

e107 + AYRO UI

How to create e107 theme with AYRO UI

Create your theme for e107 CMS version 2.3.3

AYRO UI - bootstrap 5 based components

Ayro UI Core Core framework is free to download



Free Bootstrap Snippets, UI Components, and Library | Ayro UI



Official site of AYRO UI

e107 CMS - bootstrap based content management system

Free and Open Source Content Management system without need of coding skills. User management system, blog posts, forum, static pages, contact, faqs and a lot of other things.



e107 Bootstrap CMS Open Source

e107



Official site of e107 CMS

e107 Stuff

General info

Just basic information related to e107 themes. For more information look into official documentation.

Templates:



Templates

e107 Developer Guide



Shortcodes:



Core Shortcodes

e107 Developer Guide



Constants



Constants

e107 Developer Guide



Theme Basics



Templating/Shortcodes

e107



Theme Folder

Theme folder - any standard folder name f.e. your_theme
"e107_themes/your_theme"

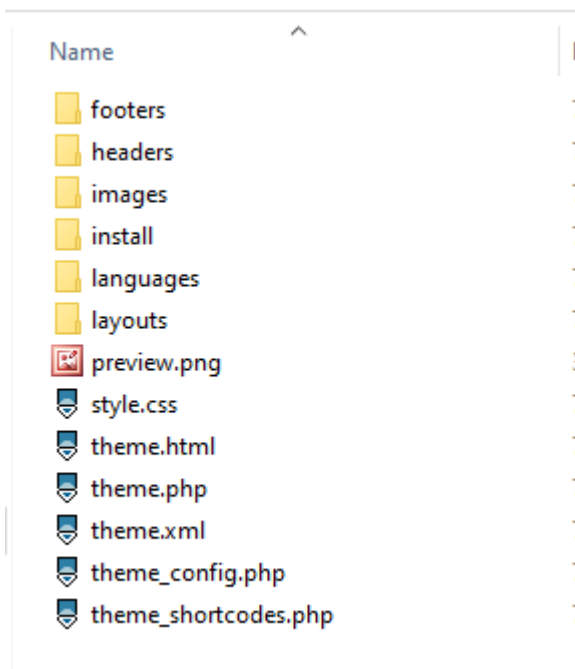
Mandatory files:

- theme.xml
- style.css
- preview.png
- theme.php
- theme.html

Optional files

- theme_config.php
- theme_library.php
- theme_shortcodes.php

Example from some other theme:



Theme Constants

Libraries

If you use libraries from theme.xml, they are set by the system. In some cases (or in older themes) you can find this:

```
define("BOOTSTRAP", 3);  
define("FONTAWESOME", 4);
```

Output can vary according to bootstrap or fontawesome version. The same applies for non-bootstrap themes.

Outdated constants

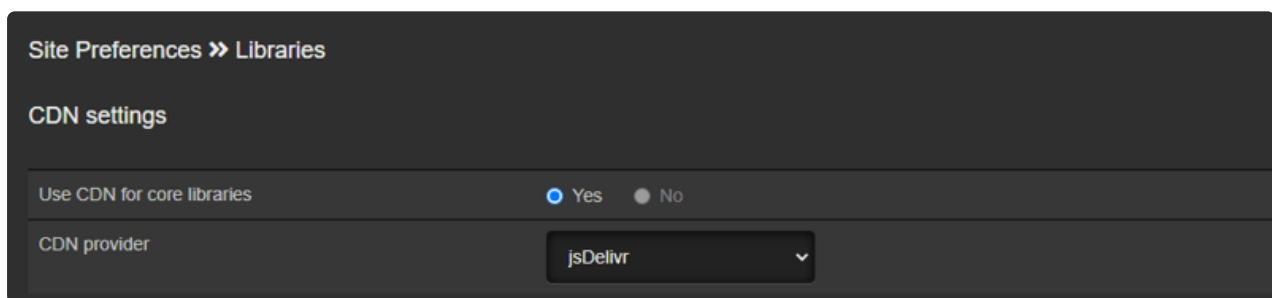
```
define('VIEWPORT', "width=device-width, initial-scale=1.0");  
define('HTMLTAG', "<html id='theme-".THEME_LAYOUT.'" lang='".CORE_LC.'" >
```

Theme Libraries

From theme.xml

```
<libraries>  
<library name="bootstrap" version="5" scope="front" />  
<library name="fontawesome" version="5" scope="front" files="js" />  
<library name="fontawesome" version="5" scope="wysiwyg" files="css" />  
</libraries>
```

The way how to include libraries from the core. You can set if to use CDN or not in site preferences:



Site preferences - Libraries

There are listed all available libraries too.

Other examples:

```
<library name="animate.css" scope="front" />  
<library name="bootstrap.editable" scope="admin" />
```

Attribute	Description	Example
name	Library name	
version	Library version	version="4" loads bootstrap4
scope	Where to load this library	front, admin, wysiwyg
file	What type of files to load	css, js

- i Those files are loaded automatically, without writing a line of code in the theme itself.

From theme.php

Old way:

```
e107::library('load', 'bootstrap');  
e107::library('load', 'fontawesome');  
+  
define("BOOTSTRAP", 3);  
define("FONTAWESOME", 4);
```


Theme Css Files

Note: see Libraries for system loaded css

From theme.xml

```
<stylesheets>
  <css file="style.css" name="Default" />
</stylesheets>
```

Last loaded file. Used for selection of variable skins.

IMPORTANT: only way how to exclude bootstrap core css to load with bootstrap theme

```
<css file="default.css" name="Default" default='true' scope='front' excl
```

From theme.php

in theme class standard way for loading css files

```
e107::css('type', 'path');
```

Examples:

```
e107::css('theme', 'css/style.css');
e107::css('theme', 'templates/forum/style_forums.css');

$inlinecss = e107::pref('themename', 'inlinecss');
if($inlinecss) {
    e107::css("inline", $inlinecss);
}
```

Load fonts:

```
e107::css('url', 'https://fonts.googleapis.com/css?family=Nunito');
```

Theme Js Files

Note: see Libraries for system loaded js

From theme.php

External:

```
e107::js("url", "https://cdn.jsdelivr.net/bootstrap/3.3.7/js/bootstrap.mi
```

External in footer:

```
e107::js("footer", "https://cdnjs.cloudflare.com/ajax/libs/jquery-easing/
```

Local from theme:

```
e107::js("theme", "js/wow.js", 'jquery');
```

Inline script:

```
e107::js("footer-inline", "$('.e-tip').tooltip({container: 'body'}));
```

JQuery

JQuery is loaded automatically. You can remove JQuery from the frontend with theme_library.php file, but it is not recommended.

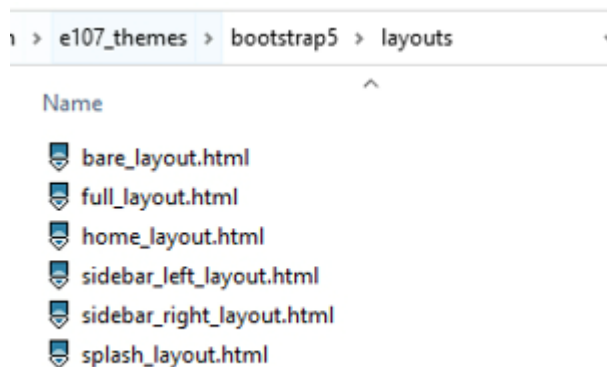
- i** You can change core libraries from theme with theme library class, but it is the advanced stuff, not covered with this.

Theme Layouts

From theme.xml

```
<layouts>
  <layout name='home' title='Home Page' >
    <custompages>FRONTPAGE</custompages>
  </layout>
  <layout name='full' title='Full Page' >
    <custompages>forum/ download/</custompages>
  </layout>
  <layout name='sidebar_left' title='Sidebar (left)' />
  <layout name='sidebar_right' title='Sidebar (right)' default='true' />
  <layout name='splash' title='Splash Page' >
    <custompages>login fpw signup</custompages>
  </layout>
  <layout name='bare' title='Bare Page' >
    <custompages></custompages>
  </layout>
</layouts>
```

From layouts folder



From theme settings

Layouts:

DEFAULT	NAME	VISIBILITY FILTER
<input type="radio"/>	Home Page home	FRONTPAGE
<input type="radio"/>	Full Page full	forum/ download/
<input type="radio"/>	Sidebar (left) sidebar_left	None
<input checked="" type="radio"/>	Sidebar (right) sidebar_right	None
<input type="radio"/>	Splash Page splash	login/ pw/ signup
<input type="radio"/>	Bare Page bare	None

Global Shortcodes

Related to theming, available anywhere (HTML files, templates)

{SITETITLE}

Global shortcode

Inserting content of site name preference

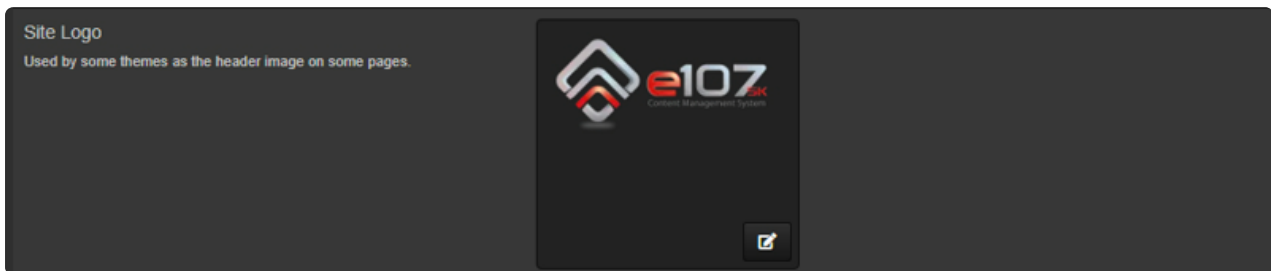
Site Preferences » Site Information	
Site Name	e107sk.com

{SITELOGO}

or {LOGO}

It is a bit complicated, very often replaced with theme prefs

source code is in `e_shortcode.php` in `siteinfo` plugin



{SITEDISCLAIMER}

Example

```
<div class="right">
  <p>{SITEDISCLAIMER}</p>
</div>
```

Content of this pref is used:

<p>Site Disclaimer</p> <p>Used by some themes. Place 'SITEDISCLAIMER' in your theme to use this value.</p>	<input type="text"/>
--	----------------------

Sitedisclaimer preference

If nothing is set, then default text is used: *Proudly powered by e107 which is released under the terms of the GNU GPL License.*

If you use YYYY in your text, it will be replaced by the actual year:

```
© Copyright 2015-YYYY Your company, All rights reserved.
```

Result: © Copyright 2015-2022 Your company, All rights reserved.

Other

Usefull code snippets

Useful Core Preferences

Some core preferences for theming

What CDN to use:

```
$cdnPref = e107::pref('theme', 'cdn', 'cdnjs');
```

Header

Theme header = navbar

- navbar branding
- main navigation
- signin navigation
- search box
- banner
- header social links
- etc

Custom theme shortcode is used in the starter theme. Core magic shortcode `{--
-HEADER---}` can be used too, if theme has only one header.

```
PAGE  
{NAVIGATION} >
```

```
PAGE  
{NAVBAR__SITESEARCH} >
```

```
PAGE  
{SIGNIN} >
```

Navbar branding

In theme configuration, you can select supported options:

Default:

- Site Logo
- Site Name

- Site Logo + Name

Special:

- Site Logo + Name + Sitetag

If you want to customize this area, look for a shortcode `{NAVBAR_BRANDING}` in `theme_shortcodes.php` of the site theme.

Footer

- sitedisclaimer
- footer navigation
- footer columns
- footer social links
- contact info
- working hours
- etc

Custom theme shortcode is used in starter theme. Core magic shortcode `{---FOOTER---` can be used too, if theme has only one footer.

Starter theme

theme.xml

```
<?xml version="1.0" encoding="utf-8"?>
<e107Theme name="Bootstrap AYUI Starter" version="1.0.0" date="2022-07-26"
  <author name="Jimako" url="https://www.e107sk.com" />
  <summary lan="">Bootstrap 5 AYRO UI e107 Starter theme</summary>
  <description lan="">Bootstrap AYRO UI e107 5 Starter theme</descript
  <keywords>
    <word>bootstrap</word>
    <word>responsive</word>
  </keywords>
  <category>generic</category>
  <screenshots>
    <image>preview.jpg</image>
  </screenshots>
  <libraries>
    <library name="bootstrap" version="5" scope="front"/>
    <library name="fontawesome" version="5" scope="front" files="js"
  <library name="fontawesome" version="5" scope="wysiwyg" files="css" /
  </libraries>
  <stylesheets>
    <css file="assets/css/bootstrap.min.css" name="Don't use this" s
    <css file="default.css" name="Default" default='true' scope='fro
  </stylesheets>
  <layouts>
    <layout name='home' title='Home Page' >
      <custompages>FRONTPAGE</custompages>
    </layout>
    <layout name='full' title='Full Page' >
      <custompages>forum/ download/</custompages>
    </layout>
    <layout name='sidebar_left' title='Sidebar (left)' />
    <layout name='sidebar_right' title='Sidebar (right)' default='tru
    <layout name='splash' title='Splash Page' >
      <custompages>login fpw signup</custompages>
    </layout>
    <layout name='bare' title='Bare Page' >
      <custompages></custompages>
    </layout>
  </layouts>
</e107Theme>
```

- Layouts are the same as in the core bootstrap5 theme
- Bootstrap css is excluded - it will be compiled together with theme styling
-

- The stylesheets version is a workaround for the little core bug. file default.css is empty for now
- style.css is loaded manually

Missing:

- using a custom icon set
- adding predefined menus

theme.php

Minimal:

```
<?php

if (!defined('e107_INIT')) {
    exit();
}

class theme implements e_theme_render
{
    public function init()
    {
    }
    public function tablestyle($caption, $text, $mode = '', $options = [])
    {
        if (!empty($caption)) {
            echo '<h4>'.$caption.'</h4>';
        }
        echo $text;
        return;
    }
}
```

Init()

For loading constants, css, js files - see Theme Head Part

tablestyle()

For outputting content.

Init() method

part of theme inside <head> </head> tags

```
public function init()
{
    e107::lan('theme');

    // Your own css fixes
    define('CORE_CSS', false);

    // CONSTANTS
    define("THEME_DISCLAIMER", '');

    // Theme meta tags
    $this->set metas();

    // Theme css
    $this->register_css();

    // Theme js
    $this->register_js();

    // Theme fonts
    $this->register_fonts();

    // Theme icons
    $this->register_icons();

    // Inline css and js
    $this->getInlineCodes();
}
```

i This structure of theme class is not mandatory. All code can be in the init() method.

This will be outputted between <head></head> tags

```
//////// Theme meta tags //////////////////////////////////////////
$this->set metas();

//////// Theme css //////////////////////////////////////////
$this->register_css();

//////// Theme js //////////////////////////////////////////
$this->register_js();

//////// Theme fonts //////////////////////////////////////////
$this->register_fonts();

//////// Theme icons //////////////////////////////////////////
$this->register_icons();

//////// Inline css and js //////////////////////////////////////////
$this->getInlineCodes();
```

Setting meta tags

```
public function set metas()
{
    e107::meta('viewport', 'width=device-width, initial-scale=1.0');
}
```

Include css files

```

public function register_css()
{
    e107::css('theme', 'e107.css'); //if define('CORE_CSS', false);

    e107::css('theme', 'sass-starter/theme.css'); //only for develop

    //bootstrap is compiled in theme.css
    //e107::css('theme', 'assets/css/bootstrap.min.css');

    e107::css('theme', 'assets/css/glightbox.min.css');
    // e107::css('theme', 'assets/css/lineicons.css'); moved to icons
    e107::css('theme', 'assets/css/loading-bar.min.css');
    e107::css('theme', 'assets/css/tiny-slider.css');

    //missing stuff, quick fixes and css no used in sass version
    e107::css('theme', 'style.css');
}

```

! e107.css is normally loaded by core. It is a needed file but sometimes there is a conflict with the theme css (classes are too general) or they change something in the future that can change the theme. It is just a personal preference to set CORE_CSS as false and copy the actual version of that file to the theme.

sass-starter is the folder with the sass version of the components.

style.css is normally loaded by core, but because of the possibility of different skins in the future, default.css is used (and loaded by core) and style.css is loaded this way. Here is used as a quick way how to add some css not supported by AyroUI by default.

Note: style.css (or theme.css) should have css for all skins, default.css and f.e. blue.css only css related to skin. But it is a far future.

The rest of css files are from the asset folder of Ayro UI bootstrap-ui-components-main folder.

Include js files

```

public function register_js()
{
    //e107::js('theme', 'bootstrap.bundle.min.js', 'jquery');

    e107::js('theme', 'assets/js/count-up.min.js', 'jquery');
    e107::js('theme', 'assets/js/glightbox.min.js', 'jquery');
    e107::js('theme', 'assets/js/loading-bar.min.js', 'jquery');
    e107::js('theme', 'assets/js/tiny-slider.js', 'jquery');
    e107::js('theme', 'assets/js/main.js', 'jquery');

    //e107::js('theme', 'fix.js', 'jquery');
}

```

Bootstrap js is loaded by core from theme.xml. Others files are from the AYRO UI js folder.

Fix.js is used for some fixes, not needed now.

Note: very often is main theme script separated into smaller files for better understanding.

Include fonts

In this case

```

public function register_fonts()
{
    e107::css('url', 'https://fonts.googleapis.com/css2?family=Inter:
}

```

- this is moved from `starter.scss` file

```
@import url("https://fonts.googleapis.com/css2?family=Inter:wght@200;300;
```

It could stay here, it would work, but e107 can consolidate and cache javascript files and cascading stylesheet files and it sometimes dislikes using `@import`

Include icons

AYRO UI uses its own icons, for using it in HTML code this solution is enough. For using it in TinyMCE and glyph manager, it should be done a different way. But this is advanced stuff, not covered for now.

```
public function register_icons()  
{  
    //e107::css('theme', 'assets/css/lineicons.css');  
    e107::css('url', 'https://cdn.lineicons.com/3.0/lineicons.css');  
}
```

Include inline codes

For now empty. Advanced stuff.

```
public function getInlineCodes()  
{  
}
```


Tablerender() method

The standard way how to output content in e107 is using rendering functionality.

If you see in core, plugins, shortcodes something like this:

```
e107::getRender()->tablerender($caption, $text ); or  
$ns->tablerender($caption, $content);
```

then `tablestyle()` from the actual theme is used.

\$caption - is caption or label. It can be hard coded (some LAN string) or it can be loaded from a template. Or sometimes it is an empty string.

\$text - is HTML output, the content itself.

Tablerender can be used more than one time on the page (f.e. one time for the main page (blog post) and next time for each menu (widgets)).

```
public function tablestyle($caption, $text, $mode = '', $options = []  
{  
    $style = varset($options['setStyle'], 'default');  
  
    $id = varset($options['uniqueId'], '');  
  
    if (!empty($caption))  
    {  
        echo '<h4>'.$caption.'</h4>';  
    }  
    echo $text;  
  
}
```

\$style is variable that is used later in method to combine **\$caption** and **\$text** with HTML tags. Its value is set by the shortcode `{SETSTYLE=xxx}`, but it can be changed later according to values of **\$id** and **\$mode** variables.

The first thing when developing a new theme is adding this 2x in tablestyle() method... first time at the beginning, next time after the last change of \$style variable;

```
if (e_DEBUG) {
    echo '
    <!-- tablestyle initial: style='.$style.' mode='.$mode.' UniqueId=
    '
}
```

It is hard to remember all values for \$id's and \$mode's, so this is way how to find them easily.

Example for news grid menu with {SETSTYLE=none}

```
<!-- Menu Start: news_grid_menu -->
<!-- tablestyle initial: style=none mode=news_grid_menu UniqueId=news-news-grid-menu -->
<!-- tablestyle initial: style=none mode=news_grid_menu UniqueId=news-news-grid-menu -->
▶<section class="blog-area pb-5">...</section>
<!-- Menu End: news_grid_menu -->
```

Then add a switch for changing style according to \$mode

```
switch ($mode) {
    case "wm":
        $style = "wm";
        break;
}
```

Then style "wm" is used (you can use directly \$mode variable but then you can't use the same style for other elements).

```
<div class="header-content">
  <!-- tablestyle initial: style=heading-1 mode=wm UniqueId= -->
  <!-- tablestyle initial: style=wm mode=wm UniqueId= -->
  <h4>Welcome Message</h4>
```

You can add switch for UniqueId too - it is used mostly for menus (widgets)

```
<div class="widget">
  <!-- Menu Start: news_months_menu -->
  <!-- tablestyle initial: style=widget mode=news_months_menu UniqueId=news-news-months-menu -->
  <!-- tablestyle initial: style=widget mode=news_months_menu UniqueId=news-news-months-menu -->
  <div class="widget">
```

Basic switch for \$style

```
switch($style)
{
    case "none":
        echo $text;
        break;

    case "main":
    case "default":
    default:
        if (!empty($caption)) {
            echo '<h4>'.$caption.'</h4>';
        }
        echo $text;
        break;

    return;
}
```

i Style "default" and default markup are different things!

! You can use e107 magic shortcodes only for layout with styles "default" and "main".

Recommended (not mandatory) styles:

- layout home - {SETSTYLE=none}{---} - homepage is mostly created from components, not page content itself
- layout full - {SETSTYLE=main}{---} - for standalone pages - title + content
- layout sidebar-right - {SETSTYLE=default}{---} - for others pages, news mainly

After adding some components (example):

```
switch($style)
{
    case "none":
        echo $text;
        break;

    case "heading-1":
        if (!empty($caption)) {
            echo '<h1 class="header-title">'.$caption.'<h1>';
        }
        echo $text;
        break;

    case "heading-3":
        if (!empty($caption)) {
            echo '<h3 class="header-title">'.$caption.'<h3>';
        }
        echo $text;
        break;

    case "widget":
        echo '<div class="widget">';
        if (!empty($caption)) {
            echo '<h5 class="widget-title">'.$caption.'<h5>';
        }
        echo $text;
        echo '</div>';
        break;

    case "footer-link":
        echo '<div class="footer-link">';
        if (!empty($caption)) {
            echo '<h6 class="footer-title">'.$caption.'<h6>';
        }
        echo $text;
        echo '</div>';
        break;

    case "main":
    case "default":
    default:
        if (!empty($caption)) {
            echo '<h4>'.$caption.'</h4>';
        }
        echo $text;
        break;

    return:
```

}

}

theme.html

Basic site layout:

```
<body id="page-top" class="{LAYOUT_ID}" {BODY_ONLOAD}>
  {---MODAL---}

  {---HEADER---}

  {---LAYOUT---}

  {---FOOTER---}
</body>
```

Because there is no HTML code for the header or footer, see to theme_shortcodes how it replaces it.

Because there is no HTML code around {---LAYOUT---} all code is in layouts files themselves.

{LAYOUT_ID} is a class generated by core - you can find this way what layout was used

```
▶ <head>...</head>
  <!-- Start theme.html -->
... ▼ <body id="page-top" class="layout-sidebar-right" > == $0
  ▼ <div id="uiModal" class="modal fade" tabindex="-1" role="dialog" aria-hidden="true">
    ▶ <div class="modal-dialog modal-lg modal-xl modal-dialog-centered modal-dialog-scro
```

{LAYOUT_ID} example

theme_config.php

```
<?php

if (!defined('e107_INIT')) { exit; }

$sitetheme = e107::getPref('sitetheme');

e107::themeLan('admin', $sitetheme, true);

class theme_config implements e_theme_config
{

    function __construct()
    {

    }

    function config()
    {
        ... array od prefs ...
    }

    function help()
    {
        return null;
    }

    function process()
    {
        return null;
    }

}
```

preloader example

```
return array(
    'preloader' => array('title' => 'Display preloader', 'type'=>'boolean'
)
```

Next example - if you want to switch between footers (it will be explained in footers component part)

```
$footers = array('default.html'=>'default', 'footer-01.html'=>'Footer 01'  
  
return array(  
    'preloader' => array('title' => 'Display preloader', 'type'=>'boolean'  
    'backtotop' => array('title' => 'Display Back Top Button', 'type'=>'boolean'  
    'footer'     => array('title'=>'Select Footer', 'type'=>'dropdown', 'width'  
);
```


Theme Shortcodes

Theme shortcodes are a way how to extend, customize or override core stuff.

No shortcode is mandatory.

```
<?php  
  
class theme_shortcodes extends e_shortcode  
{  
    ...  
}
```

Custom theme shortcodes

Shortcode	
{---HEADER---	see Layout shortcodes
{---FOOTER---	see Layout shortcodes
{NAVBAR_BRANDING}	see Navbar shortcodes
{COMPONENT}	see Content shortcodes
{THEME_PRELOADER}	see Content shortcodes
{THEME_BACKTOTOP}	see Content shortcodes
{THEME_PREF}	see Other
{THEME_PATH}	see Other

Overridden core shortcodes

{CONTACT_INFO}	make them global for use in the header and footer
----------------	---

{SIGNIN_PM_NAV}	because AYRO UI dropdowns it needs different HTML markup
{FPW_SUBMIT}	different classes for submit button

- some problems can be solved in different way (styling), but this is the quick way

Layout shortcodes

File structure and name conventions are used from AYRO UI.

e107 uses Menu Manager for adding menus (widgets) to Menu areas. All this could be solved with component shortcodes but then menu areas are not displayed correctly in Menu Manager. It is related to a way how e107 parses and detects {MENUAREA} shortcode.

Gold rule: if there is not {MENU} or {MENUAREA} in html file (component) you need to use ParseTemplate() in theme shortcodes - *see header*.

If there are menu areas in the HTML file, you can't parse them in theme shortcodes - *see footer*.

{MENUARES} in layout files are safe and they are parsed without issues.

If you don't see your menu areas like below, it is always a problem with parsing.



! Before trying to understand how those shortcodes work, you should be familiar with AYRO UI folder structure.

{---FOOTER---

```

/**
 * Special Footer Shortcode for dynamic menuarea templates.
 * @shortcode {---FOOTER---}
 * @return string
 */
function sc_footer()
{
    $theme_name = e107::getPref('sitetheme');

    $pref = e107::getThemeConfig($theme_name)->getPref();

    $key = varset($pref['footer'], 'footer-01');

    $themepath = e_THEME . $theme_name;

    $path_html = "{$themepath}/components/footers/{$key}.html";

    if (e_PAGE == "menus.php")
    {

        if (defined("e_MENUMANAGER_ACTIVE") and e_MENUMANAGER_ACTIVE)
        {
            //standard way
            if (file_exists($path_html))
            {
                $text = file_get_contents($path_html);
                // $text = e107::getParser()->parseTemplate($text);
            }
            else
            {
                $text = '';
            }

            $path_css = "components/footers/{$key}.css";

            if (file_exists($themepath . "/" . $path_css))
            {
                e107::css("url", $themepath . "/" . $path_css);
            }

        }
        else //menu selection
        {

            $text = " {MENUAREA=41}{MENUAREA=42}{MENUAREA=43}{MENUARE
        }
    }
}

```

```
else
{

    $path_html = "{$themepath}/components/footers/{$key}.html";

    if (file_exists($path_html))
    {
        $text = file_get_contents($path_html);
        // $text = e107::getParser()->parseTemplate($text);
    }
    else
    {
        $text = '';
    }

    $path_css = "components/footers/{$key}.css";

    if (file_exists($themepath . "/" . $path_css))
    {
        e107::css("theme", $path_css);
    }
}

return $text;
}
```

{---HEADER---}

{---HEADER---}

This replaces `{---HEADER---`} in theme.html.

```

/**
 * Special Header Shortcode for dynamic menuarea templates.
 * @shortcode {---HEADER---}
 * @return string
 */
function sc_header()
{
    $theme_name = e107::getPref('sitetheme');

    $pref = e107::getThemeConfig($theme_name)->getPref();

    $key = varset($pref['navbar'], 'navbar-01');

    $themepath = e_THEME . $theme_name;

    $path_html = "{$themepath}/components/navbars/{$key}.html";

    $path_css = "components/navbars/{$key}.css";

    if (file_exists($path_html))
    {
        $text = file_get_contents($path_html);
        $text = e107::getParser()->parseTemplate($text);
    }
    else
    {
        $text = '';
    }

    $path_css = "components/navbars/{$key}.css";

    if (file_exists($themepath . "/" . $path_css))
    {
        e107::css("url", $themepath . "/" . $path_css);
    }

    return $text;
}

```


Navbar shortcodes

Usefull shortcodes for navbar

{NAVBAR_BRANDING}

This shortcode is with the bootstrap3 theme under the name BOOSTRAP_BRANDING.

```
/* {NAVBAR_BRANDING} */
public function sc_navbar_branding()
{
    $pref = e107::pref('theme', 'branding');
    switch ($pref) {
        case 'logo':
            $brand = '{SITELOGO: h=50}';
            break;
        case 'sitenamelogo':
            $brand = '{SITELOGO: h=50}'.SITENAME;
            break;
        case 'sitename':
        default:
            $brand = SITENAME;
            break;
    }
    $text = '<a class="navbar-brand" href="{SITEURL}" alt="{SITENAME}'
    $text = e107::getParser()->parseTemplate($text);

    return $text;
}
```

Easy way how to avoid using this shortcode:

```
<a href="{SITEURL}"></a>
```

Theme pref is needed, see theme_config.php

PAGE

theme_config.php

>

PAGE

{NAVBAR_SITESEARCH}

>

Content shorcodes

{THEME_PRELOADER}

```
/* {THEME_PRELOADER} */
//<!-- =====>
//<!-- Preloader =====>
function sc_theme_preloader($parm = null)
{
    $pageloader = '';
    $display = e107::pref('theme', 'preloader');
    $display = varset($display, true);

    if($display) {
        $pageloader = '<!-- Preloader -->
        <div class="preloader">
            <div class="preloader-inner">
                <div class="preloader-icon">
                    <span></span>
                    <span></span>
                </div>
            </div>
        </div>
        <!-- /End Preloader -->';
    }

    return $pageloader;
}
```

{THEME_BACKTOTOP}

```
/* {THEME_BACKTOTOP} */
//<!-- ===== scroll-top ===== -->

public function sc_theme_backtotop($parm = null)
{
    $text = '';
    $display = e107::pref('theme', 'backtotop');
    $display = varset($display, true);

    if ($display)
    {
        e107::js('theme', 'js/backtotop.js', 'jquery');
        $text = '<a href="#" class="scroll-top btn-hover">
<i class="lni lni-chevron-up"></i>
</a>';
    }

    return $text;
}
```

{COMPONENT}

```
/* term snippet is used for form elements */
/* {COMPONENT: type=navbars&key=navbar-01} */
/* Warning! This can't be used for footer with menus */
public function sc_component($parm = array())
{
    if (!isset($parm['type']))
    {
        return '';
    }
    $type = $parm['type'];

    $key = varset($parm['key'], 'default');

    $themepath = e_THEME . e107::getPref('sitetheme');

    $path_html = "{$themepath}/components/{$type}/{$key}.html";
    $path_css = "components/{$type}/{$key}.css";

    if (file_exists($path_html))
    {
        $text = file_get_contents($path_html);
    }
    else
    {
        $text = '';
    }
    if (file_exists($themepath . "/" . $path_css))
    {
        e107::css("theme", $path_css);
    }

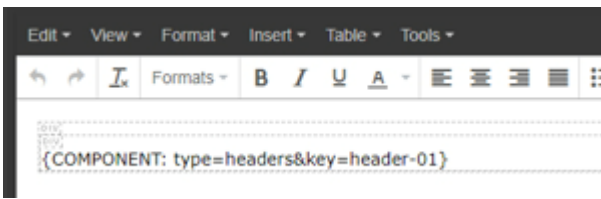
    $text = e107::getParser()->parseTemplate($text);
    return $text;
}
```

Example

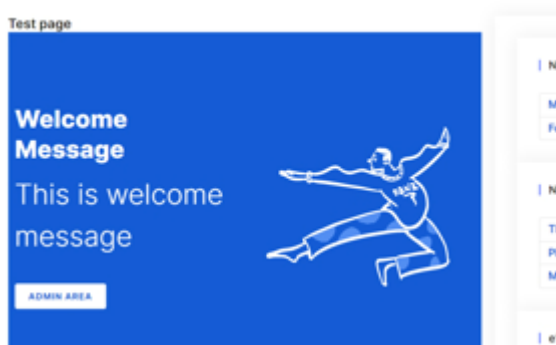
```
{COMPONENT: type=headers&key=header-02}
```

as header in home layout

You can use it directly in TinyMce



Result: (page is not styled yet)



Other

{THEME_PREF}

}

{THEME_PREF}

mainly used for classes or similar stuff that output HTML code

```
function sc_theme_pref($parm = array())
{
    $themePrefs = e107::pref('theme');

    $name = $parm['name'];
    if(!isset($name))
    {
        return "";
    }
    $default = varset($parm['default'], '');
    $value = $themePrefs[$name];
    $value = varset($value, $default);
    return $value;
}
```

Example:

```
{THEME_PREF: name=header_width&default=container}
```


{THEME_PATH}

Normally in theme.html, you can use `{THEME}` shortcode for the path to the frontend theme folder.

But it doesn't work in plugin shortcodes, so I added the custom shortcode in theme shortcodes.

```
/* {THEME_PATH} */  
function sc_theme_path() {  
    $themepath = e_THEME.e107::getPref('sitetheme');  
    return $themepath . "/";  
}
```

AYRO UI STUFF

General info

AYRO UI folder and sass structure is the main reason for the decision to use these UI Components.

Pluses:

- All components are grouped in subfolders and each file has a corresponding sass file.
- All components are displayed on their site, they can be searched etc

Minuses:

- they load separated bootstrap and override it. So they use their own root variables and their own classes (for example buttons: instead of `btn-primary` , they use `primary-btn`)

e107 uses core bootstrap classes, so this is not only about including their components but about converting it for bootstrap itself to minimize using custom HTML markup.

Folder structure

Copy folder assets into the theme and delete scss folder from it

You could copy folder assets/sass into folder sass-starter/ayro-ui, but then you lost an overview of what was implemented and what wasn't. For now, it is enough to know that AYRO UI scss files will be copied to this folder.

Components

Component type	File name	e107 version
navbars	navbar-01.html	{---HEADER---}
navbars	navbar-10.html	{---HEADER---}
headers	header-01.html	welcome message
headers	header-02.html	welcome message
footers	footer-01.html	{---FOOTER---}
footers	footer-02.html	{---FOOTER---}
sliders	slider-01.html	{FEATUREBOX}

How to change navbar

- change file name in theme shortcodes `$header_file = varset($pref, 'navbar-10.html');`
- change scss file in theme.scss

```
//@import 'ayro-ui/navbars/navbar-01';  
@import 'ayro-ui/navbars/navbar-10';
```

- check `getInlineCodes()` in theme.php for inline js code

Navbars

Headers in e107 terminology

PAGE

{NAVIGATION}

>

PAGE

{NAVBAR_SITESEARCH}

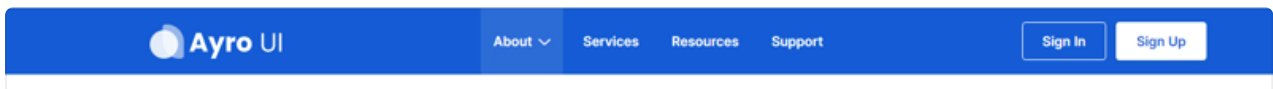
>

PAGE

{SIGNIN}

>

NAVBAR-01



- copy HTML file inside the theme folder (components/navbars/navbar-01.html)
- delete all stuff except the component itself (head, body, script tags)

This is the final result for `navbar-01` component (to see what was left)

```

<!--===== NAVBAR ONE PART START =====-->
<section class="navbar-area navbar-one">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">
        <nav class="navbar navbar-expand-lg">
          {SITELOGO: h=80}
          <button
            class="navbar-toggler"
            type="button"
            data-bs-toggle="collapse"
            data-bs-target="#navbarOne"
            aria-controls="navbarOne"
            aria-expanded="false"
            aria-label="Toggle navigation"
          >
            <span class="toggler-icon"></span>
            <span class="toggler-icon"></span>
            <span class="toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse sub-menu-bar" id="nav
            <ul class="navbar-nav m-auto">
              {NAVIGATION=main}
              {SIGNIN}
            </ul>
          </div>
        </nav>
        <!-- navbar -->
      </div>
    </div>
    <!-- row -->
  </div>
  <!-- container -->
</section>
<!--===== NAVBAR ONE PART ENDS =====-->

```

- customize navigation template
- customize signin template
- add scss file

Result



Adding inline code in theme.php

```
public function getInlineCodes()
{
    //==== close navbar-collapse when a clicked
    $code = 'let navbarTogglerOne = document.querySelector(
        ".navbar-one .navbar-toggler"
    );
    navbarTogglerOne.addEventListener("click", function () {
        navbarTogglerOne.classList.toggle("active");
    });';
    e107::js('footer-inline', $code);
}
```

NAVBAR-10

```

<!--===== NAVBAR TEN PART START =====>

<section class="navbar-area navbar-ten">
  <div class="container">
    <div class="inner-header">
      <div class="row">
        <div class="col-lg-12">
          <nav class="navbar navbar-expand-lg">
            {SITELOGO: h=80}

            <button
              class="navbar-toggler"
              type="button"
              data-bs-toggle="collapse"
              data-bs-target="#navbarEight"
              aria-controls="navbarEight"
              aria-expanded="false"
              aria-label="Toggle navigation"
            >
              <span class="toggler-icon"></span>
              <span class="toggler-icon"></span>
              <span class="toggler-icon"></span>
            </button>

            <div
              class="collapse navbar-collapse sub-menu-bar"
              id="navbarEight"
            >
              <ul class="navbar-nav ms-auto">
                {NAVIGATION=main}
              </ul>
            </div>
            {NAVBAR_SITESEARCH}
          </nav>
          <!-- navbar -->
        </div>
      </div>
    </div>
    <!-- row -->
  </div>
  <!-- container -->
</section>

<!--===== NAVBAR TEN PART ENDS =====>

```




Standards For Our
Themes
Themes HEAD Post

{NAVIGATION}

Main navigation

AYRO UI doesn't support standard bootstrap HTML markup for navbar with dropdowns so the core navigation template can't be used. Maybe it can be fixed with css, but at this point it is not possible.

- copy core navigation template in theme/templates folder

`$NAVIGATION__TEMPLATE['main']` - is usef for main navigation

`$NAVIGATION__TEMPLATE['main']['start'] = ''`; - normally ul tag but it is in navbar component

`$NAVIGATION__TEMPLATE['main']['item']` - item without sublinks

`$NAVIGATION__TEMPLATE['main']['item__active']` - active item without sublinks

`$NAVIGATION__TEMPLATE['main']['item__submenu']` - item with one sublinks (2nd or 3rd level)

`$NAVIGATION__TEMPLATE['main']['item__submenu__active']` = active item with one sublink3 (2nd or 3rd level)

A single item is easy (no sublinks)

```
<li class="nav-item">
  <a href="javascript:void(0)">Services</a>
</li>
```

Result:

```
// Main Link
$NAVIGATION_TEMPLATE['main']['item'] = '
    <li class="nav-item">
        <a href="{NAV_LINK_URL}"{NAV_LINK_OPEN} title="{NAV_LINK_DESCRIP
            {NAV_LINK_ICON}{NAV_LINK_NAME}
        </a>
    </li>
';
$NAVIGATION_TEMPLATE['main']['item_active'] = $NAVIGATION_TEMPLATE['main'
```

The item with sublinks:

```
<li class="nav-item">
    <a
        class="page-scroll active"
        data-bs-toggle="collapse"
        data-bs-target="#sub-nav1"
        aria-controls="sub-nav1"
        aria-expanded="false"
        aria-label="Toggle navigation"
        href="javascript:void(0)"
    >About
    <div class="sub-nav-toggler">
        <span><i class="lni lni-chevron-down"></i></span>
    </div>
    </a>

    <ul class="sub-menu collapse" id="sub-nav1">
        ...
    </ul>
</li>
```

As template:

```

// Main Link that has a sub menu.
$NAVIGATION_TEMPLATE['main']['item_submenu'] = '
<li class="nav-item">
  <a
    class="page-scroll"
    data-bs-toggle="collapse"
    data-bs-target="#sub-nav{NAV_LINK_ID}"
    aria-controls="sub-nav{NAV_LINK_ID}"
    aria-expanded="false"
    aria-label="Toggle navigation"
    href="javascript:void(0)"
    >{NAV_LINK_ICON}{NAV_LINK_NAME}
    <div class="sub-nav-toggler">
      <span><i class="lni lni-chevron-down"></i></span>
    </div>
  </a>
  {NAV_LINK_SUB}
</li>
';

```

The active version just has an active class.

Submenus

HTML code:

```
<ul class="sub-menu collapse" id="sub-nav1">
  <li><a href="javascript:void(0)">About Us</a></li>
  <li>
    <a
      class="page-scroll"
      data-bs-toggle="collapse"
      data-bs-target="#sub-nav2"
      aria-controls="sub-nav2"
      aria-expanded="false"
      aria-label="Toggle navigation"
      href="javascript:void(0)"
    >Our Portfolio
    <div class="sub-nav-toggler">
      <span><i class="lni lni-chevron-down"></i></span>
    </div>
  </a>
  <ul class="sub-menu collapse" id="sub-nav2">
    <li>
      <a href="javascript:void(0)">Portfolio Style 1</a>
    </li>
  </ul>
</li>
</ul>
```

```

// Sub menu start
$NAVIGATION_TEMPLATE['main']['submenu_start'] = '
    <ul class="sub-menu collapse" id="sub-nav{NAV_LINK_ID}" >
';

// Sub sub menu start
$NAVIGATION_TEMPLATE['main']['submenu_lowerstart'] = '
    <ul class="sub-menu collapse" id="sub-nav{NAV_LINK_ID}" >
';

$NAVIGATION_TEMPLATE['main']['submenu_end'] = '</ul>';
$NAVIGATION_TEMPLATE['main']['submenu_lowerend'] = '</ul>';

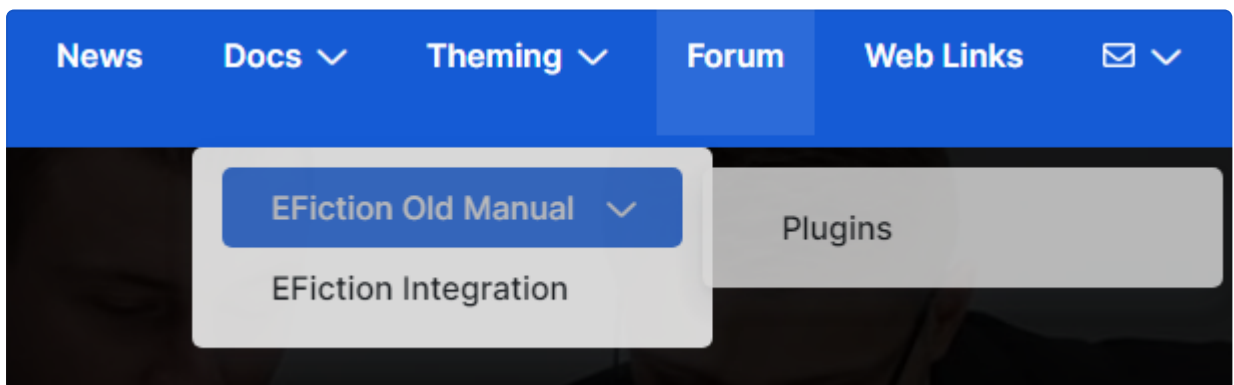
// Sub menu Link
$NAVIGATION_TEMPLATE['main']['submenu_item'] = $NAVIGATION_TEMPLATE['main

// Sub menu Link - active state
$NAVIGATION_TEMPLATE['main']['submenu_item_active'] = $NAVIGATION_TEMPLAT

// Sub Menu Link which has a sub menu.
$NAVIGATION_TEMPLATE['main']['submenu_loweritem'] = $NAVIGATION_TEMPLATE
$NAVIGATION_TEMPLATE['main']['submenu_loweritem_active'] = $NAVIGATION_TE

```

Result



{NAVBAR_SITESEARCH}

custom theme shortcode

```
/* {NAVBAR_SITESEARCH} */  
public function sc_navbar_sitesearch()  
{  
    $text = '<form class="d-flex search-form" method="get" action="'.e_HT  
        <input type="hidden" name="r" value="0" />  
        <input class="form-control me-2" type="text" placeholder="'.LAN_SEARC  
        <button name="s" class="btn btn-outline-success" type="submit"><i cla  
    </form>';  
  
    return $text;  
}
```

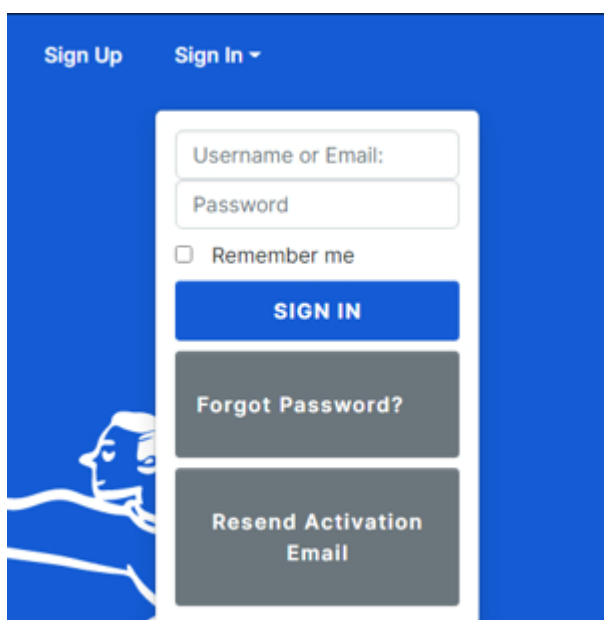
{SIGNIN}

Shortcode from signin plugin, installed by default.

Signin shortcode displays signin/up buttons with login form for quest (signin key in template) and account info for members (signout key). With installed Private Messages plugin it can display Private Message options.

Signin look - buttons

By default (with the core template) it looks like this



The weird look of the buttons is caused by Ayro-ui css. This css rule for navigation changes buttons too.

```
.navbar-one .navbar .navbar-nav .nav-item a { padding: 28px 15px;}
```

A quick fix would be to change it to (but we don't want to change original scss files)

```
.navbar-one .navbar .navbar-nav .nav-item > a { padding: 28px 15px; }
```

Working solution:


```

/*===== Navbar SIGNIN fix start =====*/

$signin_padding: $btn-padding-x $btn-padding-y;

@if $component-navbar == 1 {
  $signin_padding: 8px 24px;
}
@else if $component-navbar == 10 {
  $signin_padding: 8px 18px;
}

.navbar-one .navbar .navbar-nav.signin .nav-item a {
  padding: $signin_padding;
}
/*===== Navbar SIGNIN fix end =====*/

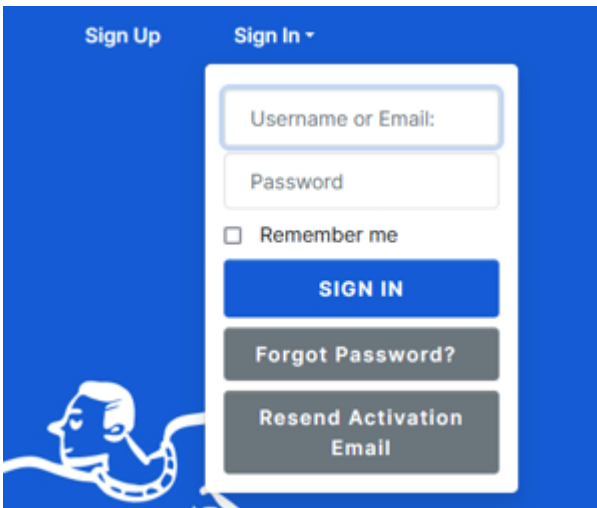
```

Signin look - form

added form scss to theme.scss

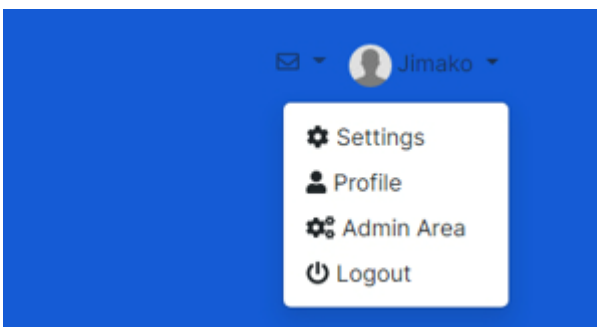
add fix inspired by search box in navbar 10

```
.signin input.tbox {  
  margin-top: 5px;  
  height: 45px;  
  background-color: var(--white);  
  color: var(--black);  
  border: 1px solid #ddd;  
  border-radius: 5px;  
  padding: 0px 20px;  
}
```



Sign out - link color

By default



It looks good but we can try to add more Ayro look. There are 2 ways - changing HTML markup to AYRO UI is used or changing css

```

$signout_navbar-color: var(#{$prefix}nav-link-color);

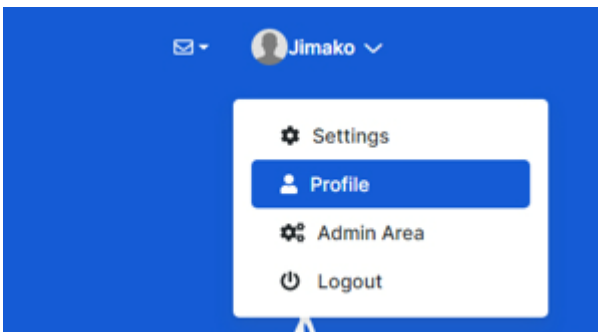
@if $component-navbar == 1 {
  $signout_navbar-color: var(--white);
}
@else if $component-navbar == 10 {
  $signout_navbar-color: var(--black);
}

.pm-nav {
  color: $signout_navbar-color;
}
.dropdown-avatar .nav-link {
  color: $signout_navbar-color;
}

```

Sign out – dropdown

Changing HTML markup in the signin template and some css fixes



Sign out – PM dropdown

Unfortunately, we can't template {SIGNIN_PM_NAV} shortcode. So override it in theme shortcodes



 Jimako 

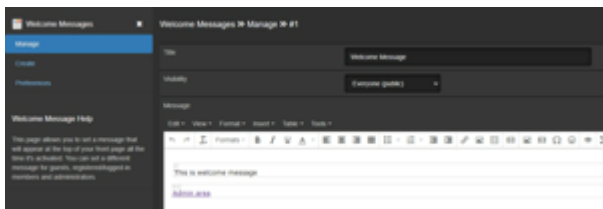
Inbox

Outbox

Send New Message

Headers

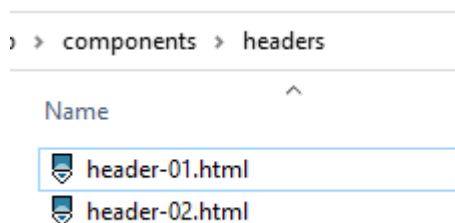
There are 2 header components. e107 has Welcome message functionality for something like this



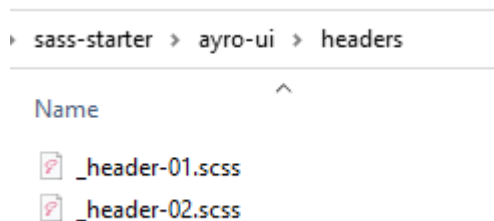
in home layout

```
{SETSTYLE=none} {COMPONENT: type=headers&key=header-01}
```

in components folder



in ayro-ui scss folder



in theme.scss (there is no conflict to load both)

```
@import 'ayro-ui/headers/header-01';  
@import 'ayro-ui/headers/header-02';
```

{SITETITLE}

PAGE

{THEME_PATH}

>

PAGE

{WMESSAGE}

>

HEADER-01

```

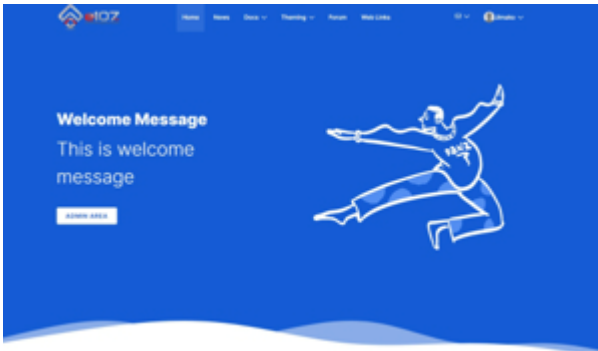
<!--===== HEADER ONE PART START =====>
<section class="header-area header-one">
  <div class="header-content-area">
    <div class="container">
      <div class="row align-items-center">
        <div class="col-lg-6 col-12">
          <div class="header-wrapper">
            <div class="header-content">
              {SETSTYLE=heading-1}
              {WMESSAGE=force}
            </div>
            <!-- header content -->
          </div>
        </div>
        <div class="col-lg-6 col-12">
          <div class="header-image d-none d-lg-block">
            <div class="image">
              
            </div>
          </div>
        </div>
      </div>
    </div>
    <!-- row -->
  </div>
  <!-- container -->
  <div class="header-shape">
    
  <!-- header-shape -->
</div>
<!-- header content area -->
</section>

<!--===== HEADER ONE PART ENDS =====>

```

Notice: {SETSTYLE=heading-1}

```
case "heading-1":
    if (!empty($caption)) {
        echo '<h1 class="header-title">'.$caption.'<h1>';
    }
    echo $text;
break;
```



HEADER-02


```

<!--===== HEADER TWO PART START =====>
<section class="header-area header-two">
  <div class="header-content-area d-flex align-items-center">
    <div class="container">
      <div class="row justify-content-lg-end">
        <div class="col-xl-5 col-lg-6 col-md-8 col-sm-10">
          <div class="header-content">
            {SETSTYLE=heading-3}
            {WMESSAGE=force}
          </div>
          <!-- header content -->
        </div>
      </div>
      <!-- row -->
    </div>
    <!-- container -->

    <div class="header-image d-none d-lg-flex align-items-center">
      <div class="image">
        
      <!-- image -->
    </div>
    <!-- header image -->
  </div>
  <!-- header content area -->
</section>

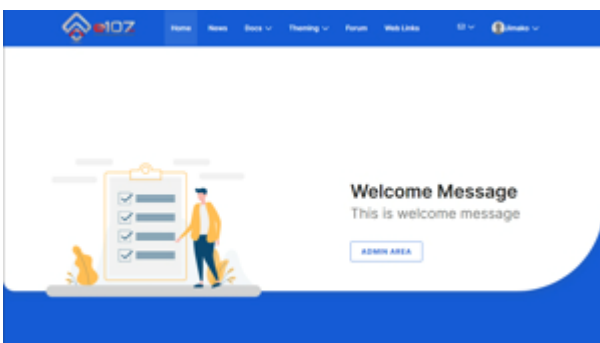
<!--===== HEADER TWO PART ENDS =====>

```

```

case "heading-3":
  if (!empty($caption)) {
    echo '<h3 class="header-title">'.$caption.'</h3>';
  }
  echo $text;
break;

```



{WMESSAGE}

```
{WMESSAGE=force}
```

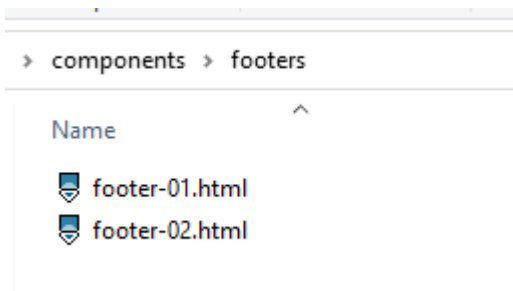
It inserts dynamic content from the Welcome Message functionality

Footers

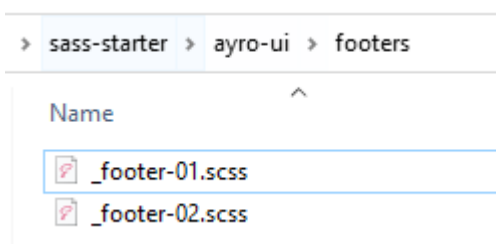
Footers are specific because of menu areas and Menu Manager needs.

The same routine:

- copy HTML components and fix the content (let there only body content)



- copy scss files and delete imported files



- import them

```
@import 'ayro-ui/footers/footer-01';  
@import 'ayro-ui/footers/footer-02';
```

- replace static content with shortcodes

FOOTER-01

```

<!--===== FOOTER ONE PART START =====-->
<footer class="footer-area footer-one">
  <div class="footer-widget">
    <div class="container">
      <div class="row">
        <div class="col-xl-6 col-lg-4 col-sm-12">
          <div class="f-about">
            <div class="footer-logo">
              {SITELOGO: w=400}
            </div>
            <p class="text">
              {SITEDESCRIPTION}
            </p>
          </div>
          <div class="footer-app-store">
            {SETSTYLE=footer}
            {MENUAREA=41}
          </div>
        </div>
        <div class="col-xl-2 col-lg-2 col-sm-4">
          {SETSTYLE=footer-link}
          {MENUAREA=42}
        </div>
        <div class="col-xl-2 col-lg-3 col-sm-4">
          {SETSTYLE=footer-link}
          {MENUAREA=43}
          <!-- footer link -->
        </div>
        <div class="col-xl-2 col-lg-3 col-sm-4">
          <!-- Start Footer Contact -->
          {SETSTYLE=footer-contact}
          {MENUAREA=44}
          <!-- End Footer Contact -->
        </div>
      </div>
    </div>
  <!-- row -->
</div>
<!-- container -->
</div>
<!-- footer widget -->

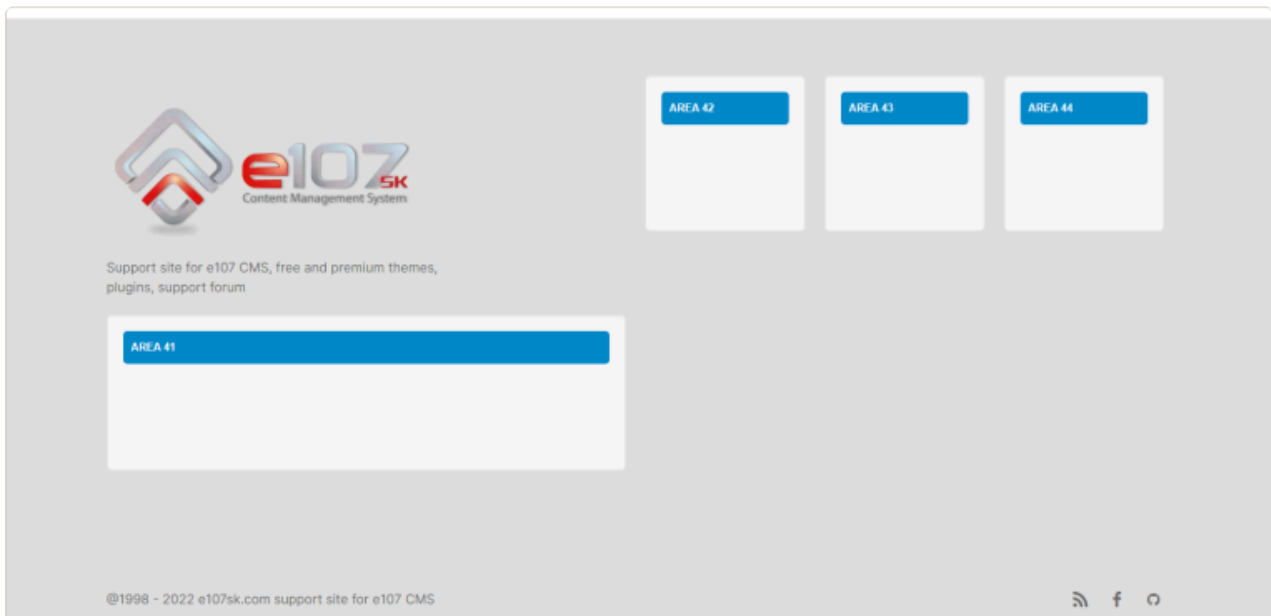
<div class="footer-copyright">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">
        <div class="
copyright
text-center

```

```

        d-md-flex
        justify-content-between
        align-items-center
    ">
        <p class="text">{SITEDISCLAIMER}</p>
        {XURL_ICONS: template=footer-01}
    </div>
    <!-- copyright -->
</div>
</div>
<!-- row -->
</div>
<!-- container -->
</div>
<!-- footer copyright -->
</footer>

```



FOOTER 01 in Menu manager

FOOTER-02

```

<!--===== FOOTER TWO PART START =====-->

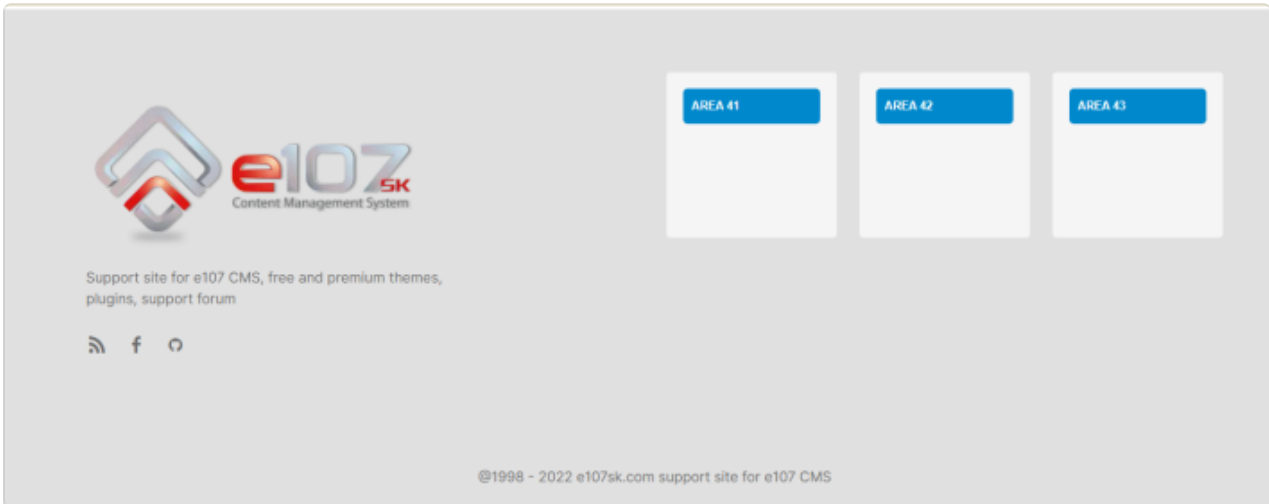
<footer class="footer-area footer-two">
  <div class="footer-widget">
    <div class="container">
      <div class="row">
        <div class="col-xl-6 col-lg-4 col-sm-12">
          <div class="f-about">
            <div class="footer-logo">
              {SITELOGO: w=0}
            </div>
            <p class="text">
              {SITEDESCRIPTION}
            </p>
            {XURL_ICONS: template=footer-01}
          </div>
        </div>
        <div class="col-xl-2 col-lg-2 col-sm-4">
          {SETSTYLE=footer-link}
          {MENUAREA=41}
        </div>
        <div class="col-xl-2 col-lg-3 col-sm-4">
          {SETSTYLE=footer-link}
          {MENUAREA=42}
        </div>
        <div class="col-xl-2 col-lg-3 col-sm-4">
          {SETSTYLE=footer-link}
          {MENUAREA=43}
        </div>
      </div>
    </div>
    <!-- row -->
  </div>
  <!-- container -->
</div>
<!-- footer widget -->

<div class="footer-copyright">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">
        <div class="copyright text-center">
          <p class="text">{SITEDISCLAIMER}</p>
        </div>
        <!-- copyright -->
      </div>
    </div>
  </div>
  <!-- row -->
</div>

```

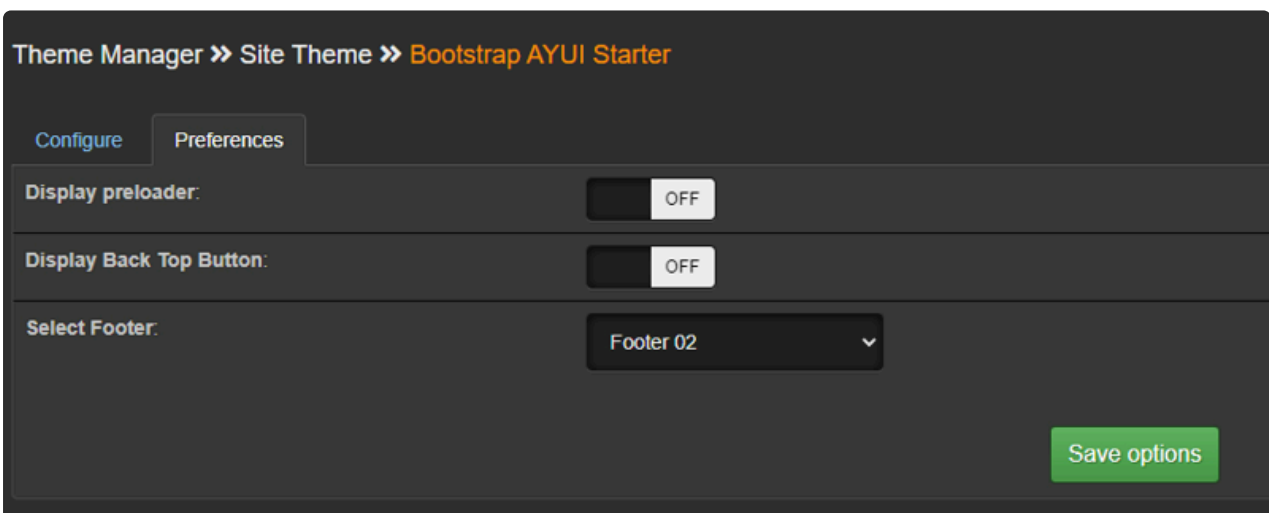
```
<!-- container -->
</div>
<!-- footer copyright -->
</footer>

<!--===== FOOTER TWO PART ENDS =====>
```



FOOTER 02 in Menu manager

For easier testing you can switch between footers in Theme preferences



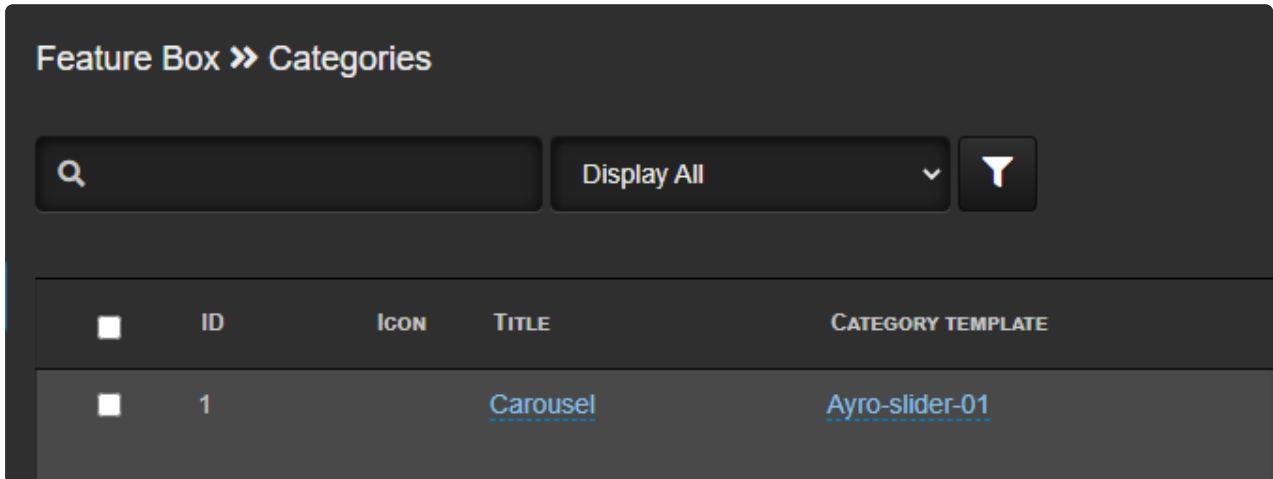
PAGE
{SITELOGO}



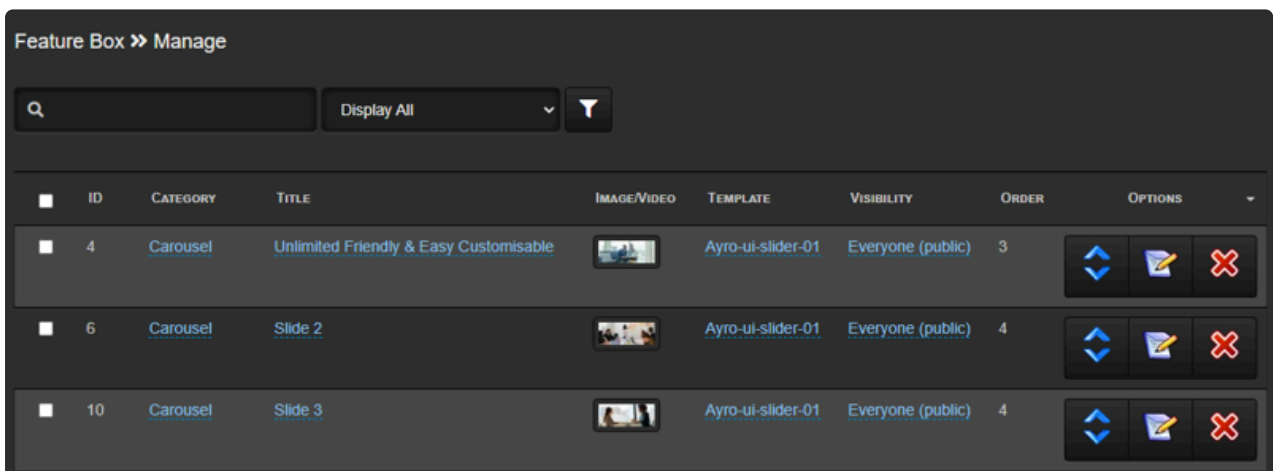
Sliders

Featurebox plugin is used

SLIDER-01



Select correct category template



Select correct category and item template

Final look

Unlimited Friendly & Easy Customisable

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard.

[MORE..](#)

SASS stuff

General info

- All SCSS files are in folder sass-starter
- All AYRO UI files will be in the folder ayro-ui
- Bootstrap 5 scss files are in the folder bootstrap
- Custom or changed AYRO UI files are in folder theme with the same subfolder structure as AYRO UI
- All AYRO UI scss files are renamed with an underscore
- No css and map files in ayro-ui subfolder
- All files (including bootstrap) are compiled to one file theme.css (for now)

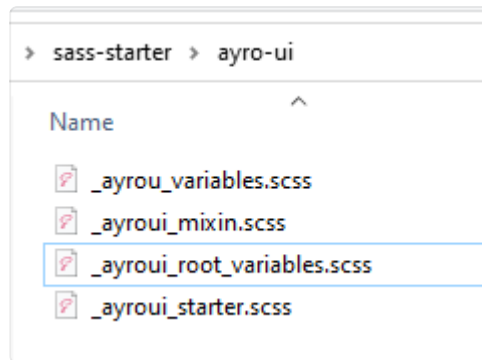
starter.scss

starter.scss file is the main scss file. It looks like this:

```
1  @import url("https://fonts.googleapis.com/css2?family=Inter:wght@200;300;400;500;600;700;800;900&display=swap")
2
3  @import "./variables";
4  @import "./mixin";
5
6  /*-----
7  | -COMMON css-
8  |-----*/
9
10 > :root-{ ...
66 }
67
68 body-{
69   --font-family: var(--font-family);
70   --color: var(--black);
71   --font-size: 16px;
72
73   @media (max-width: 991px){
74     --font-size: 14px;
75   }
76 }
```

starter,scss

- the font is loaded in theme.php, so it was deleted
- variables and mixin files renamed with prefix (they have nothing with bootstrap variables)
- root variables moved to separated file, this way:



•

theme.scss

Starter version

```
$prefix: '';  
  
@import "bootstrap5/functions";  
  
//AYRO UI files, no change, just renamed  
@import 'ayro-ui/ayroui_variables';  
@import 'ayro-ui/ayroui_mixin';  
  
// Bootstrap core  
@import "bootstrap5/variables";  
@import "bootstrap5/maps";  
@import "bootstrap5/mixins";  
@import "bootstrap5/utilities";  
  
// Layout & components  
@import "bootstrap5/root";  
@import "bootstrap5/reboot";  
  
@import 'ayro-ui/ayroui_root_variables'; //overrides bootstrap root if s  
  
@import 'theme/custom_variables'; //new variables for theme  
  
@import 'bootstrap5/bootstrap_custom'; //what left from bootstrap.scss,  
  
// custom stuff  
@import 'ayro-ui/ayroui_starter'; //custom css
```

The result is compiled bootstrap with starter css

if you don't want AYRO UI to override bootstrap root variables, delete `$prefix: '';`. (bootstrap then will use `bs_` as the prefix and they will be different than those from Ayro)

Example of how to combine AYRO + bootstrap:

Use bootstrap variable:

```
$primary: #155bd5;
```

use it AYRO root variable:

```
--primary: #{ $primary };
```

This way bootstrap primary stuff will have the same color as Ayro does.

Buttons

AYRO UI version

in theme.scss

```
@import 'ayro-ui/buttons/buttons';  
@import 'ayro-ui/buttons/button-page';
```

Customization

....

Adding component scss

This applies for all components

- copy scss file into ayro-ui folder
- rename it with underscore
- open it and delete imported files (check if there is not some new file, in this case buttons)

```
@import "../variables";  
@import "../mixin";  
@import "../buttons/buttons";
```

- add this file to theme.scss