

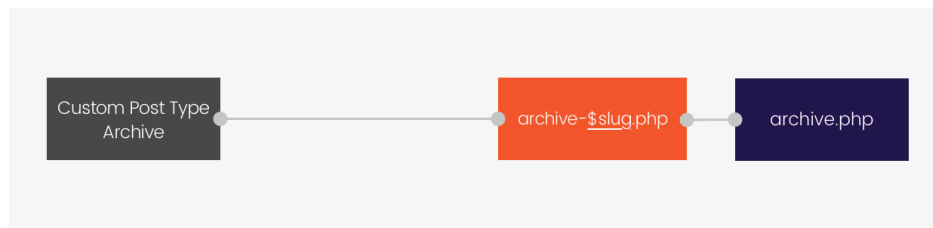


## **Development of a dynamic web site (DWD)**

**Class 11**

# Assigning custom templates to custom post types

There are situations when it is needed to use a different template than *archive.php* to create a custom post type such as the portfolio ones we previously made. Based on the template hierarchy (see the figure next page), a specific dedicated template will have to be produced based on the following pattern :



## Creating the custom template

So, in order to create a custom template to be used on the portfolio page we previously wanted, two templates are needed : one for the archive and one for the complete articles (the singles). The syntax of these templates must be : **archive-YourCustomFileName.php** and **single-YourCustomFileName.php**

Let's then create two files : **archive-portfolio.php** and **single-portfolio.php**. You can use the basic templates code to create yours, keeping the WordPress loop and simply editing the HTML. See the example of what could be the portfolio archive template below.

```

<?php get_header(); ?>

<h1 class="site__heading"><?php post_type_archive_title(); ?></h1>

<main class="site__portfolio">
    <?php if( have_posts() ) : while( have_posts() ) : the_post(); ?>

        <div class="project">
            <h2 class="project__title">
                <a href="<?php the_permalink(); ?>">
                    <?php the_title(); ?>
                </a>
            </h2>
            <?php the_post_thumbnail(); ?>
        </div>

    <?php endwhile; endif; ?>
</main>

<?php the_posts_pagination(); ?>
<?php get_footer(); ?>
  
```



# Conditional tags for custom post types

## Conditional structure

In order to verify if a current page uses a specific custom archive template made for custom post types, *is\_archive()* won't be used like we previously did. Instead, the function *is\_post\_type\_archive()* here needs to be used like shown in the example below.

```
<?php if( is_post_type_archive( 'portfolio' ) ) {  
    // specific instructions to execute  
}
```

To verify if a current page uses a single template made for custom post types, the function *is\_single()* won't be used. Instead, use the function *is\_singular()*.

```
<?php if( is_singular( 'portfolio' ) ) {  
    // specific instructions to execute  
}
```

# Sidebars and widgets

## Widgets

Widgets are dynamic elements that can present various complementary contents such as a list of the most recent posts or comments, categories, monthly archives, image galleries, RSS feed, plug-ins, etc. They are usually located in sidebars.

## Sidebars

Sidebars are where complementary contents are located. They are basically the equivalent of HTML5's `<aside>` tag.

## Declaring and displaying sidebars into templates

The first thing to do to declare the sidebar by adding the function ***register\_sidebar()*** into the *functions.php* file.

```
<?php
register_sidebar( array(
    'id' => 'blog-sidebar',
    'name' => 'Blog',                // Name showing in the admin interface
) );
```

Once the sidebar has been declared, it is now possible to easily add it to *archive.php* file using the function ***dynamic\_sidebar()***. The example below shows the minimal way to display a sidebar. Note that it must be wrapped within `<ul></ul>` since widgets are using `<li>`.

```
<?php get_header(); ?>

<h1 class="site__heading">Le blog</h1>
<div class="site__blog">
    <main class="site__content">
        <?php if( have_posts() ) : while( have_posts() ) : the_post(); ?>
            <article class="post">
                <!-- ... -->
            </article>
        <?php endwhile; endif; ?>
    </main>
    <aside class="site__sidebar">
        <ul>
            <?php dynamic_sidebar( 'blog-sidebar' ); ?>
        </ul>
    </aside>
</div>

<?php get_footer(); ?>
```

## Customizing sidebars declaration

If you don't like the `<ul>` and `<li>` structure, it is possible to change a few parameters in the declarations (functions.php).

```
<?php
register_sidebar( array(
    'id' => 'blog-sidebar',
    'name' => 'Blog',
    'before_widget' => '<div class="sidebar_widget %2$s">',
    'after_widget' => '</div>',
    'before_title' => '<p class="sidebar_widget_title">',
    'after_title' => '</p>',
));
```

### Explanation :

In the example above, `<div>` replaces `<li>`, so `<ul>` won't be necessary in the archive.php file anymore, and a custom class is added. `%2$s` let WordPress add other classes (each widget has a specific class). Also, `<h2>` headers are replaced with simple `<p>` tags.

## Adding widgets to sidebars declaration

Once sidebar has been declared and integrated, it becomes possible to add widgets to it. To do so, it is needed to go to **Dashboard > Appearance > Widgets**. Simply drag and drop the wanted widget to the sidebar, then use CSS to position and style it properly.

