>
```

Blocks

Pattern: `block--[module|-delta]].html.twig` **Base template:** `block.html.twig` (base location: `core/modules/block/templates/block.html.twig`)

1. `block--module--delta.html.twig`
2. `block--module.html.twig`
3. `block.html.twig`

“module” being the name of the module and “delta”, the internal id assigned to the block by the module.

For example, “`block-block-1.html.twig`” would be used for the first user-submitted block added from the block administration screen since it was created by the block module with the id of 1. Region-specific block templates are not available in Drupal 8.

If you had a block created by a custom module called “custom” and a delta of “my-block”, the theme hook suggestion would be called “`block-custom-my-block.html.twig`.”

Also one more example with Views, if you have a block created by views with a view name “`front_news`” and display id “`block_1`” then the theme hook suggestion would be: `block-views-block-front-news-block-1.html.twig` (notice, when you have underscores in a display id or in a view name - you have to transform them in to a single dash)

Be aware that module names are case sensitive in this context. For instance if your module is called ‘MyModule’, the most general theme hook suggestion for this module would be “`block-MyModule.html.twig`.”

See the [block.html.twig API documentation](#).

Related topics

- [Theme system overview](#): Functions and templates for the user interface that themes can override.

Persiapan

1. [Disable Drupal 8 caching during development](#). Ini perlu agar tidak bolak-balik *Clear cache*.

Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)