**Basic Theme Directory Structure**

// Theme CSS

style.css

// Custom functionality for your theme

functions.php

// Partials to include in subsequent theme files

header.php

footer.php

sidebar.php

comments.php

// "Archives", (listing views that contain multiple posts)

archive.php

author.php

date.php

taxonomy.php

tag.php

category.php

// Individual content pages

// Note that home and frontpage templates are not recommended

// and they should be replaced by page templates

singular.php

single.php

page.php

front-page.php

home.php

// Misc. Utility Pages

index.php (a catch-all if nothing else matches)

search.php

attachment.php

image.php

404.php

**Theme folder and file structure**[**#**](https://developer.wordpress.org/themes/basics/organizing-theme-files/#theme-folder-and-file-structure)

assets (dir)

      - css (dir)

      - fonts (dir)

      - images (dir)

      - js (dir)

core (dir)

inc (dir)

languages (dir)

template-parts (dir)

      - footer (dir)

      - header (dir)

      - navigation (dir)

      - page (dir)

      - post (dir)

      - other (dir)

404.php

archive.php

author.php

category.php

comments.php

content.php

content-aside.php

content-featured.php

content-gallery.php

content-image.php

content-intro.php

content-link.php

content-page.php

content-quote.php

content-single.php

content-status.php

content.php

footer.php

front-page.php

functions.php

header.php

index.php

image.php

page.php

page-bothsidebar.php

page-leftsidebar.php

page-rightsidebar.php

page-portfolio-template.php

page-test.php

README.txt

rtl.css

editor-style-rtl.css

editor-style.css

screenshot.png

search.php

searchform.php

showcase.php

sidebar.php

sidebar-footer.php

sidebar-page.php

single.php

style.css

tag.php

single-portfolio.php

single-test.php

settings.php

language - <https://developer.wordpress.org/themes/functionality/internationalization/>

**Structure**

<https://github.com/seothemes/genesis-starter-theme>

<https://wpsites.net/genesis-hooks/>

your-theme-name/ # → Root directory

├── admin/ # → load admin functions

├── assets/ # → Front-end assets

│ ├── css/ # → css

│ ├── fonts/ # → font

│ ├── images/ # → images

│ ├── js/ # → java scripts

│ └── externalframework/ # → load external css or script frameworks

│ └── bootstrap.css # → theme options functions

│ └── bootstrap.js # → theme options functions

├── helperclasses/ # → common class

│ └── htmlhelper.php # → helper functions for dynamic html tags

│ └── phphelper.php # → helper functions for php functions

├── core/ # → register all support items

│ └── customtypes-register.php # → register custom type

│ └── thememeta-register.php # → register meta boxes

**├── includes/ # → common class**

**│ └── themeoptions.php # → theme options functions**

├── config/ # → Config directory

├── lib/ # → Theme functions

│ ├── functions/ # → General functions

│ ├── plugins/ # → Plugin functions

│ ├── shortcodes/ # → Shortcode functions

│ ├── structure/ # → Structural functions

│ └── init.php # → File autoloader

├── templates/ # → Page templates

│ ├── header/ # → different header layouts

│ ├── footer/ # → different footer layouts

│ ├── navigation/ # → different navigation layouts

│ ├── page/ # → different page payouts

│ ├── post/ # → different post layouts

│ └── others/ # → different post layouts

│ └── contact.php # → theme options functions

│ └── about-us.php # → theme options functions

├── partials/ # → Page templates

│ ├── content.php # → custom page type partially load

│ ├── content-single.php # → custom page type single partially load

~~├── tests/ # → PHP Unit tests~~

~~├── vendor/ # → Composer packages~~

~~├── node\_modules/ # → Node.js packages~~

├── composer.json # → Composer settings

├── package.json # → Node dependencies

~~├── webpack.mix.js # → Laravel mix config~~

├── screenshot.png # → Theme screenshot

**├── functions.php # → Loads init files**

**├── style.css # → Blank stylesheet**├

**├── index.php # → default fallback file**

**├── header.php # → default header**

**├── footer.php # → default footer**

├── front-page.php # → **home page design**

**├── page.php # → default template for page (posttype-> page)**

├── page-sidebar.php # → sidebar design for page

├── page-customposttype-template.php # → design template for custom post

├── search.php # → for search

├── searchform.php # → for search form design

├── sidebar.php # → common sidebar design for page or post

**├── single.php # → single page for default post (posttype-> post)**

**├── single-customposttype.php # → single page for custom post type**

**├── functions-loader.php # → use instead of functions.php**

├

├── archive.php # → for comments

├── comments.php # → for comments

└── rtl.css # → stylesheet for right to left alignment

## Basic WordPress Page Hierarchy

1. index.php
2. header.php
3. footer.php
4. sidebar.php

## Homepage Display Pages

1. front-page.php
2. home.php
3. page.php
4. index.php

## Single Post Pages

1. single.php
2. singular.php

## Static Pages

Pages are rendered in the following order:

1. page templates
2. page-{slug}.php
3. page-{id}.php
4. page.php
5. singular.php
6. index.php

As stated above, if we want to have a similar look for single.php and page.php then it is better to create a singular.php.

## Archive Pages

1. author.php
2. category.php
3. taxonomy.php
4. date.php
5. tag.php
6. archive.php

Archive pages in WordPress are those pages that are used to grab posts from specific authors, categories, taxonomies, dates, tags and so on.

We can have a single archive template that is, archive.php. But to drill down into more templates we have author.php, category.php, taxonomy.php, date.php, tag.php and all of them are pretty self-explanatory.

We can further make customized templates, for e.g.

category-{slug}.php – if the slug of a category is cute-kittens, then WordPress will look for category-cute-kittens.php, if it doesn’t exist, category.php will be used.

category-{id}.php – if id = 3, then WordPress will look for category-3.php to render the page. Else it will load category.php

Similar theory of {slug} and {id} applies to “Tag” page

For custom Author templates we can use author-{nicename}.php or author-{id}.php. If the author’s name is Andy, then WordPress will look for author-andy.php, if it doesn’t exist, then WordPress will fall back to author.php to render the page.

## Other WordPress Template Files

– search.php

Search results in WordPress use the template file search.php. If it doesn’t exist, then search results are rendered from index.php.

– attachment.php

This template file is used to render attachment pages, such as images and videos. image.php and video.php are used to render images and videos respectively. If these files don’t exist then attachment.php is used.

– 404.php

Not found pages in WordPress are rendered from 404.php. If the 404 doesn’t exist, then it renders the page from index.php.

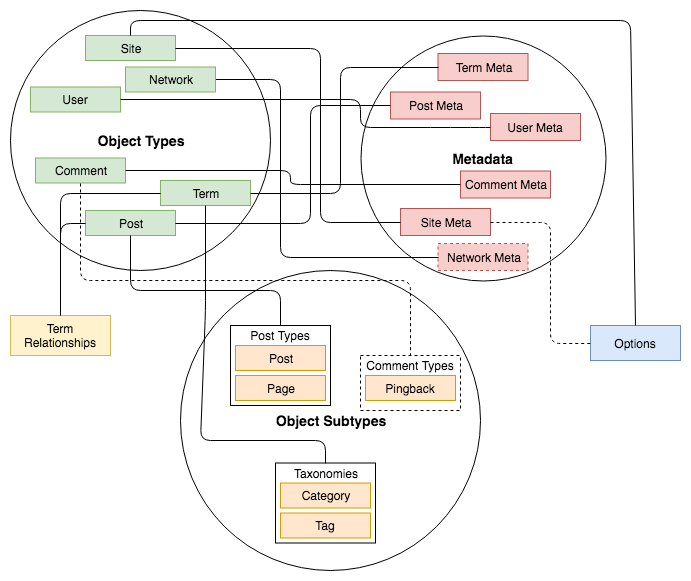
– comments.php

It is a template of comments, it is called in template files like single.php or page.php to add the comments section.

<https://felix-arntz.me/blog/types-subtypes-meta-options-data-structures-in-wordpress/>

|  | **comment** | **post** | **term** | **user** | **site** | **network** |
| --- | --- | --- | --- | --- | --- | --- |
| **class** | WP\_Comment | WP\_Post | WP\_Term | WP\_User | WP\_Site | WP\_Network |
| **retrieval** | get\_comment() | get\_post() | get\_term() | get\_user() | get\_site() | get\_network() |
| **addition** | wp\_insert\_comment() | wp\_insert\_post() | wp\_insert\_term() | wp\_insert\_user() | wp\_insert\_site() 1 | wp\_insert\_network() 2 |
| **modification** | wp\_update\_comment() | wp\_update\_post() | wp\_update\_term() | wp\_update\_user() | wp\_update\_site() 1 | wp\_update\_network() 2 |
| **deletion** | wp\_delete\_comment() | wp\_delete\_post() | wp\_delete\_term() | wp\_delete\_user() | wp\_delete\_site() 1 | wp\_delete\_network() 2 |
| **querying** | get\_comments() | get\_posts() | get\_terms() | get\_users() | get\_sites() | get\_networks() |
| **query class** | WP\_Comment\_Query | WP\_Post\_Query | WP\_Term\_Query | WP\_User\_Query | WP\_Site\_Query | WP\_Network\_Query |
| **subtypes** | kinda | yes (post types) | yes (taxonomies) | no | no | no |
| **metadata** | yes | yes | yes | yes | yes 1 | kinda |

**wordpress-data-structures-relationships**



**WHICH TEMPLATE FILE WILL WORDPRESS USE?**

THE TEMPLATE HEIRARCHY

**PAGE TYPE tries first > tries next > tries last**

404 404.php >index.php

**SEARCH search.php >index.php**

TAXONOMY taxonomy-{tax}-{term}.php > taxonomy-{tax}.php > taxonomy.php > archive.php > index.php

**HOME home.php >index.php**

ATTACHMENT {mime-type}.php >attachment.php >single.php >index.php

**SINGLE single-{post-type}.php >single.php >index.php**

PAGE {custom-template}.php > page-{slug}.php > page-{id}.php > page.php > index.php

**CATEGORY category-{slug}.php > category-{id}.php > category.php > archive.php > index.php**

TAG tag-{slug}.php > tag-{id}.php >tag.php >archive.php >index.php

**AUTHOR author-{nicename}.php > author-{id}.php > author.php > archive.php > index.php**

DATE date.php >archive.php >index.php

**ARCHIVE archive.php >index.php**

/\*\*\*\*\*\*\*\*\*\*\*/

/\*\*MyFirstTheme's functions and definitions

\* @package MyFirstTheme \* @since MyFirstTheme 1.0 \*/

/\*\*\* First, let's set the maximum content width based on the theme's design and stylesheet.

\* This will limit the width of all uploaded images and embeds. \*/

if ( ! isset( $content\_width ) )

$content\_width = 800; /\* pixels \*/

if ( ! function\_exists( 'myfirsttheme\_setup' ) ) :

/\*\* \* Sets up theme defaults and registers support for various WordPress features.

\*

\* Note that this function is hooked into the after\_setup\_theme hook, which runs

\* before the init hook. The init hook is too late for some features, such as indicating

\* support post thumbnails.

\*/

function myfirsttheme\_setup() {

/\* Make theme available for translation.

\* Translations can be placed in the /languages/ directory. \*/

load\_theme\_textdomain( 'myfirsttheme', get\_template\_directory() . '/languages' );

/\* Add default posts and comments RSS feed links to <head>. \*/

add\_theme\_support( 'automatic-feed-links' );

/\* Enable support for post thumbnails and featured images. \*/

add\_theme\_support( 'post-thumbnails' );

add\_image\_size( 'featured-image-large1', 640, 294, true );

add\_image\_size( 'featured-image-small1', 200, 129, true );

/\* Add support for two custom navigation menus. \*/

register\_nav\_menus( array(

'primary' => \_\_( 'Primary Menu', 'myfirsttheme' ),

'secondary' => \_\_('Secondary Menu', 'myfirsttheme' )

) );

/\* Enable support for the following post formats:

\* aside, gallery, quote, image, and video \*/

add\_theme\_support( 'post-formats', array ( 'aside', 'gallery', 'quote', 'image', 'video' ) );

}

endif; // myfirsttheme\_setup

add\_action( 'after\_setup\_theme', 'myfirsttheme\_setup' );

**//Step1. Include below in theme**

add\_theme\_support('menus');

add\_theme\_support('post-thumbnails');

/\*\*\*\*\*Syntax\*\*\*\*

add\_theme\_support( string $feature, needs the feature for specific formats in array )

<https://www.daddydesign.com/wordpress/how-to-add-features-in-wordpress-using-add_theme_support-function/>

**Post Thumbnails**

Add one or more of these features to enable support for post thumbnails.

**EVERYWHERE**

add\_theme\_support( 'post-thumbnails' );

**ONLY PAGES**

add\_theme\_support( 'post-thumbnails', array( 'page' ) );

**ONLY POSTS**

add\_theme\_support( 'post-thumbnails', array( 'post' ) );

**ONLY CUSTOM POST TYPES**

add\_theme\_support( 'post-thumbnails', array( 'your-post-type-name' ) );

**Custom Background**

Add this feature in order to enable support for custom backgrounds.

add\_theme\_support( 'custom-background' );

**Custom Header**

Add this feature in order to enable support for custom headers.

add\_theme\_support( 'custom-header' );

**Custom Logo**

Add this feature in order to enable support for a custom logo.

add\_theme\_support( 'custom-logo' );

**Feed Links**

Add this feature in order to enable support for automatic feed links for post and comment in the head.

add\_theme\_support( 'automatic-feed-links' );

**HTML5**

Add this feature in order to enable support for the usage of HTML5 markup for the search forms, comment forms, comment lists, gallery, and caption.

add\_theme\_support( 'html5', array( 'comment-list', 'comment-form', 'search-form', 'gallery', 'caption' ) );

**Title Tag**

Add this feature in order to enable plugins and themes to manage the document title tag.

add\_theme\_support( 'title-tag' );

**Customize Selective Refresh Widgets**

Add this feature in order to enable selective refresh for widgets being managed within the Customizer.

**view sourceprint?**

add\_theme\_support( 'customize-selective-refresh-widgets' );

**// 2. Create and register custom menu**

function register\_theme\_menus(){

register\_nav\_menus( array('primary-menu'=> \_('Primary Menu'), 'top-menu'=> \_('Top Menu'), 'main-menu'=> \_('Main Menu'), 'footer-menu'=> \_('Footer Menu') ));

}

add\_action('init','register\_theme\_menus');

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Example Register and show menu\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// SYNTAX

register\_nav\_menus( array( $location => $description ) ); // Array of Menus

register\_nav\_menu( 'primary', \_\_( 'Primary Menu', 'theme-slug' ) ); // Single Menu

register\_nav\_menus( array(

'main-menu' => \_\_( 'Main Menu', 'mytheme' ),

'primary-menu' => \_\_( 'Primary Menu', 'mytheme' ),

) );

////////HTML to show?>

<header class="header">

<nav class="nav">

<?php

if ( has\_nav\_menu( 'main' ) ) {

wp\_nav\_menu( array( 'theme\_location' => **'main-menu'**, 'container' => '', 'menu\_class' => 'navigation-main',));

}

?>

</nav>

</header>

<?php

wp\_nav\_menu( array $args = array(

'menu' => "", // (int|string|WP\_Term) Desired menu. Accepts a menu ID, slug, name, or object.

'menu\_class' => "", // (string) CSS class to use for the ul element which forms the menu. Default 'menu'.

'menu\_id' => "", // (string) The ID that is applied to the ul element which forms the menu. Default is the menu slug, incremented.

'container' => "", // (string) Whether to wrap the ul, and what to wrap it with. Default 'div'.

'container\_class' => "", // (string) Class that is applied to the container. Default 'menu-{menu slug}-container'.

'container\_id' => "", // (string) The ID that is applied to the container.

'fallback\_cb' => "", // (callable|bool) If the menu doesn't exists, a callback function will fire. Default is 'wp\_page\_menu'.

Set to false for no fallback.

'before' => "", // (string) Text before the link markup.

'after' => "", // (string) Text after the link markup.

'link\_before' => "", // (string) Text before the link text.

'link\_after' => "", // (string) Text after the link text.

'echo' => "", // (bool) Whether to echo the menu or return it. Default true.

'depth' => "", // (int) How many levels of the hierarchy are to be included. 0 means all. Default 0.

'walker' => "", // (object) Instance of a custom walker class.

'theme\_location' => "", // (string) Theme location to be used. Must be registered with

register\_nav\_menu() in order to be selectable by the user.

'items\_wrap' => "", // (str) How the list items should be wrapped. Default is a ul with an id and class. Uses printf() format

with numbered placeholders.

'item\_spacing' => "", // (string) Whether to preserve whitespace within the menu's

HTML. Accepts 'preserve' or 'discard'. Default 'preserve'.

) );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**// 2.1 wp\_nav\_menu shortcode with attributes**

**function** list\_menu($atts, $content = **null**) {

[**extract**](http://www.php.net/extract)(shortcode\_atts([array](http://www.php.net/array)(

'menu' => '',

'container' => 'div',

'container\_class' => '',

'container\_id' => '',

'menu\_class' => 'menu',

'menu\_id' => '',

'echo' => **true**,

'fallback\_cb' => 'wp\_page\_menu',

'before' => '',

'after' => '',

'link\_before' => '',

'link\_after' => '',

'depth' => 0,

'walker' => '',

'theme\_location' => ''),

$atts));

return wp\_nav\_menu( [array](http://www.php.net/array)(

'menu' => $menu,

'container' => $container,

'container\_class' => $container\_class,

'container\_id' => $container\_id,

'menu\_class' => $menu\_class,

'menu\_id' => $menu\_id,

'echo' => **false**,

'fallback\_cb' => $fallback\_cb,

'before' => $before,

'after' => $after,

'link\_before' => $link\_before,

'link\_after' => $link\_after,

'depth' => $depth,

'walker' => $walker,

'theme\_location' => $theme\_location));

}

*//Create the shortcode*

add\_shortcode("listmenu", "list\_menu");

e.g:

[listmenu menu=Sitemap menu\_class=sitemap\_menu]

**//3.Include necessary files**

if ( is\_admin() ) require\_once( TEMPLATEPATH . '/inc/test.php' );

require\_once dirname( \_\_FILE\_\_ ) . '/inc/theme\_testsettings.php';

**//4.Enqueue necessary files**

function x\_scripts() {

if ( is\_admin() ) {

wp\_enqueue\_style('theme\_default', get\_template\_directory\_uri().'/css/default.css' );

wp\_enqueue\_script('theme\_footer', get\_template\_directory\_uri().'/jss/script.js' );

}}

add\_action( 'wp\_enqueue\_scripts', 'x\_scripts' ); **or**

add\_action( 'admin\_print\_styles-appearance\_page\_{pagename}', 'x\_scripts', 11 );

**Loading scripts on WordPress**

<https://mycyberuniverse.com/loading-scripts-on-wordpress-admin-pages.html>

**1.** Load scripts everywhere on your website:

function enqueue\_my\_scripts() {

// ENQUEUE SCRIPTS…

add\_action( 'wp\_enqueue\_scripts', 'enqueue\_my\_scripts' );

**2.** Load scripts on all Admin Area pages:

function enqueue\_my\_scripts() {

// ENQUEUE SCRIPTS…

add\_action( '**admin\_enqueue\_scripts'**, 'enqueue\_my\_scripts' );

;

**3.** Load scripts on a specific page of the top level menu in Admin Area (in example edit.php - “Posts”):

function enqueue\_my\_scripts($hook) {

if ( 'edit.php' != $hook ) {

return;

// ENQUEUE SCRIPTS…

add\_action( 'admin\_enqueue\_scripts', 'enqueue\_my\_scripts' );

**4.** Load scripts on a specific page of the sub menu in Admin Panel.

function enqueue\_my\_scripts($hook) {

if ( 'appearance\_page\_my-page' != $hook ) {

return;

}

// ENQUEUE SCRIPTS…

}

add\_action( 'admin\_enqueue\_scripts', 'enqueue\_my\_scripts');

In this example are uses the settings page that located in a sub menu of the “Appearance” section of menu and this page having the menu-slug my-page.

For use this method you need to identify the name of page. The page name which seen in the address bar is not valid for this.

The name of the page in “Appearance” section can be identified by the following scheme:

themes.php?page={page menu-slug} => appearance\_page\_{page menu-slug}

The appearance\_page\_ is a prefix for pages in this section of menu. You need to add the prefix to menu-slug. So, if the address to page you want is themes.php?page=my-page.php, then the name will be appearance\_page\_my-page. With other sections of Admin Area do the same, but with different prefix. For “Settings” section of menu the prefix is settings\_page\_. If you can’t find the correct name of your plugin’s page then make do the follow.

Temporary add this function to your plugin’s main file:

function enqueue\_my\_scripts($hook) {

echo "<p style='text-align:center;'>" .$hook. "</p>";

}

add\_action('admin\_enqueue\_scripts', 'enqueue\_my\_scripts');

After that you’ll see a text line under top admin bar. There will be written the correct name of your plugin’s page. You can copy it and then delete this function.

**//5. Create theme's Admin submenu**

**// 5.1 Syntax for add into theme Admin menu and sub menu**

add\_menu\_page( str $page\_title, str $menu\_title, str $capability, str $menu\_slug, callable $func = '', str $icon\_url = '', int $position = null)

add\_menu\_page( $page\_title, $menu\_title, $capability, $parentmenu\_slug, $function, $icon\_url, $position );

add\_submenu\_page( str $parent\_slug, str $page\_title, str $menu\_title, str $capability, str $menu\_slug, callable $function = '' )

add\_menu\_page( 'menu\_page\_title', 'menu\_title', 'menu\_capability', 'menu\_slug', 'menu\_callback');

add\_submenu\_page( 'menu\_slug', 'submenu\_page\_title', 'submenu\_title', 'submenu\_capability', 'submenu\_slug', 'submenu\_callback' );

function theme\_forms\_menu()

{

//function admin submenu welcome page

add\_menu\_page('PageTitle', 'Xtheme', 'edit\_posts', 'x\_test', 'xopt', 'dashicons-star-half',3);

//function admin submenu welcome page

add\_submenu\_page('x\_test', 'Test', 'Test', 'edit\_posts', 'x\_test', 'xopt\_test');

add\_submenu\_page('x\_test', 'Register', 'Register', 'edit\_posts', 'x\_register', 'xopt\_reg');

add\_submenu\_page('x\_test', 'Support', 'Support', 'edit\_posts', 'xtheme\_supp', 'xopt\_support');

add\_submenu\_page('x\_test', 'FAQ', 'FAQ', 'edit\_posts', 'xtheme\_faq', 'xopt\_faq');

add\_submenu\_page('x\_test', 'Demos', 'Demos', 'edit\_posts', 'xtheme\_demos', 'xopt\_demos');

add\_submenu\_page('x\_test', 'Plugins', 'Plugins', 'edit\_posts', 'xtheme\_plugins', 'xopt\_plug');

add\_submenu\_page('x\_test', 'System Status', 'System Status', 'edit\_posts', 'func1', 'func1');

add\_submenu\_page('x\_test', 'Theme Options', 'Theme Options', 'edit\_posts', 'theme-opts', 'fu1');

//function admin submenu theme options page with fallback function 'baltothemeoptions\_custom'

add\_submenu\_page('x\_test', 'Title', 'Title', 'edit\_posts', m\_slug', 'fallback\_function');

}

add\_action('admin\_menu', 'theme\_forms\_menu');

**// 5.2 Syntax for add into theme Appearance sub menu**

# add\_theme\_page( *str* $page\_title, *str* $menu\_title, *str* $capability, *str* $menu\_slug, *callable* $function = '' )

add\_theme\_page( 'Settings', 'Settings', 'edit\_theme\_options', menu\_slug', 'test\_func’ );

function add\_theme\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function = '' ) {

return add\_submenu\_page('themes.php', $page\_title, $menu\_title, $capability, $menu\_slug, $func);

}

<https://hotexamples.com/examples/-/-/add_theme_page/php-add_theme_page-function-examples.html>

function theme\_forms\_menu1()

{ // add some pages

add\_options\_page('Test Settings', 'Test Settings', 'manage\_options', 'testsettings', func1');

add\_management\_page('Test Tools', 'Test Tools', 'manage\_options', 'testtools', 'mt\_tools\_page');

add\_menu\_page('TestToplevel', 'TestToplevel', 'manage\_options', 'top-level-h', 'toplevel\_page');

add\_submenu\_page('top-level-h', 'Test Sublevel', 'Test Sublevel', 'manage\_options', 'sub-page', 'sublevel\_page');

add\_submenu\_page('top-level-h', 'Test Sublevel2', 'Test Sublevel2', 'manage\_options', 'sub-pg2', 'sublevel\_pg2');

add\_theme\_page('With Spaces', 'With Spaces', 'manage\_options', 'With Spaces', 'mt\_tools\_page');

add\_pages\_page('Appending Qry Arg', 'Test Pages', 'edit\_pages', 'testpages', 'mt\_pages\_page');

}

add\_action('admin\_menu', 'theme\_forms\_menu1');

/\* ===========================================\*

**\* 6. #Custom Post Type - Default and Static**

\* =========================================== \*/

function custom\_postype(){

$labels=array(

'name'=>'Portfolio',

'singular\_name'=>'Portfolio',

'add\_new'=>'Add Item',

'all\_items'=>'All Items',

'add\_new\_item'=>'Add Item',

'edit\_item'=>'Edit Item',

'new\_item'=>'New Item',

'view\_item'=>'View Item',

'search\_item'=>'Search Portfolio',

'not\_found'=>'No items found',

'not\_found\_in\_trash'=>'No items found in trash',

'parent\_item\_colon'=>'Parent Item'

);

$args=array(

'labels'=>$labels,

'public'=>true,

'has\_archive'=>true,

'publicly\_queryable'=>true,

'query\_var'=>true,

'rewrite'=>true,

'capability\_type'=>'post',

'hierarchical'=>false,

'support'=>array('title', 'editor', 'excerpt', 'thumbnail', 'revisions'),

'taxonomies'=>array('category', 'post\_tag'),

'menu\_position'=>5,

'exclude\_from\_search'=>false

);

register\_post\_type('portfolio',$args);

}

add\_action('init','custom\_postype');

## Default Custom Post Type Templates

We created our CPT with the parameter has\_archive set to true, which will display all testimonials in an archive page.

WordPress uses a template hierarchy to display different types of content. For the custom post type, it will look for:

archive-{post\_type}.php  
single-{post\_type}.php

If these templates are not defined in the theme, then it will look for the general templates:

archive.php  
single.php

If even these templates are not defined, then it will default to the index.php template.

Eg: single-product.php, single-portfolio.php, single-testimonial.php, single-news.php

/\* ==================================================================

\* 6.1. #Create Field for Custom Post type with individual meta key

====== Meta box title ==> Hello Metabox - Portfolio Post

\* ================================================================== \*/

<https://metabox.io/how-to-create-custom-meta-boxes-custom-fields-in-wordpress/>

function ax\_register\_metabox(){

add\_meta\_box('ax-metabox','Hello Metabox', 'ax\_metabox\_display','portfolio');

}

add\_action('add\_meta\_boxes','ax\_register\_metabox');

function ax\_metabox\_display(){

// echo "Custom fields created by user with individual meta key";

?>

<div>

<div>

<label>Author</label> <input id="ax\_author" name="ax\_author" type="text" value="<?php echo esc\_attr(get\_post\_meta(get\_the\_ID(),'ax\_author',true)) ?>" />

<label>Price</label> <input id="ax\_price" name="ax\_price" type="text" value="<?php echo esc\_attr(get\_post\_meta(get\_the\_ID(),'ax\_price',true)) ?>" />

</div>

</div><?php

}

// save meta

function ax\_customposttype\_save($post\_id){

if(defined('DOING\_AUTOSAVE') && DOING\_AUTOSAVE){

return;

}

if($parent\_id=wp\_is\_post\_revision($post\_id)){

$post\_id=$parent\_id;

}

$field\_list=['ax\_author', 'ax\_price' ];

foreach($field\_list as $fieldname){

if (array\_key\_exists($fieldname,$\_POST)){

// can use either update\_post\_meta or add\_post\_meta

update\_post\_meta($post\_id, $fieldname, sanitize\_text\_field($\_POST[$fieldname]) );

//add\_post\_meta( $post\_id, $fieldname, sanitize\_text\_field($\_POST[$fieldname]));

}

}

}

add\_action('save\_post','ax\_customposttype\_save');

/\* ============================================================

\* @Create Field for Custom Post type with individual meta key

\* ============================================================ \*/

**/\* Example code for tutorial post on custom meta boxes:**

\* https://wptheming.com/2010/08/custom-metabox-for-post-type/ \*/

**/\* Registers the event post type. \*/**

function wpt\_event\_post\_type() {

$labels = array('name' => \_\_( 'Events' ),

'singular\_name' => \_\_( 'Event' ),

'add\_new' => \_\_( 'Add New Event' ),

'add\_new\_item' => \_\_( 'Add New Event' ),

'edit\_item' => \_\_( 'Edit Event' ),

'new\_item' => \_\_( 'Add New Event' ),

'view\_item' => \_\_( 'View Event' ),

'search\_items' => \_\_( 'Search Event' ),

'not\_found' => \_\_( 'No events found' ),

'not\_found\_in\_trash' => \_\_( 'No events found in trash' )

);

$supports = array(

'title',

'editor',

'thumbnail',

'comments',

'revisions',

);

$args = array(

'labels' => $labels,

'supports' => $supports,

'public' => true,

'capability\_type' => 'post',

'rewrite' => array( 'slug' => 'events' ),

'has\_archive' => true,

'menu\_position' => 30,

'menu\_icon' => 'dashicons-calendar-alt',

'register\_meta\_box\_cb' => 'wpt\_add\_event\_metaboxes',

);

register\_post\_type( 'events', $args );

}

add\_action( 'init', 'wpt\_event\_post\_type' );

/\* Adds a metabox to the right side of the screen under the “Publish” box \*/

function wpt\_add\_event\_metaboxes() {

add\_meta\_box(

'wpt\_events\_location',

'Event Location',

'wpt\_events\_location',

'events',

'side',

'default'

);

}

/\* If you wanted to have two sets of metaboxes. \*/

function add\_events\_metaboxes\_v2() {

add\_meta\_box(

'wpt\_events\_date',

'Event Date',

'wpt\_events\_date',

'events',

'side',

'default'

);

add\_meta\_box(

'wpt\_events\_location',

'Event Location',

'wpt\_events\_location',

'events',

'normal',

'high'

);

}

**/\* Output the HTML for the metabox. \*/**

function wpt\_events\_location() {

global $post;

// Nonce field to validate form request came from current site

wp\_nonce\_field( basename( \_\_FILE\_\_ ), 'event\_fields' );

// Get the location data if it's already been entered

$location = get\_post\_meta( $post->ID, 'evlocation', true );

$test = get\_post\_meta( $post->ID, 'evtest', true );

// Output the field

echo '<input type="text" name="evlocation" value="' . esc\_textarea( $location ) . '" class="widefat">

<input type="text" name="evtest" value="' . esc\_textarea( $test ) . '" class="widefat1"> ';

}

/\*\*

\* Save the metabox data

\*/

function wpt\_save\_events\_meta( $post\_id, $post ) {

// Return if the user doesn't have edit permissions.

if ( ! current\_user\_can( 'edit\_post', $post\_id ) ) {

return $post\_id;

}

// Verify this came from the our screen and with proper authorization,

// because save\_post can be triggered at other times.

if ( ! isset( $\_POST['evlocation'] ) || ! wp\_verify\_nonce( $\_POST['event\_fields'], basename(\_\_FILE\_\_) ) ) {

return $post\_id;

}

if ( ! isset( $\_POST['evtest'] ) || ! wp\_verify\_nonce( $\_POST['event\_fields'], basename(\_\_FILE\_\_) ) ) {

return $post\_id;

}

// Now that we're authenticated, time to save the data.

// This sanitizes the data from the field and saves it into an array $events\_meta.

**$events\_meta['evlocation'] = esc\_textarea( $\_POST['evlocation'] ); or**

**$events\_meta=array('evlocation' => $\_POST['evlocation'],**

**'evtest' => $\_POST['evtest'], );**

// Cycle through the $events\_meta array.

// Note, in this example we just have one item, but this is helpful if you have multiple.

foreach ( $events\_meta as $key => $value ) :

// Don't store custom data twice

if ( 'revision' === $post->post\_type ) {

return;

}

if ( get\_post\_meta( $post\_id, $key, false ) ) {

// If the custom field already has a value, update it.

update\_post\_meta( $post\_id, $key, $value );

} else {

// If the custom field doesn't have a value, add it.

add\_post\_meta( $post\_id, $key, $value);

}

if ( ! $value ) {

// Delete the meta key if there's no value

delete\_post\_meta( $post\_id, $key );

}

endforeach;

}

add\_action( 'save\_post', 'wpt\_save\_events\_meta', 1, 2 );

/\*=================================================

/\*# Register Custom Post Type\*/

/\*=================================================\*/

function custom\_post\_type\_testimonials() {

$labels = array(

'name' => \_x('Testimonials', 'Post Type General Name', 'text\_domain'),

'singular\_name' => \_x('Testimonial', 'Post Type Singular Name', 'text\_domain'),

'menu\_name' => \_\_( 'Testimonials', 'text\_domain' ),

'name\_admin\_bar' => \_\_( 'Testimonials', 'text\_domain' ),

'archives' => \_\_( 'Item Archives', 'text\_domain' ),

'parent\_item\_colon' => \_\_( 'Parent Item:', 'text\_domain' ),

'all\_items' => \_\_( 'All Items', 'text\_domain' ),

'add\_new\_item' => \_\_( 'Add New Item', 'text\_domain' ),

'add\_new' => \_\_( 'Add New', 'text\_domain' ),

'new\_item' => \_\_( 'New Item', 'text\_domain' ),

'edit\_item' => \_\_( 'Edit Item', 'text\_domain' ),

'update\_item' => \_\_( 'Update Item', 'text\_domain' ),

'view\_item' => \_\_( 'View Item', 'text\_domain' ),

'search\_items' => \_\_( 'Search Item', 'text\_domain' ),

'not\_found' => \_\_( 'Not found', 'text\_domain' ),

'not\_found\_in\_trash' => \_\_( 'Not found in Trash', 'text\_domain' ),

'featured\_image' => \_\_( 'Featured Image', 'text\_domain' ),

'set\_featured\_image' => \_\_( 'Set featured image', 'text\_domain' ),

'remove\_featured\_image' => \_\_( 'Remove featured image', 'text\_domain' ),

'use\_featured\_image' => \_\_( 'Use as featured image', 'text\_domain' ),

'insert\_into\_item' => \_\_( 'Insert into item', 'text\_domain' ),

'uploaded\_to\_this\_item' => \_\_( 'Uploaded to this item', 'text\_domain' ),

'items\_list' => \_\_( 'Items list', 'text\_domain' ),

'items\_list\_navigation' => \_\_( 'Items list navigation', 'text\_domain' ),

'filter\_items\_list' => \_\_( 'Filter items list', 'text\_domain' ),

);

$args = array(

'label' => \_\_( 'Testimonials', 'text\_domain' ),

'labels' => $labels,

'supports' => array( 'title', 'editor', 'excerpt', 'thumbnail', 'custom-fields', 'revisions'),

'hierarchical' => false,

'public' => true,

'show\_ui' => true,

'show\_in\_menu' => true,

'menu\_position' => 5,

'show\_in\_admin\_bar' => true,

'show\_in\_nav\_menus' => true,

'can\_export' => true,

'has\_archive' => true,

'exclude\_from\_search' => false,

'publicly\_queryable' => true,

'capability\_type' => 'post',

'query\_var' => true,

'rewrite' => true,

'taxonomies' => array('category','post\_tag'),

);

register\_post\_type( 'testimonials', $args );

}

add\_action( 'init', 'custom\_post\_type\_testimonials' );

/\*=================================================

/\*@Register Custom Post Type\*/

/\*=================================================\*/

/\*=================================================

// #Register Custom Taxonomy

/\*=================================================\*/

function custom\_taxonomy\_course\_service() {

$labels = array(

'name' => \_x('Courses','TaxonomyGeneralName','text\_domain'),

'singular\_name' => \_x('Course','TaxonomySingularName','text\_domain'),

'menu\_name' => \_\_( 'Courses', 'text\_domain' ),

'all\_items' => \_\_( 'All Items', 'text\_domain' ),

'parent\_item' => \_\_( 'Parent Item', 'text\_domain' ),

'parent\_item\_colon' => \_\_( 'Parent Item:', 'text\_domain' ),

'new\_item\_name' => \_\_( 'New Item Name', 'text\_domain' ),

'add\_new\_item' => \_\_( 'Add New Item', 'text\_domain' ),

'edit\_item' => \_\_( 'Edit Item', 'text\_domain' ),

'update\_item' => \_\_( 'Update Item', 'text\_domain' ),

'view\_item' => \_\_( 'View Item', 'text\_domain' ),

'separate\_items\_with\_commas' => \_\_( 'Separate items with commas', 'text\_domain' ),

'add\_or\_remove\_items' => \_\_( 'Add or remove items', 'text\_domain' ),

'choose\_from\_most\_used' => \_\_( 'Choose from the most used', 'text\_domain' ),

'popular\_items' => \_\_( 'Popular Items', 'text\_domain' ),

'search\_items' => \_\_( 'Search Items', 'text\_domain' ),

'not\_found' => \_\_( 'Not Found', 'text\_domain' ),

'no\_terms' => \_\_( 'No items', 'text\_domain' ),

'items\_list' => \_\_( 'Items list', 'text\_domain' ),

'items\_list\_navigation' => \_\_( 'Items list navigation', 'text\_domain' ),

);

$args = array(

'labels' => $labels,

'hierarchical' => false,

'public' => true,

'show\_ui' => true,

'show\_admin\_column' => true,

'show\_in\_nav\_menus' => true,

'show\_tagcloud' => true,

);

register\_taxonomy( 'course\_service', array( 'courses' ), $args );

}

add\_action( 'init', 'custom\_taxonomy\_course\_service', 0 );

/\*=================================================

// #Register Custom Taxonomy

/\*=================================================\*/

/\*=================================================

/\*#Register Custom Sidebars\*/

/\*=================================================\*/

function sidebar\_register() {

$args = array(

'name' => \_\_( 'Footer1', 'bsimple' ),

'description' => \_\_( 'Footer1', 'bsimple' ),

'id' => 'b\_c\_s',

'class' => 'bottom\_center\_sidebar',

'before\_widget' => '<div id="bottom-center-sidebar">',

'after\_widget' => '</div>',

'before\_title' => '<h2 class="widgettitle">',

'after\_title' => '</h2>',

);

register\_sidebar($args);

$args = array(

'name' => \_\_( 'Footer2', 'bsimple' ),

'description' => \_\_( 'Footer2', 'bsimple' ),

'id' => 'b\_c\_s2',

'class' => 'bottom\_center\_sidebar',

'before\_widget' => '<div id="bottom-center-sidebar">',

'after\_widget' => '</div>',

'before\_title' => '<h2 class="widgettitle">',

'after\_title' => '</h2>',

);

register\_sidebar($args);

}

add\_action( 'widgets\_init', 'sidebar\_register' );

/\*=================================================

/\*@Register Custom Sidebars\*/

/\*=================================================\*/

|  |  |
| --- | --- |
| [registered\_post\_type](https://adambrown.info/p/wp_hooks/hook/registered_post_type) | action |
| [registered\_taxonomy](https://adambrown.info/p/wp_hooks/hook/registered_taxonomy) | action |
| [registered\_taxonomy\_for\_object\_type](https://adambrown.info/p/wp_hooks/hook/registered_taxonomy_for_object_type) | action |
| [register\_form](https://adambrown.info/p/wp_hooks/hook/register_form) | action |
| [register\_meta\_args](https://adambrown.info/p/wp_hooks/hook/register_meta_args) | filter |
| [register\_new\_user](https://adambrown.info/p/wp_hooks/hook/register_new_user) | action |
| [register\_post](https://adambrown.info/p/wp_hooks/hook/register_post) | action |
| [register\_post\_type\_args](https://adambrown.info/p/wp_hooks/hook/register_post_type_args) | filter |
| [register\_setting\_args](https://adambrown.info/p/wp_hooks/hook/register_setting_args) | filter |
| [register\_sidebar](https://adambrown.info/p/wp_hooks/hook/register_sidebar) | action |
| [register\_taxonomy\_args](https://adambrown.info/p/wp_hooks/hook/register_taxonomy_args) | filter |
| [register\_url](https://adambrown.info/p/wp_hooks/hook/register_url) | filter |
| [registration\_errors](https://adambrown.info/p/wp_hooks/hook/registration_errors) | filter |
| [registration\_redirect](https://adambrown.info/p/wp_hooks/hook/registration_redirect) | filter |

<https://gist.github.com/mahisalman/111eb23fef027b1b048b>

**<!-- Wordpress Theme Development CheatSheet...-->**

**<!--1. ---- For Making Language Dynamic.-->**

<?php language\_attributes(); ?>

**<!--2. ---- For Making Character Ser Dynamic.-->**

<?php bloginfo( 'charset' ); ?>

**<!--3. ----- For Dynamic Title, set by user.-->**

<title><?php wp\_title( '|', true, 'right' ); ?></title>

**<!--4. ---- For Adding Functions.php file adn add css and js files.-->**

<?php wp\_head(); ?>

**<!-- 5. ---- For Adding Functions.php file adn add css and js files.-->**

<?php wp\_Footer(); ?>

**<!--6. ---- Function Structure.-->**

<?php function MY\_FUNCTION\_NAME (){}

add\_ACTION('','MY\_FUNCTION\_NAME'); ?>

**<!--7. ---- for making Style Sheet Dynamic.-->**

[<?php

wp\_register\_style('bootstrap\_min',get\_template\_directory\_uri().'/css/bootstrap.min.css',array(),'3.1.1','all');

wp\_enqueue\_style('bootstrap\_min');

add\_action('wp\_enqueue\_scripts','shuffle\_theme\_files'); ?>]

**<!-- 8. HAve to call for get the default jquery file.-->**

[<?php wp\_enqueue\_script('jquery'); ?>]

**<!--9. ---- for making JS file Dynamic.-->**

[<?php

wp\_enqueue\_script('jquery.min',get\_template\_directory\_uri().'/js/jquery.min.js',array('jquery'),'1.10.2',true);

add\_action('wp\_enqueue\_scripts','shuffle\_theme\_files'); ?>]

**<!--10. ---- For Making Conditional Linking of CSS or JS file.-->**

[<?php

global $wp\_styles;

wp\_enqueue\_style('font\_awesome');

$wp\_styles->add\_data( 'font\_awesome', 'conditional', 'IE' ); ?>]

<!--http://css-tricks.com/how-to-create-an-ie-only-stylesheet/ ---- for know more about conditional css.-->

**<!--11. ---- Seperate the header part and call from header.php file.-->**

<?php get\_header(); ?>

**<!--12. ---- Seperate the Footer part and call from Footer.php file.-->**

<?php get\_footer(); ?>

**<!--13. ---- For adding php files in another php file.-->**

<?php include\_once('include/theme-files.php'); ?>

**<!--19. ---- For Changing Template & Creating Default Template-->**

<?php /\* Template Name: Welcome Template \*/ get\_header(); ?>

**<!--20. ------ PHP if else condition..-->**

<?php

if (have\_posts()) : while (have\_posts()) : the\_post(); ?>

<h2><?php the\_title(); ?></h2>

<?php the\_content('<p class="serif">Read the rest of this page »</p>'); ?>

<?php endwhile; ?><?php else : ?>

<h2>404 Not Found!</h2><?php endif; ?>

**<!--21. ---- For Adding Slide in Dashboard menu.--> <?php**

function corlate\_theme\_custom\_posts(){

register\_post\_type('slide',array( 'label' => 'slides',

'labels' => array( 'name' => 'Slides',

'singular\_name' => 'Slide',

'add\_new' => 'add new',

'edit\_item' => 'edit', ),

'public' => true,

'menu\_icon' => 'dashicons-images-alt',

'supports' => array('title','editor','thumbnail', 'custom-fields','excerpt'),

)); }

add\_action('init','corlate\_theme\_custom\_posts'); ?>

**<!--22. ---- Adding the title and description part.-->**

<?php

global $post;

$args = array( 'posts\_per\_page' => 5, 'post\_type'=> 'slide' );

$myposts = get\_posts( $args );

foreach( $myposts as $post ) : setup\_postdata($post); ?> <!---- Calling Data From Database -->

<h1 class="animation animated-item-1"><?php the\_title(); ?></h1>

<h2 class="animation animated-item-2"><?php the\_excerpt(); ?></h2>

**<!--23. ---- For crop and adding slider featured image.-->**

<?php add\_theme\_support('post-thumbnails'); ?> <!---- For adding feature image. -->

<?php add\_image\_size( 'slider-bg', 1400, 730, true ); ?> <!---- For crop image in the given size. -->

<?php $slide\_bg = wp\_get\_attachment\_image\_src( get\_post\_thumbnail\_id( $post->ID ), 'slider-bg' ); ?>

<div class="item active" style="background-image: url(<?php echo $slide\_bg[0]; ?>)">

**<!-- 24. ----- Adding custom field as read more link, additional images etc.-->**

<?php $theme\_thumb = get\_post\_meta($post->ID, 'theme\_thumb', true); ?>

<img src="<?php echo $theme\_thumb;?>" class="img-responsive">

<?php $theme\_link = get\_post\_meta($post->ID, 'theme\_link', true); ?>

<a class="btn-slide animation animated-item-3" href="<?php echo $theme\_link; ?>">Read More</a>

**<!-- 25.-->**

<?php global $post;

$args = array( 'posts\_per\_page' => 6, 'post\_type'=> 'promotions', 'orderby' => 'menu\_order', 'order' => 'ASC' );

$myposts = get\_posts( $args );

foreach( $myposts as $post ) : setup\_postdata($post); ?>

<?php $promotion\_image= get\_post\_meta($post->ID, 'promotion\_image', true); ?>

<div class="col-md-4 col-sm-6 wow fadeInDown" data-wow-duration="1000ms" data-wow-delay="600ms">

<div class="feature-wrap">

<i class="fa fa-<?php echo $promotion\_image; ?>"></i> <!---- is for adding icon. -->

<h2><?php the\_title(); ?></h2> <!---- for adding title of promotion -->

<h3><?php the\_content(); ?></h3> <!---- the excerpt or the description to add.-->

</div>

</div><!--/.col-md-4--> <?php endforeach; ?>

**<!--26. ---- For Promotion-->**

<?php

register\_post\_type( 'promotions', array('label' => 'promotions',

'labels' => array( 'name' => 'promotions', 'singular\_name' => 'promotion', ),

'public' => true,

'menu\_icon' => 'dashicons-images-alt',

'supports' => array('title','editor','thumbnail','custom-fields','page-attributes') )); ?>

**<!-- 27.-->**

<?php global $post;

$args = array( 'posts\_per\_page' => 8, 'post\_type'=> 'portfolios', 'orderby' => 'menu\_order', 'order' => 'ASC' );

$myposts = get\_posts( $args );

foreach( $myposts as $post ) : setup\_postdata($post); ?>

<?php $Portfolio\_image = wp\_get\_attachment\_image\_src( get\_post\_thumbnail\_id( $post->ID ), 'Portfolio\_image'); ?>

<?php $Portfolio\_image\_full = wp\_get\_attachment\_image\_src( get\_post\_thumbnail\_id( $post->ID ), 'large'); ?>

<div class="col-xs-12 col-sm-4 col-md-3">

<div class="recent-work-wrap">

<img class="img-responsive" src="<?php echo $Portfolio\_image[0]; ?>" alt="">

<div class="overlay">

<div class="recent-work-inner">

<h3><a href="<?php echo the\_permalink(); ?>"><?php the\_title(); ?></a> </h3>

<p><?php the\_content(); ?></p>

<a class="preview" href="<?php echo $Portfolio\_image\_full[0]; ?>" rel="prettyPhoto"><i class="fa fa-eye"></i> View</a>

</div>

</div>

</div>

</div> <?php endforeach; ?>

**<!-- 28.-->**

<?php

register\_post\_type( 'portfolios', array('label' => 'portfolios',

'labels' => array( 'name' => 'portfolios', 'singular\_name' => 'portfolio', ),

'public' => true,

'menu\_icon' => 'dashicons-images-alt',

'supports' => array('title','editor','thumbnail','custom-fields','excerpt','page-attributes') ));?>

**<!--29. ----- For adding image size.-->**

<?php add\_image\_size( 'Portfolio\_image', 293 , 222, true ); ?>

**<!--30.-->**

<?php

function corlate\_theme\_widgets(){

register\_sidebar(array( 'name' => 'footer Widgets',

'description' => 'Use this widgets for your site footer conten.',

'id' => 'footer wid',

'before\_widget' => '<div class="col-md-3 col-sm-6"><div class="widget">',

'after\_widget' => '</div></div>',

'before\_title' => '<h3>',

'after\_title' => '</h3>',));}

Add\_action('widgets\_init','corlate\_theme\_widgets');?>

**<!--31.-->**

<?php if ( ! dynamic\_sidebar( 'footer\_wid' ) ): ?>

<?php endif; ?>

## Conditional Tags

<https://codex.wordpress.org/Conditional_Tags>

1. [**The Conditions For ...**](https://codex.wordpress.org/Conditional_Tags#The_Conditions_For_...)
   1. [The Main Page](https://codex.wordpress.org/Conditional_Tags#The_Main_Page) and [The Front Page](https://codex.wordpress.org/Conditional_Tags#The_Front_Page) and [The Blog Page](https://codex.wordpress.org/Conditional_Tags#The_Blog_Page)

if ( is\_front\_page() && is\_home() ) {

// Default homepage

} elseif ( is\_front\_page() ) {

// static homepage

} elseif ( is\_home() ) {

// blog page

} else {

//everything else

}

* 1. [**The Administration Panels**](https://codex.wordpress.org/Conditional_Tags#The_Administration_Panels)

[is\_admin()](https://codex.wordpress.org/Function_Reference/is_admin)**-**  Dashboard or the administration panels are being displayed.

[is\_network\_admin()](https://codex.wordpress.org/Function_Reference/is_network_admin) **-**  Network Dashboard for multisite are being displayed.

* 1. [**The Admin Bar**](https://codex.wordpress.org/Conditional_Tags#The_Admin_Bar)[is\_admin\_bar\_showing()](https://codex.wordpress.org/Function_Reference/is_admin_bar_showing)
  2. [**A Single Post Page**](https://codex.wordpress.org/Conditional_Tags#A_Single_Post_Page)[is\_single()](https://codex.wordpress.org/Function_Reference/is_single)

When a single post of any post type (except attachment and page post types) is being displayed.

**is\_single( '17' )**

When Post 17 is being displayed as a single Post.

**is\_single( 'Irish Stew' )**

When the Post with Title "Irish Stew" is being displayed as a single Post.

**is\_single( 'beef-stew' )**

When the Post with Post Slug "beef-stew" is being displayed as a single Post.

**is\_single( array( 17, 'beef-stew', 'Irish Stew' ) )**

Returns true when the single post being displayed is either post ID 17, or the *post\_name* is "beef-stew", or the *post\_title* is "Irish Stew".

**is\_single( array( 17, 1, 11 ) )**

**is\_single( array( 'beef-stew', 'pea-soup') )**

**is\_single( array( 'Beef Stew', 'Pea Soup') )**

Note: This function does not distinguish between the post ID, post title, or post name. A post named "17" would be displayed if a post ID of 17 was requested. Presumably the same holds for a post with the slug "17".

<https://codex.wordpress.org/Conditional_Tags>

~~AND so on click link …~~

<https://pippinsplugins.com/a-quick-introduction-to-using-filters/>

## Hooks: Actions and Filters

Hooks are provided by WordPress to allow your plugin to 'hook into' the rest of WordPress; that is, to call functions in your plugin at specific times, and thereby set your plugin in motion. There are two kinds of hooks:

1. [Actions](http://codex.wordpress.org/Glossary#Action) (Codex [Action Reference](https://codex.wordpress.org/Plugin_API/Action_Reference))
2. [Filters](http://codex.wordpress.org/Glossary#Filter) (Codex [Filter Reference](https://codex.wordpress.org/Plugin_API/Filter_Reference))

You can sometimes accomplish the same goal with either an action or a filter. For example, if you want your plugin to change the text of a post, you might add an action function to publish\_post (so the post is modified as it is saved to the database), or a filter function to the\_content (so the post is modified as it is displayed in the browser screen).

For a thorough listing of all action and filter hooks in WP see **Adam Brown's** [WordPress Hooks Database](http://adambrown.info/p/wp_hooks).

## Function Reference

|  |  |  |
| --- | --- | --- |
| |  | | --- | | **Filter Functions** | | * [has\_filter()](https://codex.wordpress.org/Function_Reference/has_filter) * [add\_filter()](https://codex.wordpress.org/Function_Reference/add_filter) * [apply\_filters()](https://codex.wordpress.org/Function_Reference/apply_filters) * [apply\_filters\_ref\_array()](https://codex.wordpress.org/Function_Reference/apply_filters_ref_array) * [current\_filter()](https://codex.wordpress.org/Function_Reference/current_filter) * [remove\_filter()](https://codex.wordpress.org/Function_Reference/remove_filter) * [remove\_all\_filters()](https://codex.wordpress.org/Function_Reference/remove_all_filters) * [doing\_filter()](https://codex.wordpress.org/Function_Reference/doing_filter) | |

|  |
| --- |
| **Actions Functions** |
| * [has\_action()](https://codex.wordpress.org/Function_Reference/has_action) * [add\_action()](https://codex.wordpress.org/Function_Reference/add_action) * [do\_action()](https://codex.wordpress.org/Function_Reference/do_action) * [do\_action\_ref\_array()](https://codex.wordpress.org/Function_Reference/do_action_ref_array) * [did\_action()](https://codex.wordpress.org/Function_Reference/did_action) * [remove\_action()](https://codex.wordpress.org/Function_Reference/remove_action) * [remove\_all\_actions()](https://codex.wordpress.org/Function_Reference/remove_all_actions) * [doing\_action()](https://codex.wordpress.org/Function_Reference/doing_action) |
|  |

|  |  |
| --- | --- |
| **Activation/Deactivation/Uninstall Functions** | |
| * [register\_activation\_hook()](https://codex.wordpress.org/Function_Reference/register_activation_hook) * [register\_uninstall\_hook()](https://codex.wordpress.org/Function_Reference/register_uninstall_hook) | * [register\_deactivation\_hook()](https://codex.wordpress.org/Function_Reference/register_deactivation_hook) |

## Types of Action and Filter Hooks

<https://blog.teamtreehouse.com/hooks-wordpress-actions-filters-examples>

The WordPress Codex has two important pages that can help you orient yourself with what hooks are available in WordPress.

NOTE: At the time of writing there is a bit of a transition of documentation on hooks from the [*Codex*](http://codex.wordpress.org/) to the [*Code Reference*](https://developer.wordpress.org/reference/) site. Because of this you may find hooks that are listed on the Codex, but not documented. You should be able to find some documentation for all of the hooks if you search for them in the Code Reference, however, it does not have any current pages like the Codex where all hooks are grouped in one place.

The [WordPress Action Reference page](http://codex.wordpress.org/Plugin_API/Action_Reference) has available actions listed by the following categories:

* [Actions Run During a Typical Request](http://codex.wordpress.org/Plugin_API/Action_Reference#Actions_Run_During_a_Typical_Request)
* [Actions Run During an Admin Page Request](http://codex.wordpress.org/Plugin_API/Action_Reference#Actions_Run_During_an_Admin_Page_Request)
* [Post, Page, Attachment, and Category Actions (Admin)](http://codex.wordpress.org/Plugin_API/Action_Reference#Post.2C_Page.2C_Attachment.2C_and_Category_Actions_.28Admin.29)
* [Comment, Ping, and Trackback Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Comment.2C_Ping.2C_and_Trackback_Actions)
* [Blogroll Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Blogroll_Actions)
* [Feed Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Feed_Actions)
* [Template Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Template_Actions)
* [Administrative Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Administrative_Actions)
* [Dashboard “Right Now” Widget Actions](http://codex.wordpress.org/Plugin_API/Action_Reference#Dashboard_.22Right_Now.22_Widget_Actions)

The Codex also has a similar [**Filter Reference page**](http://codex.wordpress.org/Plugin_API/Filter_Reference), which it lists by the following categories:

* [Post, Page, and Attachment (Upload) Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Post.2C_Page.2C_and_Attachment_.28Upload.29_Filters)
* [Comment, Trackback, and Ping Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Comment.2C_Trackback.2C_and_Ping_Filters)
* [Category and Term Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Category_and_Term_Filters)
* [Link Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Link_Filters)
* [Date and Time Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Date_and_Time_Filters)
* [Author and User Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Author_and_User_Filters)
* [Blogroll Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Blogroll_Filters)
* [Blog Information and Option Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Blog_Information_and_Option_Filters)
* [General Text Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#General_Text_Filters)
* [Administrative Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Administrative_Filters)
* [Rich Text Editor Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Rich_Text_Editor_Filters)
* [Template Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Template_Filters)
* [Registration & Login Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Registration_.26_Login_Filters)
* [Redirect/Rewrite Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Redirect.2FRewrite_Filters)
* [WP\_Query Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#WP_Query_Filters)
* [Media Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Media_Filters)
* [Advanced WordPress Filters](http://codex.wordpress.org/Plugin_API/Filter_Reference#Advanced_WordPress_Filters)
* [Widgets](http://codex.wordpress.org/Plugin_API/Filter_Reference#Widgets)
* [Admin Bar](http://codex.wordpress.org/Plugin_API/Filter_Reference#Admin_Bar)

Many of these filter hooks are separated into subcategories: Database Reads and Database Writes. This depends on whether you are reading from the database prior to displaying on a page or editing a screen, or you are writing code prior to saving data to the database.

Implementation in Your Own Plugins

In order to make your plugin truly extensible, you need to use the key functions mentioned above throughout the entire plugin. At first you may find it difficult, cumbersome, annoying, or many other applicable adjectives, to constantly place these additional functions throughout your code, especially when you do not see an immediate benefit or use for them.

What you will find, however, is that once you are in the habit of writing your plugins with all of these functions in mind, it will become second nature to include them.

There are a few main scenarios where you will use filters in your plugins:

* When arrays are setup. Filters will be added here so that other plugins can modify the data before it is used.
* When data objects are setup. Just like with arrays, you will use a filter on objects so that other developers can alter the object before it is used.
* When data strings are setup. With a filter available on a string, other developers can change the entire string, alter parts of it, or add onto it.

Of the scenarios above, it is most common to use filters when data is returned or just before it is used. For example, if you have a plugin that performs a posts query, it's best to pass the array of query arguments through a filter before they are passed to get\_posts() or WP\_Query so that others can manipulate the query before it is made.

When it comes to actions, there are also several main instances where you will place them:

* Before a task is executed.
* After a task is executed.
* Within your markup to allow additional markup to be inserted.

Let's consider a few examples now.

### 1. Displaying HTML With a Shortcode

Shortcodes that output HTML are extremely common (actually they are probably the most common of all shortcodes), and one of the ways that we can make our plugin's shortcodes more friendly to other developers is by providing a way for them to alter the contents of the shortcode, but without requiring that it be deregistered and registered again.

All shortcodes return rather than echoing their contents, which means that the data outputted to the screen is going to be in the form of a string before it is returned. Because the entire HTML output is in the form of a string, you can pass the string through a filter before returning it. Doing so will make it possible for other developers to alter the HTML of your shortcode.

A developer may want to add additional markup before and after the default HTML: with the filter in place, they can do that.

Your shortcode may look something like this:

function wptp\_sample\_shortcode( atts, $content = null ) {

$html = '<div class="wptp\_shortcode">';

$html .= '<p>Contents of the sample shortcode</p>';

$html .= '</div>';

return $html;

}

We can improve this by adding a filter to the return, like so:

function wptp\_sample\_shortcode( atts, $content = null ) {

$html = '<div class="wptp\_shortcode">';

$html .= '<p>Contents of the sample shortcode</p>';

$html .= '</div>';

return apply\_filters( 'wptp\_shortcode\_html', $html );

}

The HTML in our shortcode can now be modified like this:

function wptp\_modify\_html( $html ) {

return '<div class="extra\_div">' . $html . '</div>';

}

add\_filter( 'wptp\_shortcode\_html', 'wptp\_modify\_html' );

This will result in the original HTML created in the shortcode being wrapped with another div tag.

2. Querying Posts

Performing custom queries in plugins is a common practice. Let''s assume for a moment that you have written a plugin that registers a custom post type called "books", and in your plugin is a function to show created books. Your function to query the books might look something like this:

function wptp\_show\_books() {

$query\_args = array(

'post\_type' => 'books',

'posts\_per\_page' => 5

);

$books = new WP\_Query( $query\_args );

if( $books->have\_posts() ) :

while( $books->have\_posts() ) : $books->the\_post()

// show info about each book here

endwhile;

endif;

wp\_reset\_postdata();

}

But what if a user wanted to modify the kinds of books that were returned, perhaps selecting only books from a specific category? You can make this much easier for them by doing this:

function wptp\_show\_books() {

$query\_args = array(

'post\_type' => 'books',

'posts\_per\_page' => 5,

'author' => 3

);

$books = new WP\_Query( apply\_filters( 'wptp\_books\_query', $query\_args ) ) ;

if( $books->have\_posts() ) :

while( $books->have\_posts() ) : $books->the\_post()

// show info about each book here

endwhile;

endif;

wp\_reset\_postdata();

}

The only change I made was to add a filter around the $query\_args, which means other developers (or users) can modify the query arguments before they are actually passed to WP\_Query. For example, we could set the query to only show books from author 3 like this:

function wptp\_alter\_books\_query( $args ) {

$args['author'] = 3;

return $args;

}

add\_filter( 'wptp\_books\_query', 'wptp\_alter\_books\_query' );

3. Extending Markup

Let''s extend on example number 2 now and make it even better. We already added a filter that allows users to modify the query, now let''s add a few hooks to let us alter the HTML that is created.

First we''ll modify our original HTML a bit:

function wptp\_show\_books() {

$query\_args = array(

'post\_type' => 'books',

'posts\_per\_page' => 5,

'author' => 3

);

$books = new WP\_Query( apply\_filters( 'wptp\_books\_query', $query\_args ) ;

if( $books->have\_posts() ) :

echo '<div class="wptp\_books">';

while( $books->have\_posts() ) : $books->the\_post()

echo '<div class="wptp\_book">';

echo '<h3 class="wptp\_book\_title">' . get\_the\_title() . '</h3>';

echo '</div>';

endwhile;

echo '</div>';

endif;

wp\_reset\_postdata();

}

We would now like to make it possible for developers to add in extra markup at various points, such as the following:

* Before any HTML is outputted
* After the ending HTML
* Before the title of each book
* After the title of each book

You can imagine a scenario where a user would want to add a thumbnail before or after the book title. To make this possible, we use do\_action() to create hookable locations, like this:

function wptp\_show\_books() {

$query\_args = array(

'post\_type' => 'books',

'posts\_per\_page' => 5,

'author' => 3

);

$books = new WP\_Query( apply\_filters( 'wptp\_books\_query', $query\_args ) );

if( $books->have\_posts() ) :

do\_action( 'wptp\_books\_before' );

echo '<div class="wptp\_books">';

while( $books->have\_posts() ) : $books->the\_post()

echo '<div class="wptp\_book">';

do\_action( 'wptp\_before\_book\_title', get\_the\_ID() );

echo '<h3 class="wptp\_book\_title">' . get\_the\_title() . '</h3>';

do\_action( 'wptp\_after\_book\_title', get\_the\_ID() );

echo '</div>';

endwhile;

echo '</div>';

do\_action( 'wptp\_books\_after' );

endif;

wp\_reset\_postdata();

}

Note that the two inner hooks (surrounding the title) have a second parameter of get\_the\_ID(). This variable, which will be the ID of the book, will be available as a parameter to any hooked function. To add in a book thumbnail, for example, we can do this:

function wptp\_show\_book\_image( $book\_id ) {

echo get\_the\_post\_thumbnail( $book\_id, 'thumbnail' );

}

add\_action( 'wptp\_before\_book\_title', 'wptp\_show\_book\_image' );

The key difference between actions and filters is that actions are usually used for executing functions and filters are usually used for manipulating data.  
For actions there are four main functions:

* do\_action() - this defines a hookable location for actions
* add\_action() - this attaches a function to a hook created with do\_action()
* has\_action() - this checks to see if an action has been registered with do\_action()
* remove\_action() - this removes an action that was set with add\_action()

Of these four, you will use do\_action() and add\_action() the most.

For filters there are also four main functions:

* apply\_filters() - this creates a hookable location for custom filters to tie into
* add\_filter() - this attaches a custom filter to a hook created with apply\_filters()
* has\_filter() - this checks to see if a filter has been registered with apply\_filters()
* remove\_filter() - this removes a filter previously connected to apply\_filters()

As with actions, apply\_filters() and add\_filter() are the two you will use the most.

If you're confused at this point, do not worry, we are going to look at how to actually use these in the next section.

***When do I use which hook?***

And here's the advice I normally give:

* Use actions when you want to add something to the existing page such as stylesheets, JavaScript dependencies, or send an email when an event has happened.
* Use filters when you want to manipulate data coming out of the database prior to going to the browser, or coming from the browser prior to going into the database.
* [add\_filter()](http://codex.wordpress.org/Function_Reference/add_filter) – used for adding a new custom filter
* [remove\_filter()](http://codex.wordpress.org/Function_Reference/remove_filter) – used for removing an already registered filter
* [apply\_filters()](http://codex.wordpress.org/Function_Reference/apply_filters) – runs the provided data through the specified filter
* [has\_filter()](http://codex.wordpress.org/Function_Reference/has_filter) – checks whether a specific filter has been registered

function pipp\_show\_fruits() {

$fruits = array('apples', 'oranges', );

$list = '<ul>';

if(has\_filter('pipp\_add\_fruits')) {

$fruits = apply\_filters('pipp\_add\_fruits', $fruits);

}

foreach($fruits as $fruit) :

$list .= '<li>' . $fruit . '</li>';

endforeach;

$list .= '</ul>';

return $list;

}

function pipp\_add\_extra\_fruits($fruits) {

// the $fruits parameter is an array of all fruits from the pipp\_show\_fruits() function

$extra\_fruits = array( 'plums', 'kiwis', );

// combine the two arrays

$fruits = array\_merge($extra\_fruits, $fruits);

return $fruits;

}

add\_filter('pipp\_add\_fruits', 'pipp\_add\_extra\_fruits');

**Hook examples**

Capitalize words in the title

Plugin Code:

add\_filter( 'the\_title', 'capitalized\_wp\_titles\_convert');

function capitalized\_wp\_titles\_convert ($title) {

$title = ucwords($title);

return $title;

}

**Why are hooks so cool?**

* extend anything without breaking it
* faster alternative to options
* open your plugin to others

**HOOKS**

actions - do something e.g. echo

filters - manipulate a value

**Elements of a hook routine**

* Hook (noun) - when
* Acton / Filter function - what
* hook (verb) - connection ( hooked )

*! Acton/filter functions can be hooked to an action/filter hook*

**Acton routine functions**

do\_action( 'genesis\_site\_title' );

function logo\_before\_title( ){}

add\_action( 'genesis\_site\_title', 'logo\_before\_title', 10, 1 );

**wp\_head action**

<html <?php language\_attributes(); ?>>

<head> … <?php wp\_head(); ?> </head>

function wp\_head() { /\* … do\_action( 'wp\_head' ); \*/ }

**wp\_head action in Core**

add\_action( 'wp\_head', '\_wp\_render\_title\_tag', 1 );

add\_action( 'wp\_head', 'wp\_enqueue\_scripts', 1 );

add\_action( 'wp\_head', 'locale\_stylesheet' );

add\_action( 'wp\_head', 'noindex', 1 );

add\_action( 'wp\_head', 'print\_emoji\_detection\_script', 7 );

add\_action( 'wp\_head', 'wp\_print\_styles', 8 );

add\_action( 'wp\_head', 'wp\_print\_head\_scripts', 9 );

add\_action( 'wp\_head', 'wp\_generator' );

add\_action( 'wp\_head', 'rel\_canonical' );

add\_action( 'wp\_head', 'wp\_site\_icon', 99 );

**Filter routine functions**

$title = apply\_filters( 'the\_title', $title );

function capitalize\_title( $title ){

return ucwords( $title );

}

add\_filter( 'the\_title', 'capitalize\_title', 10, 1 );

**the\_content filter**

the\_content();

function the\_content( $more\_link\_text = null, $strip\_teaser = false) {

$content = get\_the\_content( $more\_link\_text, $strip\_teaser );

$content = apply\_filters( 'the\_content', $content );

echo $content;

}

**the\_content filter in Core**

add\_filter( 'the\_content', 'wptexturize' );

add\_filter( 'the\_content', 'convert\_smilies' );

add\_filter( 'the\_content', 'convert\_chars' );

add\_filter( 'the\_content', 'wpautop' );

add\_filter( 'the\_content', 'shortcode\_unautop' );

add\_filter( 'the\_content', 'prepend\_attachment');

**Naming your hooks**

when

do\_action( 'before\_output' );

what + when

$content = apply\_filters( 'content\_before\_output', $content);

! never use the same name for actons and flters

**Prefix!**

do\_action( 'advanced\_ads\_before\_output' );

$content = apply\_filters( 'advanced\_ads\_content\_before\_output', $content );

**Dynamic Hooks**

when changing post status, e.g. pending\_to\_publish

do\_action( "**{**$old\_status**}**\_to\_**{**$new\_status**}**", $post );

when getting an option, e.g. opton\_default\_category

function get\_option( $option, $default = false ) {

/\* … return apply\_filters( 'option\_' . $option, maybe\_unserialize( $value ) );\*/

}

**The all hook**

"all" = will be called for all action and filter hooks

function my\_show\_filter(){ echo current\_filter(); // == current\_action(); }

add\_action( 'all', 'my\_show\_filter' );

**Functions**

function my\_capitalize\_title( $title ){

return ucwords( $title );

}

add\_filter( 'the\_title', 'my\_capitalize\_title' );

class my\_class {

public function capitalize\_title( $title ){

return ucwords( $title );

}

}

add\_filter( 'the\_title', array( 'my\_class', 'capitalize\_title' ) );

class my\_class {

public function \_\_construct(){

add\_filter( 'the\_title', array( $this, 'capitalize\_title' ) );

}

public function capitalize\_title( $title ){

return ucwords( $title );

}

}

**using PHP functions**

**instead of**

function my\_capitalize\_title( $title ){

return ucwords( $title );

}

add\_filter( 'the\_title', 'my\_capitalize\_title' );

**use this**

add\_filter( 'the\_title', 'ucwords' );

**Priority**

default: 10

lower numbers run first

add\_filter( 'the\_content', 'inject\_ad\_before\_paragraph', 5 );

add\_filter( 'the\_content', 'wpautop' );

What is going to happen?

**Arguments**

do\_action( 'delete\_term', $term, $tt\_id, $taxonomy, $deleted\_term );

up to 4 arguments can be used

add\_action( 'delete\_term', '\_wp\_delete\_tax\_menu\_item', 10, 3 );

function \_wp\_delete\_tax\_menu\_item( $object\_id = 0, $tt\_id, $taxonomy ) { /\* ..\*/ }

**Remove**

function disable\_emojis() {

remove\_action( 'wp\_head', 'print\_emoji\_detection\_script', 7 );

remove\_action( 'admin\_print\_scripts', 'print\_emoji\_detection\_script' );

remove\_action( 'wp\_print\_styles', 'print\_emoji\_styles' );

remove\_action( 'admin\_print\_styles', 'print\_emoji\_styles' );

remove\_filter( 'the\_content\_feed', 'wp\_staticize\_emoji' );

remove\_filter( 'comment\_text\_rss', 'wp\_staticize\_emoji' );

remove\_filter( 'wp\_mail', 'wp\_staticize\_emoji\_for\_email' );

}

add\_action( 'init', 'disable\_emojis' );

**Testing for hooks**

function \_wp\_render\_title\_tag() {

/\*… \*/

if ( ! did\_action( 'wp\_head' ) && ! doing\_action( 'wp\_head' ) )

{ return; }

echo …;

}

function build\_query\_string() {

/\* … \*/

if ( has\_filter( 'query\_string' ) ) { /\* … \*/ }

}

**Debugging Tips**

use DEBUG\_LOG to log errors in wp-content/debug.log fle

error\_log();

use a plugin like DEBUG OBJECTS to list flters and actons

**Want to learn more?**

* code
* htps://codex.wordpress.org/Plugin\_API
* search
* plugin or theme manuals

! there are more flters or actons than you can think of

**Learned something?**

add\_action( 'end\_of\_talk', 'answer\_questions' );

function answer\_questions( $questions ){

foreach( $questions as $\_q ){

try

{ echo find\_answer\_to( $\_q ); }

catch(Exception $e)

{ mail( 'thomas.maier@webgilde.com', 'contact', $\_q ); }

}

}

do\_action( 'end\_of\_talk' );

**SYNTAX:**

**ACTION:**

**do\_action($tag, $args);**

where:  
$tag = the name of the action hook (eg, wp\_head)

$args = optional parameter(s) passed to the actions (or functions written by you) which register with the particular action hook. Action hooks can have many or no parameters such as the wp\_head example shown above.

**do\_action\_ref\_array( $tag, $args );**

**add\_action( 'action\_hook', 'your\_function\_name', [priority], [accepted\_args] );**

**FILTER:**

* **apply\_filters( $tag, $value );**
* **apply\_filters\_ref\_array( $tag, $args );**

**add\_filter ( 'hook\_name', 'your\_filter', [priority], [accepted\_args] );**

where:

* hook\_name = name of the hook you want register to
* your\_filter = the name of your filter (or function)
* priority = (optional) integer which represents the order your filter should be applied. If no value is added, it defaults to 10.
* accepted\_args = (optional) integer argument defining how many arguments your function can accept (default 1) because your filter must accpet at least one parameter which will be returned.

**e.g:** $title = apply\_filters('wp\_title', $title, $sep, $seplocation);

add\_filter( 'wp\_title', 'mytest\_add\_sitename\_to\_title', 10, 2 );

mytest\_add\_sitename\_to\_title( $title, $sep ) {

/\* Retrieve site name. \*/

$name = get\_bloginfo( 'name' );

/\* Append site name to $title. \*/

$title .= $sep . ' ' . $name;

/\* Return the title. \*/

return $title;

}

JS

/\*\*\*\*alert message for testing\*\*\*\*\*/

jQuery(document).ready(function(){ alert('Hello!'); });

//////////////////////////Sample Script Format////////////////////

jQuery(document).ready(function () { //alert('Hello!'); });

(function ($) {

$(document).ready(function () { //alert('Hello!'); });

})(this.jQuery);

(function ($) {

//var $allVideos = $(this).find(selectors.join(','));

//$allVideos = $allVideos.not("object object"); // SwfObj conflict patch

})(window.jQuery || window.Zepto);

jQuery(function ($) {

// Quantity buttons

$('div.quantity:not(.buttons\_added), td.quantity:not(.buttons\_added)')

.addClass('buttons\_added') .append('<input type="button" value="+" class="plus" />')

.prepend('<input type="button" value="-" class="minus" />');

$('.plus').val('\uf067');

$('.minus').val('\uf068');

$(document).on('click', '.plus, .minus', function () {

// Get values

var $qty = $(this).closest('.quantity').find('.qty'),

currentVal = parseFloat($qty.val()),

max = parseFloat($qty.attr('max')),

min = parseFloat($qty.attr('min')),

step = $qty.attr('step');

// Format values

if (!currentVal || currentVal === '' || currentVal === 'NaN') currentVal = 0;

if (max === '' || max === 'NaN') max = '';

if (min === '' || min === 'NaN') min = 0;

if (step === 'any' || step === '' || step === undefined || parseFloat(step) === 'NaN') step = 1;

// Change the value

if ($(this).is('.plus')) {

if (max && (max == currentVal || currentVal > max)) {

$qty.val(max);

} else { $qty.val(currentVal + parseFloat(step)); }

} else {

if (min && (min == currentVal || currentVal < min)) {

$qty.val(min);

} else if (currentVal > 0) { $qty.val(currentVal - parseFloat(step)); }

}

// Trigger change event

$qty.trigger('change');

});

});

**Example for Sidebar Creation**

function themename\_widgets\_init() {

register\_sidebar( array(

'name' => \_\_( 'Primary Sidebar', 'theme\_name' ),

'id' => 'sidebar-1',

'before\_widget' => '<aside id="%1$s" class="widget %2$s">',

'after\_widget' => '</aside>',

'before\_title' => '<h3 class="widget-title">',

'after\_title' => '</h3>',

) );

register\_sidebar( array(

'name' => \_\_( 'Secondary Sidebar', 'theme\_name' ),

'id' => 'sidebar-2',

'before\_widget' => '<ul><li id="%1$s" class="widget %2$s">',

'after\_widget' => '</li></ul>',

'before\_title' => '<h3 class="widget-title">',

'after\_title' => '</h3>',

) );

}

add\_action( 'widgets\_init', 'my\_register\_sidebars' );

function my\_register\_sidebars() {

/\* Register the 'primary' sidebar. \*/

register\_sidebar(

array(

'id' => 'primary',

'name' => \_\_( 'Primary Sidebar' ),

'description' => \_\_( 'A short description of the sidebar.' ),

'before\_widget' => '<div id="%1$s" class="widget %2$s">',

'after\_widget' => '</div>',

'before\_title' => '<h3 class="widget-title">',

'after\_title' => '</h3>',

)

);

/\* Repeat register\_sidebar() code for additional sidebars. \*/

}

**1. Example:**[**#Example:**](https://developer.wordpress.org/themes/functionality/sidebars/#example)

1. Create sidebar-primary.php

2. Add the following code:

<div id="sidebar-primary" class="sidebar">

<?php dynamic\_sidebar( 'primary' ); ?>

</div>

Note that dynamic\_sidebar takes a single parameter of $index, which can be either the sidebar’s name or id.

**2. Load your Sidebar**[**#Load your Sidebar**](https://developer.wordpress.org/themes/functionality/sidebars/#load-your-sidebar)

To load your sidebar in your theme, use the get\_sidebar function. This should be inserted into the template file where you want the sidebar to display. To load the default sidebar.php use the following:

<?php get\_sidebar(); ?>

To display the Primary sidebar, pass the $name parameter to the function:

<?php get\_sidebar( 'primary' ); ?>

**3. Customizing your Sidebar**[**#Customizing your Sidebar**](https://developer.wordpress.org/themes/functionality/sidebars/#customizing-your-sidebar)There are lots of ways that you can customize your sidebars. Here are some examples:

**4. Display Default Sidebar Content**[**#Display Default Sidebar Content**](https://developer.wordpress.org/themes/functionality/sidebars/#display-default-sidebar-content)

You may wish to display content if the user hasn’t added any widgets to the sidebar yet. To do this, you use the is\_sidebar\_active() function to check to see if the sidebar has any widgets. This accepts the $index parameter which should be the ID of the sidebar that you wish to check.

This code checks to see if the sidebar is active, if not it displays some content:

<div id="sidebar-primary" class="sidebar">

<?php if ( is\_active\_sidebar( 'primary' ) ) : ?>

<?php dynamic\_sidebar( 'primary' ); ?>

<?php else : ?>

<!-- Time to add some widgets! -->

<?php endif; ?>

</div>

**5. Display Default Widgets**[**#Display Default Widgets**](https://developer.wordpress.org/themes/functionality/sidebars/#display-default-widgets)

You may want your sidebar to be populated with some widgets by default. For example, display the Search, Archive, and Meta Widgets.  To do this you would use:

<div id="primary" class="sidebar">

<?php do\_action( 'before\_sidebar' ); ?>

<?php if ( ! dynamic\_sidebar( 'sidebar-primary' ) ) : ?>

<aside id="search" class="widget widget\_search">

<?php get\_search\_form(); ?>

</aside>

<aside id="archives" class"widget">

<h3 class="widget-title"><?php \_e( 'Archives', 'shape' ); ?></h3>

<ul>

<?php wp\_get\_archives( array( 'type' => 'monthly' ) ); ?>

</ul>

</aside>

<aside id="meta" class="widget">

<h3 class="widget-title"><?php \_e( 'Meta', 'shape' ); ?></h3>

<ul>

<?php wp\_register(); ?>

<li><?php wp\_loginout(); ?></li>

<?php wp\_meta(); ?>

</ul>

</aside>

<?php endif; ?>

</div>

<https://developer.wordpress.org/themes/functionality/sidebars/>

# Using Action & Filter Hooks

# Simple Usage

A common action hook usage is to add custom content to the page rendered in the browser. This sample code will add a text div tag to the header using the themify\_header\_end hook:

function theme\_header\_end() {

echo '<div class="custom-text">Custom Text</div>';

};

add\_action('themify\_header\_end', 'theme\_header\_end');

To add the content somewhere else, simply specify which action hook name you want to use as the insertion point. For example you can insert the content at the end of the post.

function theme\_post\_end() {

echo '<div class="custom-text">Custom Text at the end of the post</div>';

};

add\_action('themify\_post\_end', 'theme\_post\_end');

# Conditional Tags

Inside the function called by the action hook, you can use WordPress conditional tags to display different content on different pages.

function my\_text\_before\_content() {

if(is\_home()) {

echo 'Text for home page.';

} elseif(is\_page()){

if (is\_page(7)) {

echo 'Text for page with ID 7.';

} else {

echo 'Text for regular page.';

}

} elseif(is\_single()){

echo 'Text for single post.';

} elseif (is\_category()) {

echo 'Text for category archive page.';

}

};

add\_action('themify\_content\_before', 'my\_text\_before\_content');

# Header Banner

Let's take the above samples to a practical use. For example, you can use it to display a header banner image without having to edit the header.php template

.

function add\_banner\_header\_end(){

echo '<div class="header-banner"><img alt="header banner" src="http://example.com/url/image.jpg" /></div>';

}

add\_action('themify\_header\_end', 'add\_banner\_header\_end');

# Conditional Header Banner

To display different header banner on different pages:

function add\_banner\_header\_end(){

if(is\_home())

echo '<div class="header"><img alt="header banner" src="http://test.com/url/img-home.jpg" /></div>';

elseif(is\_page(7))

echo '<div class="header"><img alt="header banner" src="http://test.com/url/img-page-id-7.jpg" /></div>';

elseif(is\_page())

echo '<div class="header"><img alt="header banner" src="http://test.com/url/img-page.jpg" /></div>';

elseif(is\_single())

echo '<div class="header-banner"><img alt="header" src="http://test.com/url/img-single.jpg" /></div>';

elseif(is\_category())

echo '<div class="header-banner"><img alt="header" src="http://test.com/url/img-category.jpg" /></div>';

}

add\_action('themify\_header\_end', 'add\_banner\_header\_end');

# Add Welcome Message

To add a welcome message on the homepage:

function add\_welcome\_message(){

if(is\_home()) {

echo '<div class="welcome-text">Welcome message</div>';

}

};

add\_action('themify\_content\_before', 'add\_welcome\_message');

Then you need to add this to Themify > Styling > Custom CSS to style the .welcome-message class:

.welcome-message {

padding: 15px; font-size: 120%; text-align: center;

}

# Run Shortcodes

You can also run shortcodes in the hooks. The example below will add the [[post\_slider]] shortcode right after the header area on the homepage only.

function show\_post\_slider\_layout\_before() {

echo do\_shortcode('[post\_slider limit="8" visible="4"]');

};

add\_action('themify\_layout\_before', 'show\_post\_slider\_layout\_before');

# Add Header Widget

To add a custom widgetized area, first you need to register it as a sidebar:

function theme\_register\_sidebar(){

register\_sidebar(array(

'name' => 'Custom Header Widget',

'id' => 'custom-header-widget',

'before\_widget' => '

<div class="widget %2$s" id="%1$s">',

'after\_widget' => '</div>',

'before\_title' => '<strong class="widgettitle">',

'after\_title' => '</strong>',

));

}

add\_action( 'widgets\_init', 'theme\_register\_sidebar' );

Now add this code to display the header widget:

function new\_header\_widget(){

echo '<div class="header-widget">';

if ( !function\_exists('dynamic\_sidebar') || !dynamic\_sidebar('custom-header-widget') ) ;

echo '</div>';

}

add\_action('themify\_header\_end', 'new\_header\_widget');

# Run PHP Function

Some plugins requires you to add a PHP function call to the template files. [Breadcrumb NavXT](http://mtekk.us/code/breadcrumb-navxt/#installation) is one of them. You may use hooks to add PHP function without having to edit the template. The sample below will add the breadcrumb function in the post\_start hook of the single post page.

function breadcrumbs\_on\_post\_start(){

if(function\_exists('bcn\_display')){

echo '<div class="breadcrumbs">'; bcn\_display(); echo '</div>';

}

}

add\_action('themify\_post\_start', 'breadcrumbs\_on\_post\_start');

Additionally, if you would like to trigger this only on single views, you would use

function breadcrumbs\_on\_post\_start(){

if( function\_exists('bcn\_display') && is\_single()){

echo '<div class="breadcrumbs">'; bcn\_display(); echo '</div>';

}

}

add\_action('themify\_post\_start', 'breadcrumbs\_on\_post\_start');

Note the additional call to the is\_single() function.

# Usage of filter hooks

As an example, we will change the format of the date shown in the post.

function custom\_date\_format($date){

return 'Y/d/m';

}

add\_filter('themify\_loop\_date', 'custom\_date\_format');

You'll notice a few differences here compared to action hooks. For example, filters always receive at least one variable, and you can modify the content of this variable, of as in the example above, dismiss it by returning something else. Let's see an example where the variable is also returned, along with extra parameter, reversing the order of the posts shown in a Query Category page:

function custom\_post\_order($query){

return $query . '&order=ASC';

}

add\_filter('themify\_query\_posts\_page\_args', 'custom\_post\_order');

#### QUERY POSTS PAGE BY TAGS

To create a Query Posts page that queries posts by their tags instead of their categories, follow these steps:

1. edit or create the page to use as Query Posts  
2. enable Query Posts selecting a category for the Query Category. This is necessary otherwise the page is a regular content page. While you're still in the edit screen, check the browser address bar, look for something like post.php?post=555 and take note of the number, for example, 555.  
3. go to Posts > Tags and take note of the slug of the tag you'll be using, for example sometag.  
4. use the following code in a custom-functions.php file

/\*\* Check if it's a certain page and if so, query by tag instead of by category.

\*

\* @param string **$query** Parameters for the query.

\* @return string \*/

function custom\_themify\_query\_posts\_page\_args( $query ) {

// Check that we're in the page whose ID is, for example, 555, the number we took note before.

if ( is\_page( 555 ) ) {

// Remove the categories to query.

$query = str\_replace( '/cat=(.\*?)&/', '', $query );

// Add the tag slug that you took note before.

$query = 'tag=sometag&' . $query;

}

// Return the query to be performed.

return $query;

}

add\_filter('themify\_query\_posts\_page\_args', 'custom\_themify\_query\_posts\_page\_args');

# Common Problem: Blank Screen

You can write several blocks of PHP code in the custom-functions.php file like:

<?php // function 1?>

<?php // function 2?>

But you must be very careful to avoid leaving blank spaces, like a carriage return or a "space bar" white space before the first <?php in the file or after the last ?> in the file.

# Hooks in Themify themes

Here's a list of the hooks found in all themes:

<body>

<?php themify\_body\_start(); ?>

<div id="pagewrap">

<div id="headerwrap">

<?php themify\_header\_before(); ?>

<header id="header">

<?php themify\_header\_start(); ?>

<?php themify\_header\_end(); ?>

</header>

<?php themify\_header\_after(); ?>

</div>

<div id="body">

<?php themify\_layout\_before(); ?>

<div id="layout">

<?php themify\_content\_before(); ?>

<div id="content">

<?php themify\_content\_start(); ?>

<?php themify\_post\_before(); ?>

<article class="post">

<?php themify\_post\_start(); ?>

<?php themify\_post\_end(); ?>

</article>

<?php themify\_post\_after(); ?>

<?php themify\_comment\_before(); ?>

<div class="commentwrap">

<?php themify\_comment\_start(); ?>

<?php themify\_commentform\_before(); ?>

<div id="commentform">

<?php themify\_commentform\_start(); ?>

<?php themify\_commentform\_end(); ?>

</div>

<?php themify\_commentform\_after(); ?>

<?php themify\_comment\_end(); ?>

</div>

<?php themify\_comment\_after(); ?>

<?php themify\_content\_end(); ?>

</div>

<?php themify\_content\_after(); ?>

<?php themify\_sidebar\_before(); ?>

<div id="sidebar">

<?php themify\_sidebar\_start(); ?>

<?php themify\_sidebar\_end(); ?>

</div>

<?php themify\_sidebar\_after(); ?>

</div>

<?php themify\_layout\_after(); ?>

</div>

<div id="footerwrap">

<?php themify\_footer\_before(); ?>

<footer id="footer">

<?php themify\_footer\_start(); ?>

<?php themify\_footer\_end(); ?>

</footer>

<?php themify\_footer\_after(); ?>

</div>

</div>

<?php themify\_body\_end(); ?>

# </body>

# Hooks in e-commerce themes

The e-commerce themes feature an additional set of hooks:  
Before and after the breadcrumbs

<?php themify\_breadcrumb\_before(); ?>

<div class="breadcrumb breadcrumbs shopdock-breadcrumbs"></div>

<?php themify\_breadcrumb\_after(); ?>

For products

<div class="product product-<?php the\_ID(); ?>"></div>

<div class="product-imagewrap"><figure class="product-image"><?php themify\_product\_image\_start(); ?>

<a> <img alt="" /></a>

<?php themify\_product\_image\_end(); ?></figure></div>

<h3 class="product-title"><?php themify\_product\_title\_start(); ?></h3>

<?php themify\_product\_title\_end(); ?>

For products in carousel

<ul class="product-slides">

<li>

<div class="product product-##">

<div class="product-imagewrap"><figure class="product-image"><?php themify\_product\_image\_start(); ?>

<img alt="" />

<?php themify\_product\_image\_end(); ?></figure></div></div>

</li>

</ul>

<h3 class="product-title"><?php themify\_product\_title\_start(); ?>

<a> <?php the\_title(); ?> </a>

<?php themify\_product\_title\_end(); ?></h3>

<https://themify.me/docs/hooks>

<https://developer.wordpress.org/themes/functionality/administration-menus/>

**Core-Supported Features**

* 1. **Administration Menus**

Theme authors might need to provide a settings screen, so users can customize how their theme is used or works. The best way to do this is to create an administration menu item that allows the user to access that settings screen from all the administration screens.

## Function Reference [#Function Reference](https://developer.wordpress.org/themes/functionality/administration-menus/#function-reference)

# Menu Pages [#Menu Pages](https://developer.wordpress.org/themes/functionality/administration-menus/#menu-pages)

[add\_menu\_page()](https://codex.wordpress.org/Function_Reference/add_menu_page)  
[add\_object\_page()](https://codex.wordpress.org/Function_Reference/add_object_page)  
[add\_utility\_page()](https://codex.wordpress.org/Function_Reference/add_utility_page)  
[remove\_menu\_page()](https://codex.wordpress.org/Function_Reference/remove_menu_page)

# Sub-menu Pages [#Sub-menu Pages](https://developer.wordpress.org/themes/functionality/administration-menus/#sub-menu-pages)

[add\_submenu\_page()](https://codex.wordpress.org/Function_Reference/add_submenu_page)  
[remove\_submenu\_page()](https://codex.wordpress.org/Function_Reference/remove_submenu_page)

# WordPress Administration Menus [#WordPress Administration Menus](https://developer.wordpress.org/themes/functionality/administration-menus/#wordpress-administration-menus)

[add\_dashboard\_page()](https://codex.wordpress.org/Function_Reference/add_dashboard_page)  
[add\_posts\_page()](https://codex.wordpress.org/Function_Reference/add_posts_page)  
[add\_media\_page()](https://codex.wordpress.org/Function_Reference/add_media_page)  
[add\_links\_page()](https://codex.wordpress.org/Function_Reference/add_links_page)  
[add\_pages\_page()](https://codex.wordpress.org/Function_Reference/add_pages_page)  
[add\_comments\_page()](https://codex.wordpress.org/Function_Reference/add_comments_page)  
[add\_theme\_page()](https://codex.wordpress.org/Function_Reference/add_theme_page)  
[add\_plugins\_page()](https://codex.wordpress.org/Function_Reference/add_plugins_page)  
[add\_users\_page()](https://codex.wordpress.org/Function_Reference/add_users_page)  
[add\_management\_page()](https://codex.wordpress.org/Function_Reference/add_management_page)  
[add\_options\_page()](https://codex.wordpress.org/Function_Reference/add_options_page)

## Every Plot Needs a Hook [#Every Plot Needs a Hook](https://developer.wordpress.org/themes/functionality/administration-menus/#every-plot-needs-a-hook)

To add an administration menu, there are three things you need to do:

1. Create a function that contains the menu-building code.
2. Register the above function using the admin\_menu action hook – or network\_admin\_menu, if you are adding a menu for the Network.
3. Create the HTML output for the screen, displayed when the menu item is clicked.

Most developers overlook the second step in this list. You can’t simply call the menu code. You need to put it inside a function, and then register this function.

Here is a simple example of these three steps described. This will add a sub-level menu item under the Settings top-level menu. When selected, that menu item will show a very basic screen.

/\*\* Step 2 (from text above). \*/

add\_action( 'admin\_menu', 'my\_menu' );

/\*\* Step 1. \*/

function my\_menu() {

add\_options\_page( 'My Options', 'My Menu', 'manage\_options', 'my-unique-identifier', 'my\_options' );

}

/\*\* Step 3. \*/

function my\_options() {

if ( !current\_user\_can( 'manage\_options' ) ) {

wp\_die( \_\_( 'You do not have sufficient permissions to access this page.' ) );

}

echo 'Here is where I output the HTML for my screen.';

echo '</div><pre>';

}

In this example, the function my\_menu() adds a new item to the Settings administration menu via the [add\_options\_page()](https://codex.wordpress.org/Function_Reference/add_options_page) function.

Note: Note that the [add\_action()](https://codex.wordpress.org/Function_Reference/add_action) call in Step 2 registers the my\_menu() function under the [admin\_menu](https://codex.wordpress.org/Plugin_API/Action_Reference/admin_menu) hook. Without that, [**add\_action()**](https://developer.wordpress.org/reference/functions/add_action/) call, a PHP error for undefined function will be thrown. Finally, the add\_options\_page() call refers to the my\_options() function which contains the actual page to be displayed (and PHP code to be processed) when someone clicks the menu item.

These steps are described in more detail in the sections below. Remember to enclose the creation of the menu and the page in functions, and use the admin\_menu [hook](https://codex.wordpress.org/Plugin_API/Action_Reference) to get the whole process started at the right time.

# Determining Location for New Menus [#Determining Location for New Menus](https://developer.wordpress.org/themes/functionality/administration-menus/#determining-location-for-new-menus)

Before creating a new menu, first decide if the menu should be a top-level menu, or a sub-level menu item. A top-level menu displays as new section in the administration menus and contains sub-level menu items. This means a sub-level menu item is a member of an existing top-level menu.

It is rare that a theme would require the creation of a new top-level menu. If the theme introduces an entirely new concept to WordPress and needs many screens to do it, then that theme may warrant a new top-level menu. Adding a top-level menu should only be considered if you really need multiple, related screens to make WordPress do something it was not originally designed to accomplish. Examples of new top-level menus might include job management or conference management. Please note that, with the native [post type](https://codex.wordpress.org/Glossary#Post_Type) registration, WordPress automatically creates top-level menus to manage this kind of features.

If the creation of a top-level menu is not necessary, you need to decide under what top-level menu to place your new sub-level menu item. As a point of reference, several themes add sub-level menu items underneath existing WordPress top-level menus.

Use this guide of the WordPress top-level menus to determine the correct location for your sub-level menu item:

* Dashboard – information central for your site and include the Updates option for updating WordPress core, plugins, and themes.
* Posts – displays tools for writing posts (time oriented content).
* Media – uploading and managing your pictures, videos, and audio.
* Links – manage references to other blogs and sites of interest.
* Pages – displays tools for writing your static content called pages.
* Comments – controlling and regulation reader to responses to posts.
* Appearance – displays controls for manipulation of theme/style files, sidebars, etc.
* Plugins – displays controls dealing with plugin management, not configuration options for a plugin itself.
* Users – displays controls for user management.
* Tools – manage the export, import, and even backup of blog data.
* Settings – displays plugin options that only administrators should view.
* Network Admin – displays plugin options that are set on the Network. Instead of admin\_menu, you should use network\_admin\_menu (see also [Create A Network](https://codex.wordpress.org/Create_A_Network))

# Top-Level Menus [#Top-Level Menus](https://developer.wordpress.org/themes/functionality/administration-menus/#top-level-menus)

If you have decided your theme requires a brand-new top-level menu, the first thing you’ll need to do is to create one with the add\_menu\_page() function. Note: skip to [Sub-Level Menus](https://developer.wordpress.org/themes/functionality/administration-menus/#sub-level-menus) if you don’t need a top-level menu.

Parameter values:

* page\_title – the text to be displayed in the title tags of the page when the menu is selected.
* menu\_title – the on-screen name text for the menu.
* capability – the capability required for this menu to be displayed to the user. When using the Settings API to handle your form, you should use manage\_options here as the user won’t be able to save options without it. User levels are deprecated and should not be used here.
* menu\_slug – the slug name to refer to this menu by (should be unique for this menu). Prior to Version 3.0 this was called the file (or handle) parameter. If the function parameter is omitted, the menu\_slug should be the PHP file that handles the display of the menu page content.
* function – the function that displays the page content for the menu page.
* icon\_url – the URL to the icon to be used for this menu. This parameter is optional.
* position – the position in the menu order this menu should appear. By default, if this parameter is omitted, the menu will appear at the bottom of the menu structure. To see the current menu positions, use print\_r( $GLOBALS[ 'menu' ] ) after the menu has loaded.
* Sub-Level Menus – once your top-level menu is defined, or you have chosen to use an existing WordPress top-level menu, you are ready to define one or more sub-level menu items using the add\_submenu\_page() function.

# Sub-Level Menus [#Sub-Level Menus](https://developer.wordpress.org/themes/functionality/administration-menus/#sub-level-menus)

If you want your new menu item to be a sub-menu item, you can create it using the add\_submenu\_page() function instead.

Parameter values:

* parent\_slug – the slug name for the parent menu, or the file name of a standard WordPress admin file that supplies the top-level menu in which you want to insert your sub-menu, or your plugin file if this sub-menu is going into a custom top-level menu. Examples:
  + Dashboard – add\_submenu\_page('index.php', ...)
  + Posts – add\_submenu\_page('edit.php', ...)
  + Media – add\_submenu\_page('upload.php', ...)
  + Links – add\_submenu\_page('link-manager.php', ...)
  + Pages – add\_submenu\_page('edit.php?post\_type=page', ...)
  + Comments – add\_submenu\_page('edit-comments.php', ...)
  + Custom Post Types – add\_submenu\_page('edit.php?post\_type=your\_post\_type', ...)
  + Appearance – add\_submenu\_page('themes.php', ...)
  + Plugins – add\_submenu\_page('plugins.php', ...)
  + Users – add\_submenu\_page('users.php', ...)
  + Tools – add\_submenu\_page('tools.php', ...)
  + Settings – add\_submenu\_page('options-general.php', ...)
* page\_title – text that will go into the HTML page title for the page when the submenu is active.
* menu\_title – the text to be displayed in the title tags of the page when the menu is selected.
* capability – the capability required for this menu to be displayed to the user. User levels are deprecated and should not be used here.
* menu\_slug – for existing WordPress menus, the PHP file that handles the display of the menu page content. For sub-menus of a custom top-level menu, a unique identifier for this sub-menu page.
* function – the function that displays the page content for the menu page. Technically, as in the add\_menu\_page function, the function parameter is optional, but if it is not supplied, then WordPress will basically assume that including the PHP file will generate the administration screen, without calling a function.

# Using Wrapper Functions [#Using Wrapper Functions](https://developer.wordpress.org/themes/functionality/administration-menus/#using-wrapper-functions)

Since most sub-level menus belong under the Settings, Tools, or Appearance menus, WordPress supplies wrapper functions that make adding a sub-level menu items to these top-level menus easier. Note that the function names may not match the names seen in the admin UI as they have changed over time:

//Dashboard - add\_dashboard\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Posts - add\_posts\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Media - add\_media\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Links - add\_links\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Pages - add\_pages\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Comments - add\_comments\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Appearance - add\_theme\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Plugins - add\_plugins\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Users - add\_users\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Tools - add\_management\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Settings - add\_options\_page( $page\_title, $menu\_title, $capability, $menu\_slug, $function);

//Also see Theme Options for the currently recommended way of creating options via the Customizer API.

# Example [#Example](https://developer.wordpress.org/themes/functionality/administration-menus/#example)

Here’s a quick example illustrating how to insert a top-level menu page and a sub-menu page, where the title on the sub-menu page is different from the top-level page. In this example, register\_my\_theme\_more\_settings\_menu is the name of the function that displays the first sub-menu page:

function register\_my\_theme\_settings\_menu() {

add\_menu\_page( "My Theme's Settings", "My Theme", "manage\_options", "my-theme-settings-menu" );

}

function register\_my\_theme\_more\_settings\_menu() {

add\_submenu\_page( "my-themes-settings-menu", "More Settings for My Theme", "More Settings", "manage\_options",

"my-theme-more-settings-menu" );

}

add\_action( "admin\_menu", "register\_my\_theme\_settings\_menu");

add\_action( "admin\_menu", "register\_my\_theme\_more\_settings\_menu");

Here’s an example of adding an option page under a custom post type menu block (see also here):

CODE

# Inserting the Pages [#Inserting the Pages](https://developer.wordpress.org/themes/functionality/administration-menus/#inserting-the-pages)

Here is an example of how to insert multiple menus into various places:

// Hook for adding admin menus

add\_action('admin\_menu', 'mt\_add\_pages');

// Action function for hook above

function mt\_add\_pages() {

// Add a new submenu under Settings:

add\_options\_page(\_\_('Test Settings','menu-test'), \_\_('Test Settings','menu-test'), 'manage\_options', 'testsettings', 'mt\_settings\_page');

// Add a new submenu under Tools:

add\_management\_page( \_\_('Test Tools','menu-test'), \_\_('Test Tools','menu-test'), 'manage\_options', 'testtools', 'mt\_tools\_page');

// Add a new top-level menu (ill-advised):

add\_menu\_page(\_\_('Test Toplevel','menu-test'), \_\_('Test Top-level','menu-test'), 'manage\_options', 'mt-top-level-handle', 'mt\_toplevel\_page' );

// Add a submenu to the custom top-level menu:

add\_submenu\_page('mt-top-level-handle', \_\_('Test Sub-Level','menu-test'), \_\_('Test Sub-Level','menu-test'), 'manage\_options', 'sub-page', 'mt\_sublevel\_page');

// Add a second submenu to the custom top-level menu:

add\_submenu\_page('mt-top-level-handle', \_\_('Test Sub-Level 2','menu-test'), \_\_('Test Sub-Level 2','menu-test'), 'manage\_options', 'sub-page2', 'mt\_sublevel\_page2');

}

// mt\_settings\_page() displays the page content for the Test settings sub-menu

function mt\_settings\_page() {

echo "</pre>

<h2>" . \_\_( 'Test Settings', 'menu-test' ) . "</h2>

<pre>

"; }

// mt\_tools\_page() displays the page content for the Test Tools sub-menu

function mt\_tools\_page() {

echo "</pre>

<h2>" . \_\_( 'Test Tools', 'menu-test' ) . "</h2>

<pre>

"; }

// mt\_toplevel\_page() displays the page content for the custom Test Top-Level menu

function mt\_toplevel\_page() {

echo "</pre>

<h2>" . \_\_( 'Test Top-Level', 'menu-test' ) . "</h2>

<pre>

"; }

// mt\_sublevel\_page() displays the page content for the first sub-menu

// of the custom Test Toplevel menu

function mt\_sublevel\_page() {

echo "</pre>

<h2>" . \_\_( 'Test Sub-Level', 'menu-test' ) . "</h2>

<pre>

"; }

// mt\_sublevel\_page2() displays the page content for the second sub-menu

// of the custom Test Top-Level menu

function mt\_sublevel\_page2() {

echo "</pre>

<h2>" . \_\_( 'Test Sub-Level 2', 'menu-test' ) . "</h2>

<pre>

"; }

# Sample Menu Page [#Sample Menu Page](https://developer.wordpress.org/themes/functionality/administration-menus/#sample-menu-page)

Note: See the [Settings API](https://codex.wordpress.org/Settings_API) for information on creating settings pages.

The previous example contains several dummy functions, such as mt\_settings\_page(), as placeholders for actual page content. Let’s expand on them. What if you wanted to create an option called mt\_favorite\_color that allows the site owner to type in their favorite color via a Settings page? The mt\_options\_page() function will need to output a data entry form on the screen, as well as to process the entered data.

Here is a function that does this:

// mt\_settings\_page() displays the page content for the Test settings sub-menu

function mt\_settings\_page() {

//must check that the user has the required capability

if (!current\_user\_can('manage\_options'))

{

wp\_die( \_\_('You do not have sufficient permissions to access this page.') );

}

// Variables for the field and option names

$opt\_name = 'mt\_favorite\_color';

$hidden\_field\_name = 'mt\_submit\_hidden';

$data\_field\_name = 'mt\_favorite\_color';

// Read in existing option value from database

$opt\_val = get\_option( $opt\_name );

// See if the user has posted us some information

// If they did, this hidden field will be set to 'Y'

if( isset($\_POST[ $hidden\_field\_name ]) && $\_POST[ $hidden\_field\_name ] == 'Y' ) {

// Read their posted value

$opt\_val = $\_POST[ $data\_field\_name ];

// Save the posted value in the database

update\_option( $opt\_name, $opt\_val );

// Put a "Settings updated" message on the screen ?>

<div class="updated"></div>

<div class="wrap">

<?php // Header

echo "<h2>" . \_\_( 'Menu Test Settings', 'menu-test' ) . "</h2>";

// Settings form ?>

<form action="" method="post" name="form1"></form>

<?php \_e("Favorite Color:", 'menu-test' ); ?>

<hr />

</div>

<?php } ?>

A few notes:

* + The WordPress functions such as add\_menu\_page() and add\_submenu\_page() take a capability which will be used to determine whether the top-level or sub-level menu is displayed.
  + The function which is hooked in to handle the output of the page must check that the user has the required capability as well.
  + The WordPress administration functions take care of validating the user login, so you don’t have to worry about it in your function.
  + The function example above has been internationalized — see the [I18n for WordPress Developers](https://codex.wordpress.org/I18n_for_WordPress_Developers) for more details.
  + The function processes any entered data before putting the data entry form on the screen, so that the new values will be shown in the form (rather than the values from the database).
  + You don’t have to worry about this working the first time, because the WordPress update\_option function will automatically add an option to the database if it doesn’t already exist.
  + These admin-menu-adding procedures are parsed every single time you navigate to a page in Admin. So if you are writing a theme which has no options page, but add one later, you can just add it using the instructions above and re-upload, and tweak until you’re happy with it. In other words, menus are not “permanently added” or put into a database upon activating a theme. They’re parsed on the fly, so you can add or subtract menu items at will, re-upload, and the change will be reflected right away.

# Page Hook Suffix [#Page Hook Suffix](https://developer.wordpress.org/themes/functionality/administration-menus/#page-hook-suffix)

Every function that adds a new administration menu – add\_menu\_page(), add\_submenu\_page() and its specialized versions such as add\_options\_page() – returns a special value called Page Hook Suffix. It can be used later as a hook to which an action called only on that particular page can be registered.

One such action hook is load-{page\_hook}, where {page\_hook} is the value returned by one of these add\_\*\_page() functions. This hook is called when the particular page is loaded. In the example below, it is used to display the “Theme is not configured” notice on all admin pages except for plugin’s options page:

add\_action('admin\_menu', 'my\_menu');

// Here you can check if plugin is configured (e.g. check if some option is set). If not, add new hook.

// In this example hook is always added.

add\_action( 'admin\_notices', 'my\_admin\_notices' );

function my\_menu() {

// Add the new admin menu and page and save the returned hook suffix

$hook\_suffix = add\_options\_page('My Options', 'My Theme', 'manage\_options', 'my-unique-identifier', 'my\_options');

// Use the hook suffix to compose the hook and register an action executed when plugin's options page is loaded

add\_action( 'load-' . $hook\_suffix , 'my\_load\_function' );

}

function my\_load\_function() {

// Current admin page is the options page for our plugin, so do not display the notice

// (remove the action responsible for this)

remove\_action( 'admin\_notices', 'my\_admin\_notices' );

}

function my\_admin\_notices() {

echo '<pre>

<div class="updated fade" id="notice">

My Plugin is not configured yet. Please do it now.</div>

</pre>';

}

function my\_options() {

if (!current\_user\_can('manage\_options')) {

wp\_die( \_\_('You do not have sufficient permissions to access this page.') );

}

echo '</pre>

<div class="wrap">';

echo 'Here is where the form would go if I actually had options.';

echo '</div>

<pre>

';

}

<https://wpengineer.com/2139/adding-settings-to-an-existing-page-using-the-settings-api/>

// Register and define the settings

// https://wpengineer.com/2139/adding-settings-to-an-existing-page-using-the-settings-api/

add\_action('admin\_init', 'ozhwpe\_admin\_init');

function ozhwpe\_admin\_init(){

register\_setting(

'writing', // settings page

'ozhwpe\_options', // option name

'ozhwpe\_validate\_options' // validation callback

);

add\_settings\_field(

'ozhwpe\_notify\_boss', // id

'Boss Email', // setting title

'ozhwpe\_setting\_input', // display callback

**'writing', // settings page**

'default' // settings section

);

}

// Display and fill the form field

function ozhwpe\_setting\_input() {

// get option 'boss\_email' value from the database

$options = get\_option( 'ozhwpe\_options' );

$value = $options['boss\_email'];

// echo the field

?>

<input id='boss\_email' name='ozhwpe\_options[boss\_email]'

type='text' value='<?php echo esc\_attr( $value ); ?>' /> Boss wants to get a mail <?php

}

### Core Settings Pages and Sections Names

We've decided to target the 'default' section of the 'writing' Settings page, but of course it's possible to add form fields to all other existing page. Here is a comprehensive list of all these places:

| SETTINGS PAGE | SETTINGS NAME | SECTIONS |
| --- | --- | --- |
| [General Settings](http://yourblog.com/wp-admin/options-general.php) | 'general' | 'default' |
| [Writing Settings](http://yourblog.com/wp-admin/options-writing.php) | 'writing' | 'default' 'remote\_plublishing' 'post\_via\_email' |
| [Reading Settings](http://yourblog.com/wp-admin/options-reading.php) | 'reading' | 'default' |
| [Discussion Settings](http://yourblog.com/wp-admin/options-discussion.php) | 'discussion' | 'default' 'avatars' |
| [Media Settings](http://yourblog.com/wp-admin/options-media.php) | 'discussion' | 'default' 'embeds' 'uploads' |
| [Privacy Settings](http://yourblog.com/wp-admin/options-privacy.php) | 'privacy' | 'default' |
| [Permalink Settings](http://yourblog.com/wp-admin/options-permalink.php) | 'permalink' | 'optional' |

<https://developer.wordpress.org/themes/getting-started/>

## CHAPTERS

* [Getting Started](https://developer.wordpress.org/themes/getting-started/)
  + [Who Should Read This Handbook?](https://developer.wordpress.org/themes/getting-started/who-should-read-this-handbook/)
  + [What is a Theme?](https://developer.wordpress.org/themes/getting-started/what-is-a-theme/)
  + [WordPress Licensing & the GPL](https://developer.wordpress.org/themes/getting-started/wordpress-licensing-the-gpl/)
  + [Setting up a Development Environment](https://developer.wordpress.org/themes/getting-started/setting-up-a-development-environment/)
  + [Theme Development Examples](https://developer.wordpress.org/themes/getting-started/theme-development-examples/)
* [Theme Basics](https://developer.wordpress.org/themes/basics/)
  + [Template Files](https://developer.wordpress.org/themes/basics/template-files/)
    - [Main Stylesheet (style.css)](https://developer.wordpress.org/themes/basics/main-stylesheet-style-css/)
  + [Post Types](https://developer.wordpress.org/themes/basics/post-types/)
  + [Organizing Theme Files](https://developer.wordpress.org/themes/basics/organizing-theme-files/)
  + [Template Hierarchy](https://developer.wordpress.org/themes/basics/template-hierarchy/)
  + [Template Tags](https://developer.wordpress.org/themes/basics/template-tags/)
  + [The Loop](https://developer.wordpress.org/themes/basics/the-loop/)
  + [Theme Functions](https://developer.wordpress.org/themes/basics/theme-functions/)
  + [Linking Theme Files & Directories](https://developer.wordpress.org/themes/basics/linking-theme-files-directories/)
  + [Including CSS & JavaScript](https://developer.wordpress.org/themes/basics/including-css-javascript/)
  + [Conditional Tags](https://developer.wordpress.org/themes/basics/conditional-tags/)
  + [Categories, Tags, & Custom Taxonomies](https://developer.wordpress.org/themes/basics/categories-tags-custom-taxonomies/)
* [Template Files Section](https://developer.wordpress.org/themes/template-files-section/)
  + [Post Templates](https://developer.wordpress.org/themes/template-files-section/post-template-files/)
  + [Page Templates](https://developer.wordpress.org/themes/template-files-section/page-template-files/)
  + [Attachment Template Files](https://developer.wordpress.org/themes/template-files-section/attachment-template-files/)
  + [Custom Post Type Template Files](https://developer.wordpress.org/themes/template-files-section/custom-post-type-template-files/)
  + [Partial and Miscellaneous Template Files](https://developer.wordpress.org/themes/template-files-section/partial-and-miscellaneous-template-files/)
    - [Comment Template](https://developer.wordpress.org/themes/template-files-section/partial-and-miscellaneous-template-files/comment-template/)
  + [Taxonomy Templates](https://developer.wordpress.org/themes/template-files-section/taxonomy-templates/)
  + [404 Pages](https://developer.wordpress.org/themes/functionality/404-pages/)
* [Theme Functionality](https://developer.wordpress.org/themes/functionality/)
  + [Core-Supported Features](https://developer.wordpress.org/themes/functionality/core-supported/)
    - [Administration Menus](https://developer.wordpress.org/themes/functionality/administration-menus/)
    - [Custom Headers](https://developer.wordpress.org/themes/functionality/custom-headers/)
    - [Custom Logo](https://developer.wordpress.org/themes/functionality/custom-logo/)
    - [Post Formats](https://developer.wordpress.org/themes/functionality/post-formats/)
    - [Sticky Posts](https://developer.wordpress.org/themes/functionality/sticky-posts/)
  + [Sidebars](https://developer.wordpress.org/themes/functionality/sidebars/)
  + [Widgets](https://developer.wordpress.org/themes/functionality/widgets/)
  + [Navigation Menus](https://developer.wordpress.org/themes/functionality/navigation-menus/)
  + [Pagination](https://developer.wordpress.org/themes/functionality/pagination/)
  + [Media](https://developer.wordpress.org/themes/functionality/media/)
    - [Audio](https://developer.wordpress.org/themes/functionality/media/audio/)
    - [Images](https://developer.wordpress.org/themes/functionality/media/images/)
    - [Galleries](https://developer.wordpress.org/themes/functionality/media/galleries/)
    - [Video](https://developer.wordpress.org/themes/functionality/media/video/)
  + [Featured Images & Post Thumbnails](https://developer.wordpress.org/themes/functionality/featured-images-post-thumbnails/)
  + [Internationalization](https://developer.wordpress.org/themes/functionality/internationalization/)
  + [Localization](https://developer.wordpress.org/themes/functionality/localization/)
  + [Accessibility](https://developer.wordpress.org/themes/functionality/accessibility/)
* [Theme Options – The Customizer API](https://developer.wordpress.org/themes/customize-api/)
  + [Customizer Objects](https://developer.wordpress.org/themes/customize-api/customizer-objects/)
  + [Tools for Improved User Experience](https://developer.wordpress.org/themes/customize-api/tools-for-improved-user-experience/)
  + [The Customizer JavaScript API](https://developer.wordpress.org/themes/customize-api/the-customizer-javascript-api/)
  + [JavaScript/Underscore.js-Rendered Custom Controls](https://developer.wordpress.org/themes/customize-api/javascriptunderscore-js-rendered-custom-controls/)
  + [Advanced Usage](https://developer.wordpress.org/themes/customize-api/advanced-usage/)
* [Theme Security](https://developer.wordpress.org/themes/theme-security/)
  + [Data Sanitization/Escaping](https://developer.wordpress.org/themes/theme-security/data-sanitization-escaping/)
  + [Data Validation](https://developer.wordpress.org/themes/theme-security/data-validation/)
  + [Using Nonces](https://developer.wordpress.org/themes/theme-security/using-nonces/)
  + [Common Vulnerabilities](https://developer.wordpress.org/themes/theme-security/common-vulnerabilities/)
* [Theme Privacy](https://developer.wordpress.org/themes/theme-privacy/)
* [Advanced Theme Topics](https://developer.wordpress.org/themes/advanced-topics/)
  + [Child Themes](https://developer.wordpress.org/themes/advanced-topics/child-themes/)
  + [UI Best Practices](https://developer.wordpress.org/themes/advanced-topics/ui-best-practices/)
  + [JavaScript Best Practices](https://developer.wordpress.org/themes/advanced-topics/javascript-best-practices/)
  + [Theme Testing](https://developer.wordpress.org/themes/advanced-topics/theme-testing/)
  + [Validating Your Theme](https://developer.wordpress.org/themes/advanced-topics/validating-your-theme/)
  + [Plugin API Hooks](https://developer.wordpress.org/themes/advanced-topics/plugin-api-hooks/)
* [Releasing Your Theme](https://developer.wordpress.org/themes/release/)
  + [Required Theme Files](https://developer.wordpress.org/themes/release/required-theme-files/)
  + [Testing](https://developer.wordpress.org/themes/release/testing/)
  + [Theme Review Guidelines](https://developer.wordpress.org/themes/release/theme-review-guidelines/)
  + [Writing Documentation](https://developer.wordpress.org/themes/release/writing-documentation/)
  + [Submitting Your Theme to WordPress.org](https://developer.wordpress.org/themes/release/submitting-your-theme-to-wordpress-org/)
* [References](https://developer.wordpress.org/themes/references/)
  + [List of Template Tags](https://developer.wordpress.org/themes/references/list-of-template-tags/)
  + [List of Conditional Tags](https://developer.wordpress.org/themes/references/list-of-conditional-tags/)

Default WordPress Generated CSS Cheat Sheet for Beginners

[[https://pdf.printfriendly.com/camo/a9144473692a55629dbe7f7dbc5da37608da42ce/68747470733a2f2f73322e676f6f676c6575736572636f6e74656e742e636f6d2f73322f66617669636f6e733f646f6d61696e3d7777772e7770626567696e6e65722e636f6d](http://www.wpbeginner.com/wp-themes/default-wordpress-generated-css-cheat-sheet-for-beginners/)**wpbeginner.com**/wp-themes/default-wordpress-generated-css-cheat-sheet-for-beginners/](http://www.wpbeginner.com/wp-themes/default-wordpress-generated-css-cheat-sheet-for-beginners/)

**Default Body Class Styles**

The most important of these is the <body> tag. Here are some examples of common classes that WordPress might add to this element:

.rtl {}

.home {}

.blog {}

.archive {}

.date {}

.search {}

.paged {}

.attachment {}

.error404 {}

.single postid-(id) {}

.attachmentid-(id) {}

.attachment-(mime-type) {}

.author {}

.author-(user\_nicename) {}

.category {}

.category-(slug) {}

.tag {}

.tag-(slug) {}

.page-parent {}

.page-child parent-pageid-(id) {}

.page-template page-template-(template file name) {}

.search-results {}

.search-no-results {}

.logged-in {}

.paged-(page number) {}

.single-paged-(page number) {}

.page-paged-(page number) {}

.category-paged-(page number) {}

.tag-paged-(page number) {}

.date-paged-(page number) {}

.author-paged-(page number) {}

.search-paged-(page number) {}

**Default Post Styles**

.post-id {}

.post {}

.page {}

.attachment {}

.sticky {}

.hentry {}

.category-misc {}

.category-example {}

.tag-news {}

.tag-wordpress {}

.tag-markup {}

.format-image {}

.format-gallery {}

.format-chat {}

.format-link {}

.format-quote {}

.format-status {}

.format-video {}

**Default Menu Styles**

In our article called [how to style WordPress navigation menus](https://www.wpbeginner.com/wp-themes/how-to-style-wordpress-navigation-menus/) we discuss how you can go about adding your own class to your menu. We are going to assume you have read it and that you gave your nav menu its own class name of “main-menu”.

#header .main-menu{} // container class

#header .main-menu ul {} // container class first unordered list

#header .main-menu ul ul {} //unordered list within an unordered list

#header .main-menu li {} // each navigation item

#header .main-menu li a {} // each navigation item anchor

#header .main-menu li ul {} // unordered list if there is drop down items

#header .main-menu li li {} // each drop down navigation item

#header .main-menu li li a {} // each drap down navigation item anchor

.current\_page\_item{} // Class for Current Page

.current-cat{} // Class for Current Category

.current-menu-item{} // Class for any other current Menu Item

.menu-item-type-taxonomy{} // Class for a Category

.menu-item-type-post\_type{} // Class for Pages

.menu-item-type-custom{} // Class for any custom item that you added

.menu-item-home{} // Class for the Home Link

Notice that whenever you create a menu in WordPress it is automatically wrapped in a div. This div only has a class name if you specify it (we chose “main-menu”). From there you are just styling the various list elements.

**Default WISIWYG Editor Styles**

.entry-content img {}

.alignleft, img.alignleft {}

.alignright, img.alignright {}

.aligncenter, img.aligncenter {}

.alignnone, img.alignnone {}

.wp-caption {}

.wp-caption img {}

.wp-caption p.wp-caption-text {}

.wp-smiley {}

blockquote.left {}

blockquote.right {}

.gallery dl {}

.gallery dt {}

.gallery dd {}

.gallery dl a {}

.gallery dl img {}

.gallery-caption {}

.size-full {}

.size-large {}

.size-medium {}

.size-thumbnail {}

You can see that there are several class pertaining just to images. If, for example, the user decides to include a left-aligned image then WordPress will add the “alignleft” class.

**Default WordPress Widget Styles**

.widget {}

#searchform {}

.widget\_search {}

.screen-reader-text {}

.widget\_meta {}

.widget\_meta ul {}

.widget\_meta ul li {}

.widget\_meta ul li a {}

.widget\_links {}

.widget\_links ul {}

.widget\_links ul li {}

.widget\_links ul li a {}

.widget\_archive {}

.widget\_archive ul {}

.widget\_archive ul li {}

.widget\_archive ul li a {}

.widget\_archive select {}

.widget\_archive option {}

.widget\_pages {}

.widget\_pages ul {}

.widget\_pages ul li {}

.widget\_pages ul li a {}

.widget\_links {}

.widget\_links li:after {}

.widget\_links li:before {}

.widget\_tag\_cloud {}

.widget\_tag\_cloud a {}

.widget\_tag\_cloud a:after {}

.widget\_tag\_cloud a:before {}

.widget\_calendar {}

#calendar\_wrap {}

#calendar\_wrap th {}

#calendar\_wrap td {}

#wp-calendar tr td {}

#wp-calendar caption {}

#wp-calendar a {}

#wp-calendar #today {}

#wp-calendar #prev {}

#wp-calendar #next {}

#wp-calendar #next a {}

#wp-calendar #prev a {}

.widget\_categories {}

.widget\_categories ul {}

.widget\_categories ul li {}

.widget\_categories ul ul.children {}

.widget\_categories a {}

.widget\_categories select{}

.widget\_categories select#cat {}

.widget\_categories select.postform {}

.widget\_categories option {}

.widget\_categories .level-0 {}

.widget\_categories .level-1 {}

.widget\_categories .level-2 {}

.widget\_categories .level-3 {}

.recentcomments {}

#recentcomments {}

#recentcomments li {}

#recentcomments li a {}

.widget\_recent\_comments {}

.widget\_recent\_entries {}

.widget\_recent\_entries ul {}

.widget\_recent\_entries ul li {}

.widget\_recent\_entries ul li a {}

.textwidget {}

.widget\_text {}

.textwidget p {}

**Default Comment Form Styles**

/\*Comment Output\*/

.commentlist .reply {}

.commentlist .reply a {}

.commentlist .alt {}

.commentlist .odd {}

.commentlist .even {}

.commentlist .thread-alt {}

.commentlist .thread-odd {}

.commentlist .thread-even {}

.commentlist li ul.children .alt {}

.commentlist li ul.children .odd {}

.commentlist li ul.children .even {}

.commentlist .vcard {}

.commentlist .vcard cite.fn {}

.commentlist .vcard span.says {}

.commentlist .vcard img.photo {}

.commentlist .vcard img.avatar {}

.commentlist .vcard cite.fn a.url {}

.commentlist .comment-meta {}

.commentlist .comment-meta a {}

.commentlist .commentmetadata {}

.commentlist .commentmetadata a {}

.commentlist .parent {}

.commentlist .comment {}

.commentlist .children {}

.commentlist .pingback {}

.commentlist .bypostauthor {}

.commentlist .comment-author {}

.commentlist .comment-author-admin {}

.commentlist {}

.commentlist li {}

.commentlist li p {}

.commentlist li ul {}

.commentlist li ul.children li {}

.commentlist li ul.children li.alt {}

.commentlist li ul.children li.byuser {}

.commentlist li ul.children li.comment {}

.commentlist li ul.children li.depth-{id} {}

.commentlist li ul.children li.bypostauthor {}

.commentlist li ul.children li.comment-author-admin {}

#cancel-comment-reply {}

#cancel-comment-reply a {}

/\*Comment Form \*/

#respond { }

#reply-title { }

#cancel-comment-reply-link { }

#commentform { }

#author { }

#email { }

#url { }

#comment

#submit

.comment-notes { }

.required { }

.comment-form-author { }

.comment-form-email { }

.comment-form-url { }

.comment-form-comment { }

.form-allowed-tags { }

.form-submit

<https://gist.github.com/chasereeves/3885925/revisions?diff=split>

<http://hookr.io/>

<http://hookr.io/4.7.4/shortcodes/#index=a>

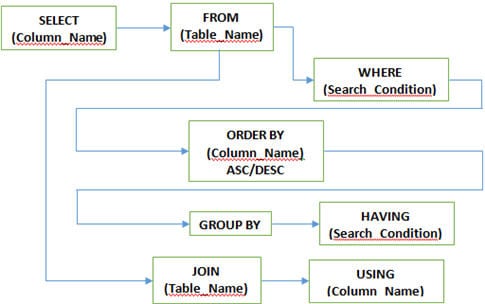
<https://www.oodlestechnologies.com/blogs/A-Sample-Of-Creating-theme-option-with-WordPress/>

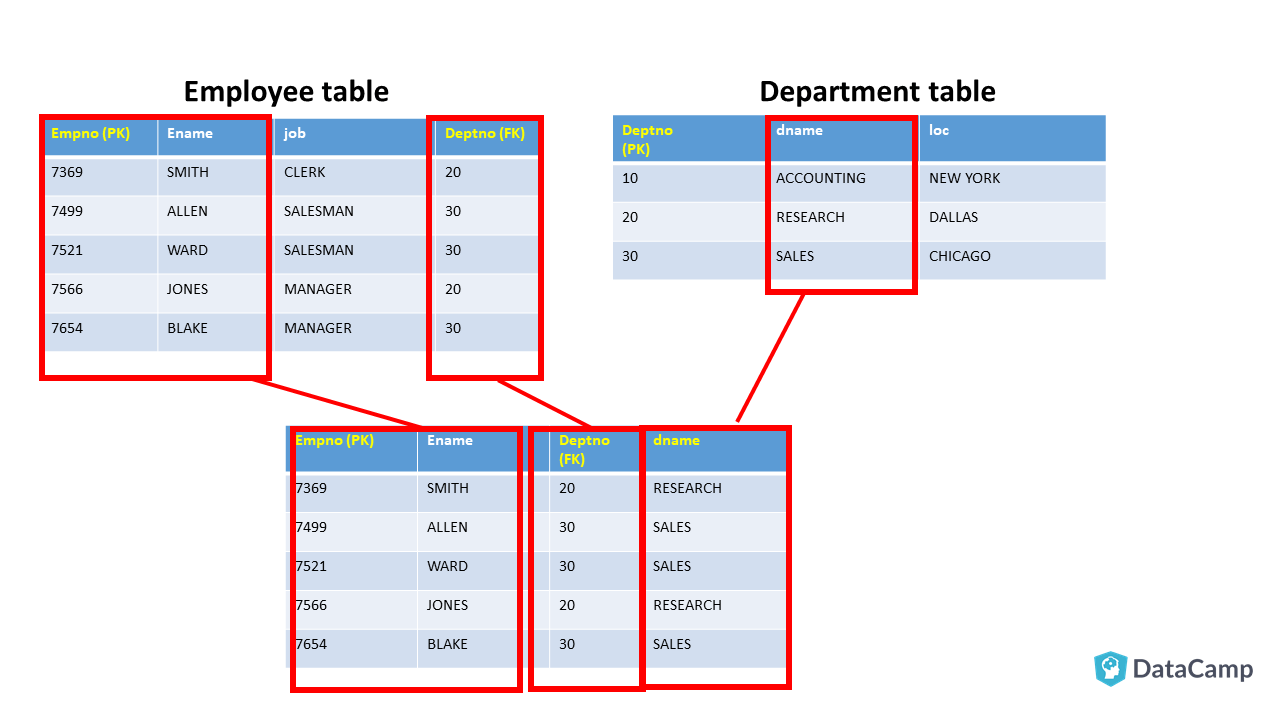
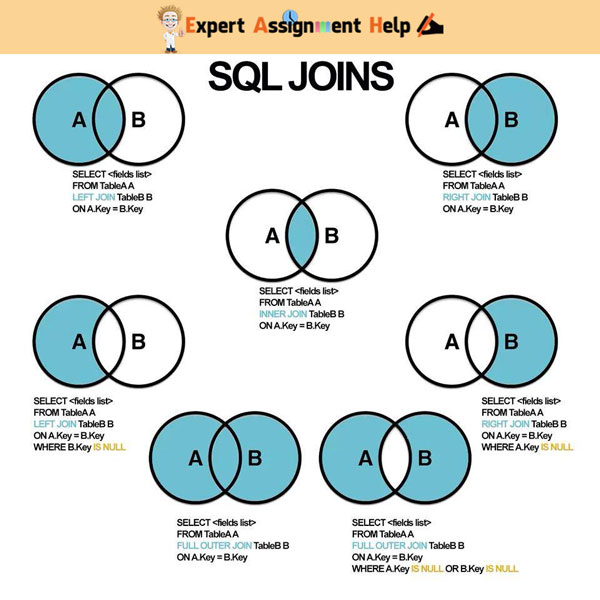
<https://hotexamples.com/examples/-/-/add_theme_page/php-add_theme_page-function-examples.html>

<https://webdesignerwall.com/tutorials/building-custom-wordpress-theme>

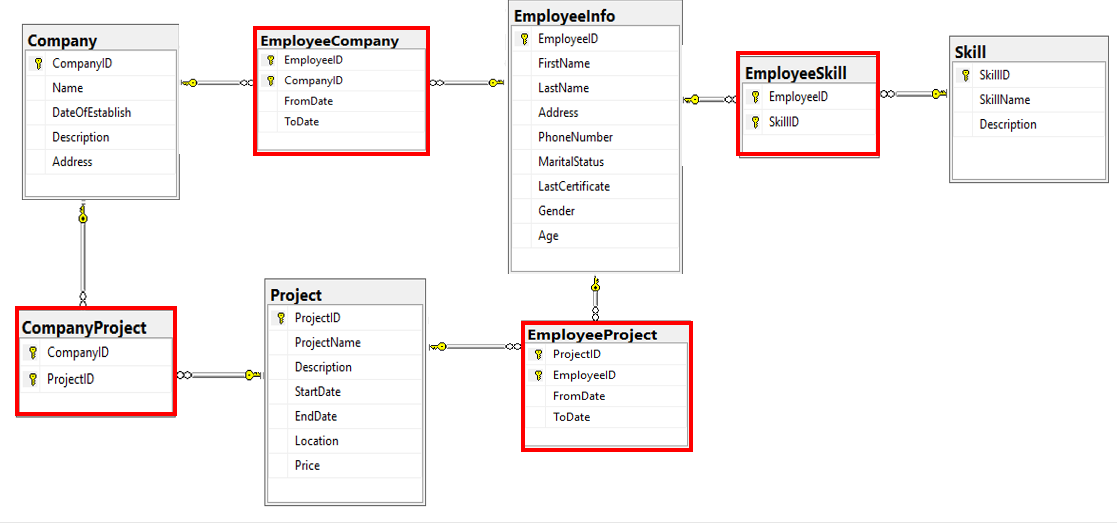
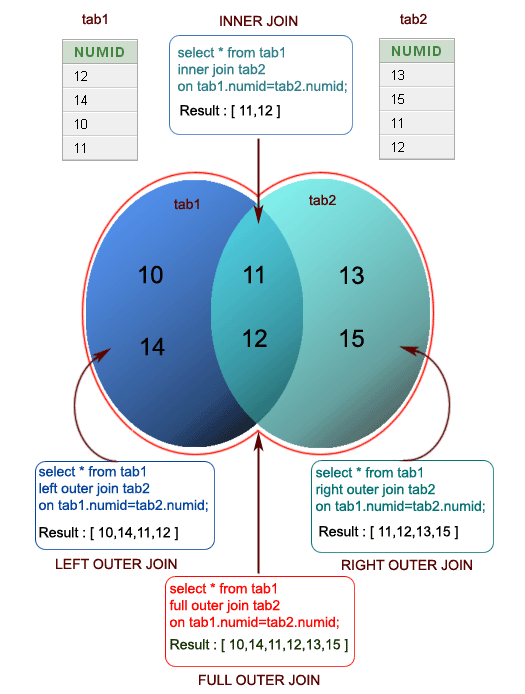
<https://premium.wpmudev.org/blog/wordpress-theme-customizer-guide/>

SQL





<https://www.w3resource.com/sql/joins/sql-joins.php>



use group\_concat() and sub-queries to get your desired result.

**EmployeeId = e.id (id of employee)**

**EmployeeName = e.name (name of the employee)**

**Skills = skills (all skill that employee have)**

**Companies = companies (all companies in which employee is working/worked)**

**Projects = projects (all projects in which employee's company participeted)**

select e.id, e.name,

( select GROUP\_CONCAT(s.name)

from skill s left join employee\_skill es on es.skill\_id = s.id

where es.employee\_id = e.id

) as skills,

( select GROUP\_CONCAT(c.name)

from company c left join employee\_company ec on ec.company\_id = c.id

where ec.employee\_id = e.id

)as companies,

( select GROUP\_CONCAT(p.name)

from project p left join company\_project cp on cp.project\_id = p.id

where cp.company\_id in(

select id from employee\_company where employee\_company.employee\_id = e.id

)

)as projects

from employee\_info e

--UPDATE Same task in SQL SERVER

select e.id, e.name,

skills = stuff(( SELECT ',' + s.name

from skill s left join employee\_skill es on es.skill\_id = s.id

where es.employee\_id = e.id

for xml path(''), type).value('.', 'nvarchar(max)'), 1, 1, ''),

companies = stuff(( SELECT ',' + c.name

from company c left join employee\_company ec on ec.company\_id = c.id

where ec.employee\_id = e.id

for xml path(''), type).value('.', 'nvarchar(max)'), 1, 1, ''),

projects = stuff(( SELECT ',' + p.name

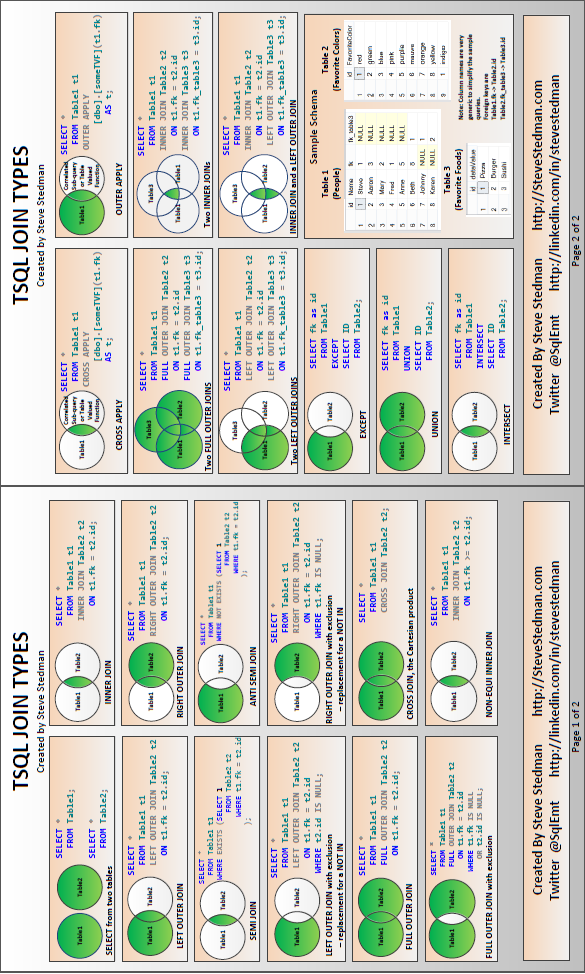
from project p left join company\_project cp on cp.project\_id = p.id

where cp.company\_id in(

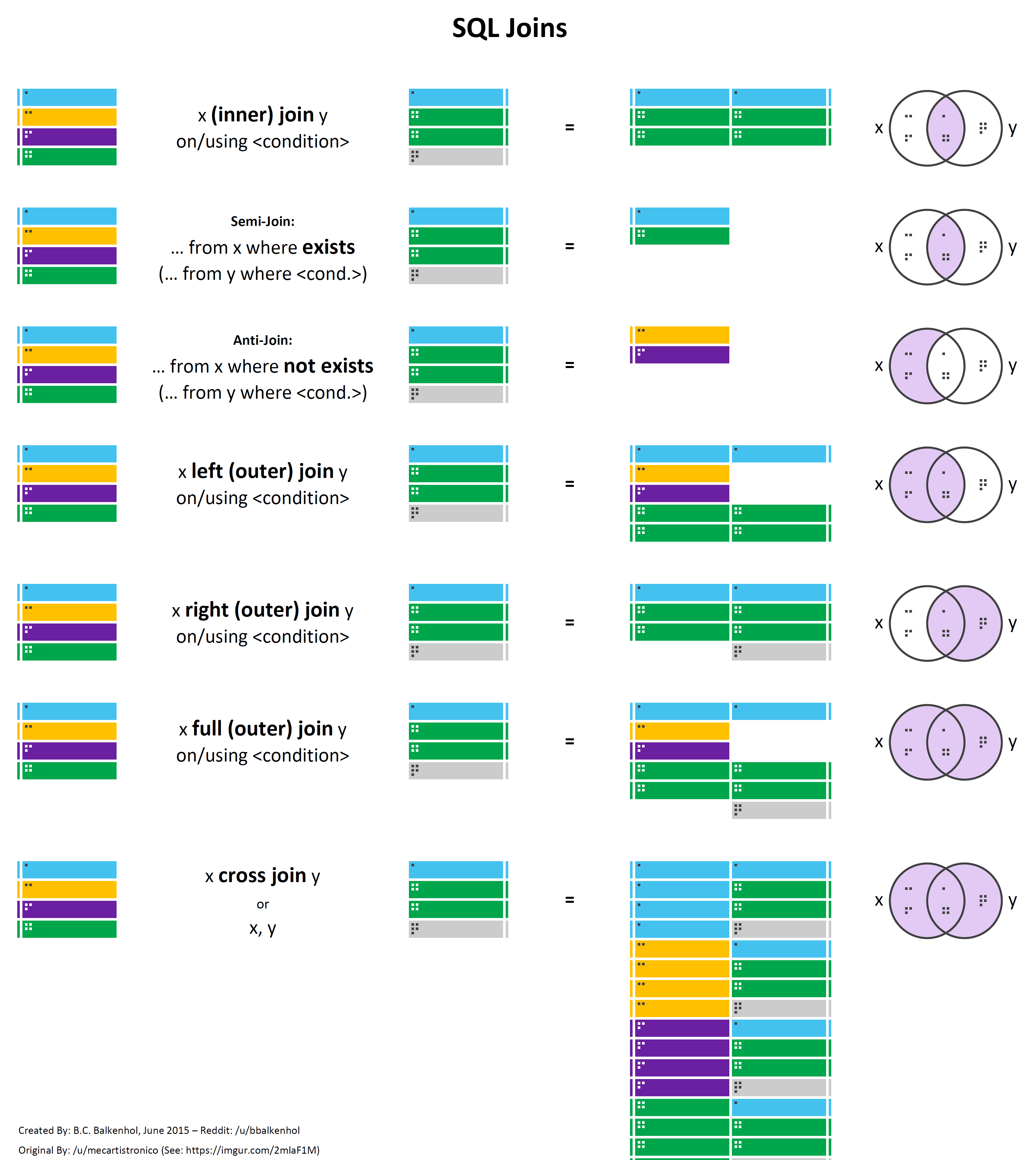
select id from employee\_company where employee\_company.employee\_id = e.id )

for xml path(''), type).value('.', 'nvarchar(max)'), 1, 1, '')

from employee\_info e



<https://visualizeright.com/2019/03/15/joins-and-advanced-joins/>



Examples

--https://www.datacamp.com/community/tutorials/sql-reporting-analysis

SELECT \* FROM employee;

SELECT ename, sal\*(20/100) FROM employee;

SELECT ename, sal+COALESCE(comm,0) FROM employee;

SELECT ename AS "Emp Name", sal\*(20/100) as "20% of Salary" FROM employee;

SELECT deptno FROM employee;

SELECT distinct ename, deptno, job FROM employee;

SELECT ename, job FROM employee WHERE job='CLERK';

SELECT ename FROM employee WHERE job='MANAGER' AND deptno=30;

SELECT ename FROM employee WHERE job='MANAGER' OR deptno=30;

SELECT ename, job from employee WHERE NOT job='SALESMAN';

SELECT ename, job from employee WHERE NOT job='SALESMAN' AND sal>800;

SELECT \* FROM employee WHERE sal BETWEEN 1000 AND 2000;

SELECT \* FROM employee WHERE ename LIKE 'B%';

SELECT \* FROM employee WHERE ename LIKE 'A%E%';

SELECT ename, job, hiredate FROM employee WHERE job IN ('CLERK','SALESMAN');

SELECT ename, job, hiredate FROM employee WHERE deptno IN (select deptno FROM department WHERE loc='CHICAGO');

SELECT ename, job, sal FROM employee WHERE comm IS NULL;

SELECT ename, job, sal, comm FROM employee WHERE comm IS NOT NULL;

SELECT ename, job, sal, comm FROM employee WHERE hiredate>'21-FEB-81';

SELECT ename, job, sal, comm FROM employee WHERE deptno=ANY(SELECT deptno from dept WHERE loc='NEW YORK');

SELECT ename, job, sal, comm FROM employee WHERE deptno=ALL(SELECT deptno from dept WHERE dname='SALES');

SELECT ename, job, sal, comm FROM employee WHERE hiredate>'21-FEB-81' ORDER BY sal desc;

SELECT ename, job, sal, comm, deptno FROM employee WHERE hiredate>'21-FEB-81' ORDER BY deptno ASC, ename DESC;

SELECT lower(ename) as enamelower, upper(ename) as enameupper, LENGTH(ename) as len\_ename FROM employee;

SELECT SUBSTR(ename,2,3) as substr\_ename FROM employee;

SELECT count(ename) FROM employee WHERE ename LIKE 'A%';

SELECT sum(sal) as "TOTAL SAL", avg(sal) as "AVG SAL", min(sal) as "MIN SAL", max(sal) as "MAX SAL", deptno FROM employee

SELECT sum(sal) as "TOTAL SAL", deptno, job FROM employee GROUP BY deptno, job;

SELECT sum(sal) as "TOTAL SAL", deptno, job FROM employee GROUP BY deptno, job HAVING sum(sal)>1000 ORDER BY sum(sal);

SELECT empno, ename, dname FROM employee, department; -- OR

SELECT empno, ename, dname FROM employee CROSS JOIN department;

SELECT ename, dname FROM employee e,department d WHERE e.deptno=d.deptno; --OR

SELECT ename, dname FROM employee e JOIN department d ON e.deptno=d.deptno;

SELECT e.ename, e.sal, s.grade FROM employee e, salgrade s WHERE e.sal between s.losal AND s.hisal;

SELECT e.ename, e.sal, d.dname, s.grade FROM employee e, department d, salgrade s WHERE e.deptno=d.deptno AND e.sal BETWEEN s.losal AND s.hisal;

SELECT e.ename, d.dname, e.sal FROM employee e JOIN department d USING (deptno) WHERE deptno=20;

SELECT e.ename, d.dname, e.sal FROM employee e JOIN department d USING (d.deptno) WHERE d.deptno=20;

SELECT e.ename, s.grade FROM salgrade s LEFT OUTER JOIN employee e ON e.sal BETWEEN s.losal AND s.hisal;

SELECT e.ename, s.grade FROM employee e RIGHT OUTER JOIN department d ON e.deptno=d.deptno;

SELECT e.ename, s.grade FROM employee e FULL OUTER JOIN department d ON e.deptno=d.deptno;

SELECT e.ename as Employee, m.ename as Manager FROM employee e, employee m WHERE e.mgr=m.empno;

SELECT empno, ename FROM employee WHERE sal>(SELECT sal from employee where ename='JAMES');

SELECT ename, job FROM employee WHERE job=(SELECT job FROM employee WHERE empno=7521);

SELECT deptno, max(sal) FROM employee GROUP BY deptno HAVING max(sal)>=(SELECT max(sal) FROM employee WHERE deptno=20); --Single-row Subquery

SELECT ename, job, sal FROM employee WHERE sal IN (SELECT sal FROM employee WHERE job='MANAGER'); --Multiple-row Subquery

SELECT dname FROM department --UNION

UNION

SELECT dname FROM department, employee WHERE department.deptno=employee.deptno AND employee.hiredate<to\_date('23-OCT-1999');

--UNION ALLand cannor use DISTINCT

SELECT empno, ename, job FROM employee UNION ALL SELECT empno, ename, job

-- INTERSECT

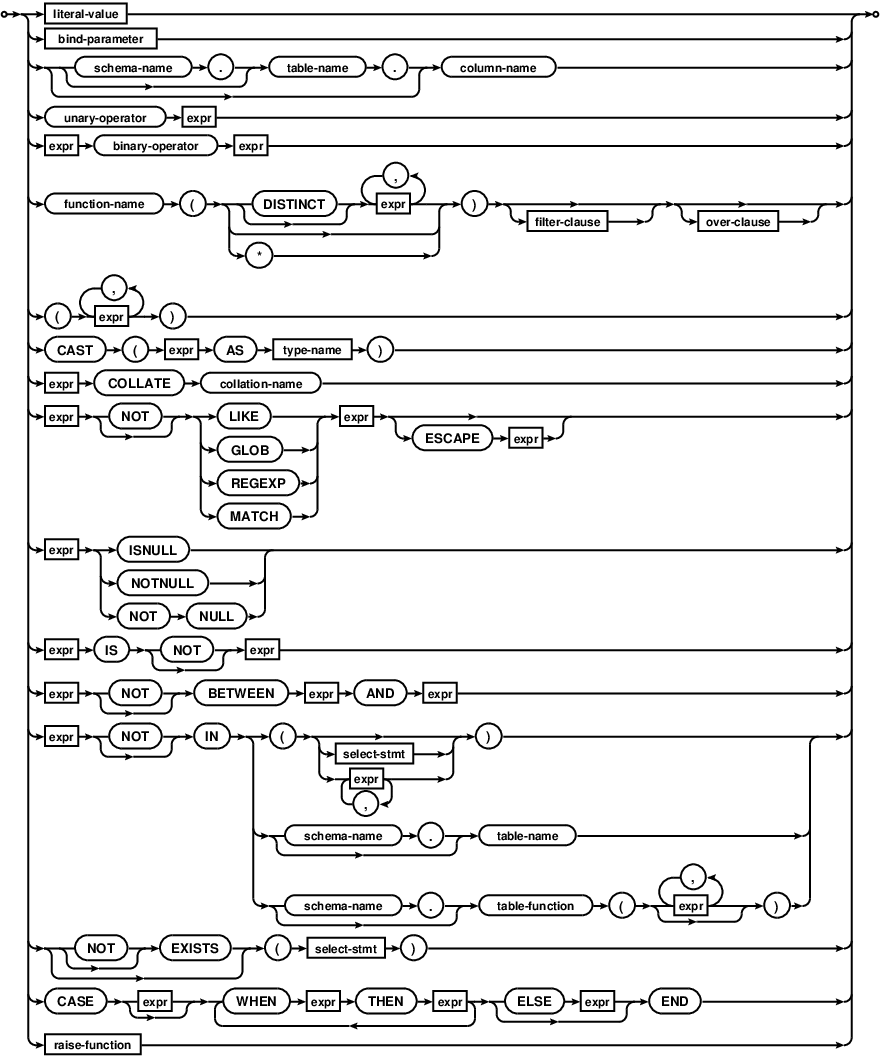
SELECT empno, ename, job FROM employee INTERSECT SELECT empno, ename, job FROM emp;

--EXCEPT

SELECT prod\_name, qty FROM products WHERE qty BETWEEN 1 AND 100 EXCEPT SELECT prod\_name, qty FROM products WHERE qty BETWEEN 50 AND 75;

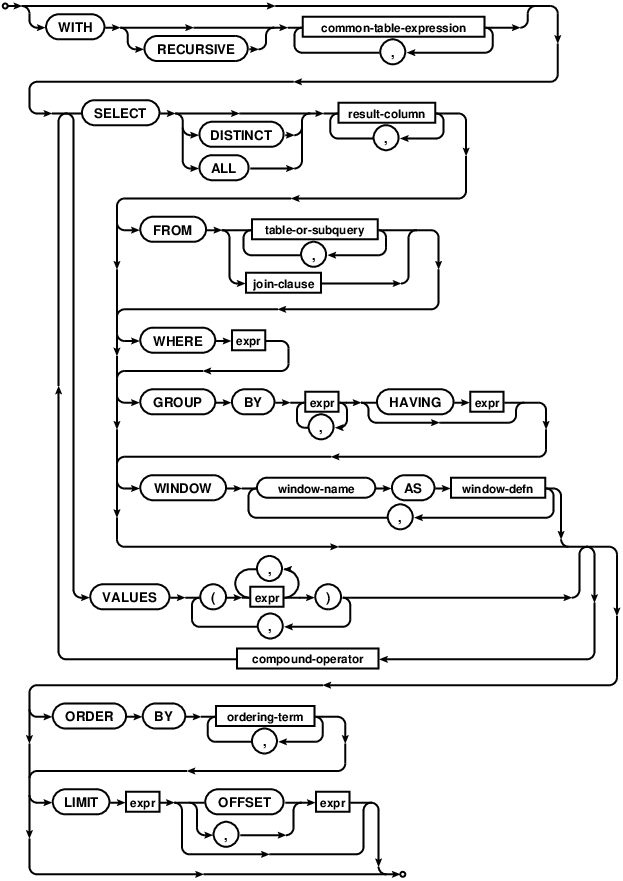
SQL Expression:

<https://www.sqlite.org/draft/lang_expr.html>

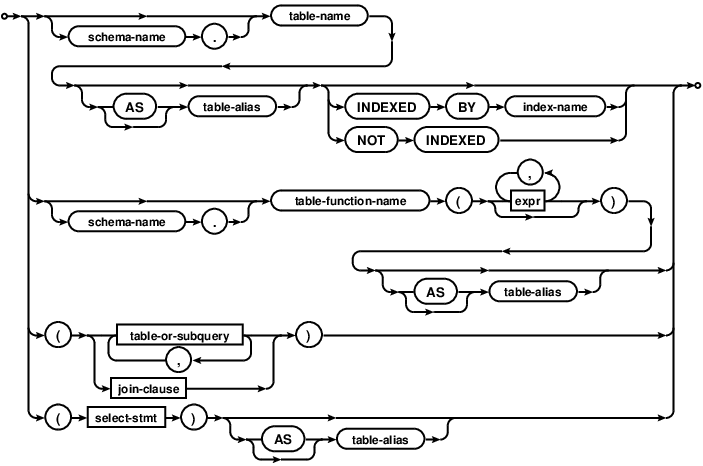


|  |  |
| --- | --- |
| [**filter-clause:**](https://www.sqlite.org/draft/syntax/filter-clause.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\filter-clause.gif** |
| [**literal-value:**](https://www.sqlite.org/draft/syntax/literal-value.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\literal-value.gif** |
| [**over-clause:**](https://www.sqlite.org/draft/syntax/over-clause.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\over-clause.gif** |
| [**raise-function:**](https://www.sqlite.org/draft/syntax/raise-function.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\raise-function.gif** |
| [**select-stmt:**](https://www.sqlite.org/draft/syntax/select-stmt.html) |  |
| [**common-table-expression:**](https://www.sqlite.org/draft/syntax/common-table-expression.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\common-table-expression.gif** |
| [**compound-operator:**](https://www.sqlite.org/draft/syntax/compound-operator.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\compound-operator.gif** |
| [**join-clause:**](https://www.sqlite.org/draft/syntax/join-clause.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\join-clause.gif** |
| [**ordering-term:**](https://www.sqlite.org/draft/syntax/ordering-term.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\ordering-term.gif** |
| [**result-column:**](https://www.sqlite.org/draft/syntax/result-column.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\result-column.gif** |
| [**table-or-subquery:**](https://www.sqlite.org/draft/syntax/table-or-subquery.html) |  |
| [**window-defn:**](https://www.sqlite.org/draft/syntax/window-defn.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\window-defn.gif** |
| [**type-name:**](https://www.sqlite.org/draft/syntax/type-name.html) | **C:\Users\SD027\AppData\Local\Microsoft\Windows\INetCache\Content.Word\type-name.gif** |

[**select-stmt:**](https://www.sqlite.org/draft/syntax/select-stmt.html)



[**table-or-subquery:**](https://www.sqlite.org/draft/syntax/table-or-subquery.html)



# [jQuery](http://jquery.com/) Quick API Reference

* 1. **SELECTORS**

**Basics**

[\*](https://api.jquery.com/all-selector/)

[.class](https://api.jquery.com/class-selector/)

[element](https://api.jquery.com/element-selector/)

[#id](https://api.jquery.com/id-selector/)

[selector1, selectorN, ...](https://api.jquery.com/multiple-selector/)

**Hierarchy**

[parent &gt; child](https://api.jquery.com/child-selector/)

[ancestor descendant](https://api.jquery.com/descendant-selector/)

[prev + next](https://api.jquery.com/next-adjacent-Selector/)

[prev ~ siblings](https://api.jquery.com/next-siblings-selector/)

**Basic Filters**

[:animated](https://api.jquery.com/animated-selector/)

[:eq()](https://api.jquery.com/eq-selector/)

[:even](https://api.jquery.com/even-selector/)

[:first](https://api.jquery.com/first-selector/)

[:gt()](https://api.jquery.com/gt-selector/)

[:header](https://api.jquery.com/header-selector/)

[:lang()](https://api.jquery.com/lang-selector/)

[:last](https://api.jquery.com/last-selector/)

[:lt()](https://api.jquery.com/lt-selector/)

[:not()](https://api.jquery.com/not-selector/)

[:odd](https://api.jquery.com/odd-selector/)

[:root](https://api.jquery.com/root-selector/)

[:target](https://api.jquery.com/target-selector/)

**Content Filters**

[:contains()](https://api.jquery.com/contains-selector/)

[:empty](https://api.jquery.com/empty-selector/)

[:has()](https://api.jquery.com/has-selector/)

[:parent](https://api.jquery.com/parent-selector/)

**Visibility Filters**

[:hidden](https://api.jquery.com/hidden-selector/)

[:visible](https://api.jquery.com/visible-selector/)

**Attribute**

[[name|="value"]](https://api.jquery.com/attribute-contains-prefix-selector/)

[[name\*="value"]](https://api.jquery.com/attribute-contains-selector/)

[[name~="value"]](https://api.jquery.com/attribute-contains-word-selector/)

[[name$="value"]](https://api.jquery.com/attribute-ends-with-selector/)

[[name="value"]](https://api.jquery.com/attribute-equals-selector/)

[[name!="value"]](https://api.jquery.com/attribute-not-equal-selector/)

[[name^="value"]](https://api.jquery.com/attribute-starts-with-selector/)

[[name]](https://api.jquery.com/has-attribute-selector/)

[[name="value"][name2="value2"]](https://api.jquery.com/multiple-attribute-selector/)

**Child Filters**

[:first-child](https://api.jquery.com/first-child-selector/)

[:first-of-type](https://api.jquery.com/first-of-type-selector/)

[:last-child](https://api.jquery.com/last-child-selector/)

[:last-of-type](https://api.jquery.com/last-of-type-selector/)

[:nth-child()](https://api.jquery.com/nth-child-selector/)

[:nth-last-child()](https://api.jquery.com/nth-last-child-selector/)

[:nth-last-of-type()](https://api.jquery.com/nth-last-of-type-selector/)

[:nth-of-type()](https://api.jquery.com/nth-of-type-selector/)

[:only-child](https://api.jquery.com/only-child-selector/)

[:only-of-type()](https://api.jquery.com/only-of-type-selector/)

**Forms**

[:button](https://api.jquery.com/button-selector/)

[:checkbox](https://api.jquery.com/checkbox-selector/)

[:checked](https://api.jquery.com/checked-selector/)

[:disabled](https://api.jquery.com/disabled-selector/)

[:enabled](https://api.jquery.com/enabled-selector/)

[:focus](https://api.jquery.com/focus-selector/)

[:file](https://api.jquery.com/file-selector/)

[:image](https://api.jquery.com/image-selector/)

[:input](https://api.jquery.com/input-selector/)

[:password](https://api.jquery.com/password-selector/)

[:radio](https://api.jquery.com/radio-selector/)

[:reset](https://api.jquery.com/reset-selector/)

[:selected](https://api.jquery.com/selected-selector/)

[:submit](https://api.jquery.com/submit-selector/)

[:text](https://api.jquery.com/text-selector/)

* 1. **ATTRIBUTES / CSS**

**Attributes**

[.attr()](https://api.jquery.com/attr/)

[.prop()](https://api.jquery.com/prop/)

[.removeAttr()](https://api.jquery.com/removeAttr/)

[.removeProp()](https://api.jquery.com/removeProp/)

[.val()](https://api.jquery.com/val/)

**CSS**

[.addClass()](https://api.jquery.com/addClass/)

[.css()](https://api.jquery.com/css/)

[jQuery.cssHooks](https://api.jquery.com/jQuery.cssHooks/)

[jQuery.cssNumber](https://api.jquery.com/jQuery.cssNumber/)

[jQuery.escapeSelector()](https://api.jquery.com/jQuery.escapeSelector/)

[.hasClass()](https://api.jquery.com/hasClass/)

[.removeClass()](https://api.jquery.com/removeClass/)

[.toggleClass()](https://api.jquery.com/toggleClass/)

**Dimensions**

[.height()](https://api.jquery.com/height/)

[.innerHeight()](https://api.jquery.com/innerHeight/)

[.innerWidth()](https://api.jquery.com/innerWidth/)

[.outerHeight()](https://api.jquery.com/outerHeight/)

[.outerWidth()](https://api.jquery.com/outerWidth/)

[.width()](https://api.jquery.com/width/)

**Offset**

[.offset()](https://api.jquery.com/offset/)

[.offsetParent()](https://api.jquery.com/offsetParent/)

[.position()](https://api.jquery.com/position/)

[.scrollLeft()](https://api.jquery.com/scrollLeft/)

[.scrollTop()](https://api.jquery.com/scrollTop/)

**Data**

[jQuery.data()](https://api.jquery.com/jQuery.data/)

[.data()](https://api.jquery.com/data/)

[jQuery.hasData()](https://api.jquery.com/jQuery.hasData/)

[jQuery.removeData()](https://api.jquery.com/jQuery.removeData/)

[.removeData()](https://api.jquery.com/removeData/)

* 1. **MANIPULATION**

**Copying**

[.clone()](https://api.jquery.com/clone/)

**DOM Insertion, Around**

[.wrap()](https://api.jquery.com/wrap/)

[.wrapAll()](https://api.jquery.com/wrapAll/)

[.wrapInner()](https://api.jquery.com/wrapInner/)

**DOM Insertion, Inside**

[.append()](https://api.jquery.com/append/)

[.appendTo()](https://api.jquery.com/appendTo/)

[.html()](https://api.jquery.com/html/)

[.prepend()](https://api.jquery.com/prepend/)

[.prependTo()](https://api.jquery.com/prependTo/)

[.text()](https://api.jquery.com/text/)

**DOM Insertion, Outside**

[.after()](https://api.jquery.com/after/)

[.before()](https://api.jquery.com/before/)

[.insertAfter()](https://api.jquery.com/insertAfter/)

[.insertBefore()](https://api.jquery.com/insertBefore/)

**DOM Removal**

[.detach()](https://api.jquery.com/detach/)

[.empty()](https://api.jquery.com/empty/)

[.remove()](https://api.jquery.com/remove/)

[.unwrap()](https://api.jquery.com/unwrap/)

**DOM Replacement**

[.replaceAll()](https://api.jquery.com/replaceAll/)

[.replaceWith()](https://api.jquery.com/replaceWith/)

* 1. **TRAVERSING**

**Filtering**

[.eq()](https://api.jquery.com/eq/)

[.filter()](https://api.jquery.com/filter/)

[.first()](https://api.jquery.com/first/)

[.has()](https://api.jquery.com/has/)

[.is()](https://api.jquery.com/is/)

[.last()](https://api.jquery.com/last/)

[.map()](https://api.jquery.com/map/)

[.not()](https://api.jquery.com/not/)

[.slice()](https://api.jquery.com/slice/)

**Miscellaneous Traversing**

[.add()](https://api.jquery.com/add/)

[.addBack()](https://api.jquery.com/addBack/)

[.andSelf()](https://api.jquery.com/andSelf/)

[.contents()](https://api.jquery.com/contents/)

[.each()](https://api.jquery.com/each/)

[.end()](https://api.jquery.com/end/)

**Tree Traversal**

[.children()](https://api.jquery.com/children/)

[.closest()](https://api.jquery.com/closest/)

[.find()](https://api.jquery.com/find/)

[.next()](https://api.jquery.com/next/)

[.nextAll()](https://api.jquery.com/nextAll/)

[.nextUntil()](https://api.jquery.com/nextUntil/)

[.parent()](https://api.jquery.com/parent/)

[.parents()](https://api.jquery.com/parents/)

[.parentsUntil()](https://api.jquery.com/parentsUntil/)

[.prev()](https://api.jquery.com/prev/)

[.prevAll()](https://api.jquery.com/prevAll/)

[.prevUntil()](https://api.jquery.com/prevUntil/)

[.siblings()](https://api.jquery.com/siblings/)

* 1. **EVENTS**

**Browser Events**

[~~.error()~~](https://api.jquery.com/error/)

[.resize()](https://api.jquery.com/resize/)

[.scroll()](https://api.jquery.com/scroll/)

**Document Loading**

[~~.load()~~](https://api.jquery.com/load-event/)

[.ready()](https://api.jquery.com/ready/)

[~~.unload()~~](https://api.jquery.com/unload/)

**Event Handler Attachment**

[.bind()](https://api.jquery.com/bind/)

[.delegate()](https://api.jquery.com/delegate/)

[~~.die()~~](https://api.jquery.com/die/)

[~~.live()~~](https://api.jquery.com/live/)

[.off()](https://api.jquery.com/off/)

[.on()](https://api.jquery.com/on/)

[.one()](https://api.jquery.com/one/)

[.trigger()](https://api.jquery.com/trigger/)

[.triggerHandler()](https://api.jquery.com/triggerHandler/)

[.unbind()](https://api.jquery.com/unbind/)

[.undelegate()](https://api.jquery.com/undelegate/)

**Form Events**

[.blur()](https://api.jquery.com/blur/)

[.change()](https://api.jquery.com/change/)

[.focus()](https://api.jquery.com/focus/)

[.focusin()](https://api.jquery.com/focusin/)

[.focusout()](https://api.jquery.com/focusout/)

[.select()](https://api.jquery.com/select/)

[.submit()](https://api.jquery.com/submit/)

**Keyboard Events**

[.keydown()](https://api.jquery.com/keydown/)

[.keypress()](https://api.jquery.com/keypress/)

[.keyup()](https://api.jquery.com/keyup/)

**Mouse Events**

[.click()](https://api.jquery.com/click/)

[.contextMenu()](https://api.jquery.com/contextmenu/)

[.dblclick()](https://api.jquery.com/dblclick/)

[.hover()](https://api.jquery.com/hover/)

[.mousedown()](https://api.jquery.com/mousedown/)

[.mouseenter()](https://api.jquery.com/mouseenter/)

[.mouseleave()](https://api.jquery.com/mouseleave/)

[.mousemove()](https://api.jquery.com/mousemove/)

[.mouseout()](https://api.jquery.com/mouseout/)

[.mouseover()](https://api.jquery.com/mouseover/)

[.mouseup()](https://api.jquery.com/mouseup/)

[~~.toggle()~~](https://api.jquery.com/toggle-event/)

**Event Object**

[event.currentTarget](https://api.jquery.com/event.currentTarget/)

[event.delegateTarget](https://api.jquery.com/event.delegateTarget/)

[event.data](https://api.jquery.com/event.data/)

[event.isDefaultPrevented()](https://api.jquery.com/event.isDefaultPrevented/)

[event.isImmediatePropagationStopped()](https://api.jquery.com/event.isImmediatePropagationStopped/)

[event.isPropagationStopped()](https://api.jquery.com/event.isPropagationStopped/)

[event.metaKey](https://api.jquery.com/event.metaKey/)

[event.namespace](https://api.jquery.com/event.namespace/)

[event.pageX](https://api.jquery.com/event.pageX/)

[event.pageY](https://api.jquery.com/event.pageY/)

[event.preventDefault()](https://api.jquery.com/event.preventDefault/)

[event.relatedTarget](https://api.jquery.com/event.relatedTarget/)

[event.result](https://api.jquery.com/event.result/)

[event.stopImmediatePropagation()](https://api.jquery.com/event.stopImmediatePropagation/)

[event.stopPropagation()](https://api.jquery.com/event.stopPropagation/)

[event.target](https://api.jquery.com/event.target/)

[event.timeStamp](https://api.jquery.com/event.timeStamp/)

[event.type](https://api.jquery.com/event.type/)

[event.which](https://api.jquery.com/event.which/)

* 1. **EFFECTS**

**Basics**

[.hide()](https://api.jquery.com/hide/)

[.show()](https://api.jquery.com/show/)

[.toggle()](https://api.jquery.com/toggle/)

**Custom**

[.animate()](https://api.jquery.com/animate/)

[.clearQueue()](https://api.jquery.com/clearQueue/)

[.delay()](https://api.jquery.com/delay/)

[.dequeue()](https://api.jquery.com/dequeue/)

[jQuery.dequeue()](https://api.jquery.com/jQuery.dequeue/)

[.finish()](https://api.jquery.com/finish/)

[jQuery.fx.interval](https://api.jquery.com/jQuery.fx.interval/)

[jQuery.fx.off](https://api.jquery.com/jQuery.fx.off/)

[jQuery.speed](https://api.jquery.com/jQuery.speed/)

[.queue()](https://api.jquery.com/queue/)

[jQuery.queue()](https://api.jquery.com/jQuery.queue/)

[.stop()](https://api.jquery.com/stop/)

**Fading**

[.fadeIn()](https://api.jquery.com/fadeIn/)

[.fadeOut()](https://api.jquery.com/fadeOut/)

[.fadeTo()](https://api.jquery.com/fadeTo/)

[.fadeToggle()](https://api.jquery.com/fadeToggle/)

**Sliding**

[.slideDown()](https://api.jquery.com/slideDown/)

[.slideToggle()](https://api.jquery.com/slideToggle/)

[.slideUp()](https://api.jquery.com/slideUp/)

* 1. **AJAX**

**Global Ajax Event Handlers**

[.ajaxComplete()](https://api.jquery.com/ajaxComplete/)

[.ajaxError()](https://api.jquery.com/ajaxError/)

[.ajaxSend()](https://api.jquery.com/ajaxSend/)

[.ajaxStart()](https://api.jquery.com/ajaxStart/)

[.ajaxStop()](https://api.jquery.com/ajaxStop/)

[.ajaxSuccess()](https://api.jquery.com/ajaxSuccess/)

**Helper Functions**

[jQuery.param()](https://api.jquery.com/jQuery.param/)

[.serialize()](https://api.jquery.com/serialize/)

[.serializeArray()](https://api.jquery.com/serializeArray/)

**Low-Level Interface**

[jQuery.ajax()](https://api.jquery.com/jQuery.ajax/)

[jQuery.prefilter()](https://api.jquery.com/jQuery.ajaxPrefilter/)

[jQuery.ajaxSetup()](https://api.jquery.com/jQuery.ajaxSetup/)

[jQuery.ajaxTransport()](https://api.jquery.com/jQuery.ajaxTransport/)

**Shorthand Methods**

[jQuery.get()](https://api.jquery.com/jQuery.get/)

[jQuery.getJSON()](https://api.jquery.com/jQuery.getJSON/)

[jQuery.getScript()](https://api.jquery.com/jQuery.getScript/)

[jQuery.post()](https://api.jquery.com/jQuery.post/)

[.load()](https://api.jquery.com/load/)

* 1. **CORE**

**jQuery Object**

[jQuery()](https://api.jquery.com/jQuery/)

[jQuery.noConflict()](https://api.jquery.com/jQuery.noConflict/)

[~~jQuery.sub()~~](https://api.jquery.com/jQuery.sub/)

[jQuery.holdReady()](https://api.jquery.com/jQuery.holdReady/)

[jQuery.when()](https://api.jquery.com/jQuery.when/)

**Deferred Object**

[jQuery.Deferred()](https://api.jquery.com/jQuery.Deferred/)

[deferred.always()](https://api.jquery.com/deferred.always/)

[deferred.done()](https://api.jquery.com/deferred.done/)

[deferred.fail()](https://api.jquery.com/deferred.fail/)

[~~deferred.isRejected()~~](https://api.jquery.com/deferred.isRejected/)

[~~deferred.isResolved()~~](https://api.jquery.com/deferred.isResolved/)

[deferred.notify()](https://api.jquery.com/deferred.notify/)

[deferred.notifyWith()](https://api.jquery.com/deferred.notifyWith/)

[deferred.pipe()](https://api.jquery.com/deferred.pipe/)

[deferred.progress()](https://api.jquery.com/deferred.progress/)

[deferred.promise()](https://api.jquery.com/deferred.promise/)

[deferred.reject()](https://api.jquery.com/deferred.reject/)

[deferred.rejectWith()](https://api.jquery.com/deferred.rejectWith/)

[deferred.resolve()](https://api.jquery.com/deferred.resolve/)

[deferred.resolveWith()](https://api.jquery.com/deferred.resolveWith/)

[deferred.state()](https://api.jquery.com/deferred.state/)

[deferred.then()](https://api.jquery.com/deferred.then/)

[.promise()](https://api.jquery.com/promise/)

**Utilities**

[~~jQuery.boxModel~~](https://api.jquery.com/jQuery.boxModel/)

[~~jQuery.browser~~](https://api.jquery.com/jQuery.browser/)

[jQuery.contains()](https://api.jquery.com/jQuery.contains/)

[jQuery.each()](https://api.jquery.com/jQuery.each/)

[jQuery.extend()](https://api.jquery.com/jQuery.extend/)

[jQuery.globalEval()](https://api.jquery.com/jQuery.globalEval/)

[jQuery.grep()](https://api.jquery.com/jQuery.grep/)

[jQuery.inArray()](https://api.jquery.com/jQuery.inArray/)

[jQuery.isArray()](https://api.jquery.com/jQuery.isArray/)

[jQuery.isEmptyObject()](https://api.jquery.com/jQuery.isEmptyObject/)

[jQuery.isFunction()](https://api.jquery.com/jQuery.isFunction/)

[jQuery.isNumeric()](https://api.jquery.com/jQuery.isNumeric/)

[jQuery.isPlainObject()](https://api.jquery.com/jQuery.isPlainObject/)

[jQuery.isWindow()](https://api.jquery.com/jQuery.isWindow/)

[jQuery.isXMLDoc()](https://api.jquery.com/jQuery.isXMLDoc/)

[jQuery.makeArray()](https://api.jquery.com/jQuery.makeArray/)

[jQuery.map()](https://api.jquery.com/jQuery.map/)

[jQuery.merge()](https://api.jquery.com/jQuery.merge/)

[jQuery.noop()](https://api.jquery.com/jQuery.noop/)

[jQuery.now()](https://api.jquery.com/jQuery.now/)

[jQuery.parseHTML()](https://api.jquery.com/jQuery.parseHTML/)

[jQuery.parseJSON()](https://api.jquery.com/jQuery.parseJSON/)

[jQuery.parseXML()](https://api.jquery.com/jQuery.parseXML/)

[jQuery.proxy()](https://api.jquery.com/jQuery.proxy/)

[jQuery.support](https://api.jquery.com/jQuery.support/)

[jQuery.trim()](https://api.jquery.com/jQuery.trim/)

[jQuery.type()](https://api.jquery.com/jQuery.type/)

[jQuery.unique()](https://api.jquery.com/jQuery.unique/)

[jQuery.uniqueSort()](https://api.jquery.com/jQuery.uniqueSort/)

**DOM Element Methods**

[.get()](https://api.jquery.com/get/)

[.index()](https://api.jquery.com/index/)

[.size()](https://api.jquery.com/size/)

[.toArray()](https://api.jquery.com/toArray/)

**Internals**

[.jquery](https://api.jquery.com/jquery-2/)

[~~.context~~](https://api.jquery.com/context/)

[jQuery.error()](https://api.jquery.com/jQuery.error/)

[.length](https://api.jquery.com/length/)

[.pushStack()](https://api.jquery.com/pushStack/)

[~~.selector~~](https://api.jquery.com/selector/)

**Callbacks Object**

[jQuery.Callbacks()](https://api.jquery.com/jQuery.Callbacks/)

[callbacks.add()](https://api.jquery.com/callbacks.add/)

[callbacks.disable()](https://api.jquery.com/callbacks.disable/)

[callbacks.disabled()](https://api.jquery.com/callbacks.disabled/)

[callbacks.empty()](https://api.jquery.com/callbacks.empty/)

[callbacks.fire()](https://api.jquery.com/callbacks.fire/)

[callbacks.fired()](https://api.jquery.com/callbacks.fired/)

[callbacks.fireWith()](https://api.jquery.com/callbacks.fireWith/)

[callbacks.has()](https://api.jquery.com/callbacks.has/)

[callbacks.lock()](https://api.jquery.com/callbacks.lock/)

[callbacks.locked()](https://api.jquery.com/callbacks.locked/)

[callbacks.remove()](https://api.jquery.com/callbacks.remove/)

**Emmet | Cheat Sheet** (2017-03)

[Syntax](#_bookmark0) 1

[Child: > 1](#_bookmark0)

[Sibling: + 1](#_bookmark0)

[Climb-up: ^ 1](#_bookmark0)

[Grouping: () 1](#_bookmark0)

[Multiplication: \* 2](#_bookmark1)

[Naming and numbering: $ 2](#_bookmark1) [ID and CLASS attributes 2](#_bookmark1)

[Custom attributes 2](#_bookmark1)

[Text: {} 2](#_bookmark1)

[Implicit tag names 2](#_bookmark1)

[HTML](#_bookmark2) 3

[Form, Input, Button 4](#_bookmark3)

[CSS](#_bookmark4) 7

[Visual Formatting 7](#_bookmark4)

[Margin & Padding 9](#_bookmark5)

[Box Sizing 9](#_bookmark5)

[Font 9](#_bookmark5)

[Text 11](#_bookmark6)

[Background 13](#_bookmark7)

[Color 14](#_bookmark8)

[Generated content 14](#_bookmark8)

[Outline 15](#_bookmark9)

[Tables 15](#_bookmark9)

[Border 15](#_bookmark9)

[Lists 18](#_bookmark10)

[Print 18](#_bookmark10)

[Others 18](#_bookmark10)

[Animation 19](#_bookmark11)

[Flex 20](#_bookmark12)

[Transform, Transition 21](#_bookmark13)

[XSL 2](#_bookmark14)2

[HTML DOCTYPES 2](#_bookmark15)3

[http://docs .emmet io/](http://docs.emmet.io/)

<https://docs.emmet.io/cheat-sheet/>

**SYNTAX**

**Child: >**

**nav>ul>li**

<nav>

<ul><li></li></ul>

</nav>

## Sibling: +

**div+p+bq**

<div></div>

<p></p>

<blockquote></blockquote>

## Climb-up: ^

**div+div>p>span+em^bq**

<div></div>

<div>

<p><span></span><em></em></p>

<blockquote></blockquote>

</div>

div+div>p>span+em^^bq

<div></div>

<div>

<p><span></span><em></em></p>

</div>

<blockquote></blockquote>

## Grouping: ()

**div>(header>ul>li\*2>a)+footer>p**

<div>

<header>

<ul>

<li><a href=""></a></li>

<li><a href=""></a></li>

</ul>

</header>

<footer> <p></p> </footer>

</div>

**(div>dl>(dt+dd)\*3)+footer>p**

<div> <dl>

<dt></dt>

<dd></dd>

<dt></dt>

<dd></dd>

</dl> </div>

<footer> <p></p>

</footer>

## Multiplication: \*

**ul>li\*5**

<ul>

<li></li>

………

<li></li>

</ul>

**Naming and numbering: $**

**ul>li.sample$\*5**

<ul>

<li class="sample1"></li>

<li class="sample2"></li>

<li class="sample3"></li>

<li class="sample4"></li>

<li class="sample5"></li>

</ul>

**h$[title=topic$]{Headline $}\*3**

<h1 title="topic1">Headline 1</h1>

<h2 title="topic2">Headline 2</h2>

<h3 title="topic3">Headline 3</h3>

**ul>li.item$$$\*5**

<ul>

<li class="item001"></li>

….

<li class="item004"></li>

<li class="item005"></li>

</ul>

**ul>li.item$@-\*5**

<ul>

<li class="item5"></li>

<li class="item4"></li>

…….

<li class="item1"></li>

</ul>

**ul>li.item$@3\*5**

<ul>

<li class="item3"></li>

…….

<li class="item6"></li>

<li class="item7"></li>

</ul>

**ID and CLASS attributes**

**#header**

<div id="header"></div>

**.title**

<div class="title"></div>

**form#search.wide**

<form id="search" class="wide"></ form>

**p.class1.class2.class3**

<p class="class1 class2 class3"></p>

## Custom attributes

**p[title="Hello world"]**

<p title="Hello world"></p>

**td[rowspan=2 colspan=3 title]**

<td rowspan="2" colspan="3" title=""></td>

**[a=‘value1‘ b="value2"]**

<div a="value1" b="value2"></div>

## Text: {}

**a{Click me}**

<a href="">Click me</a>

**p>{Click }+a{here}+{ to continue}**

<p>Click <a href="">here</a> to continue</p>

## Implicit tag names

**.class**

<div class="class"></div>

**em>.class**

<em><span class="class"></span></ em>

**ul>.class**

<ul>

<li class="class"></li>

</ul>

**table>.row>.col**

<table>

<tr class="row">

<td class="col"></td>

</tr>

</table>

# HTML

## Form, Input, Button

**form** <form action=""></form>

**form:get** <form action="" method="get"></ form>

**form:post** <form action="" method="post"></ form>

**label** <label for=""></label>

**input** <input type="text" />

**inp** <input type="text" name="" id="" />

**input:hidden, input:h** <input type="hidden" name="" />

**input:text, input:t** <input type="text" name="" id="" />

**input:search** <input type="search" name="" id="" />

**input:email** <input type="email" name="" id="" />

**input:url** <input type="url" name="" id="" />

**input:password, input:p** <input type="password" name="" id="" />

**input:datetime** <input type="datetime" name="" id="" />

**input:date** <input type="date" name="" id="" />

**input:datetime-local** <input type="datetime-local" name="" id="" />

**input:month** <input type="month" name="" id="" />

**input:week** <input type="week" name="" id="" />

**input:time** <input type="time" name="" id="" />

**input:tel** <input type="tel" name="" id="" />

**input:number** <input type="number" name="" id="" />

**input:color** <input type="color" name="" id=""/>

**input:checkbox, input:c** <input type="checkbox" name="" id="" />

**input:radio, input:r** <input type="radio" name="" id=""/>

**input:range** <input type="range" name="" id=""/>

**input:file, input:f** <input type="file" name="" id="" />

**input:submit, input:s** <input type="submit" value="" />

**input:image, input:I** <input type="image" src="" alt=""/>

**input:button, input:b** <input type="button" value="" />

**isindex** <isindex />

**input:reset** <input type="reset" value="" />

**select** <select name="" id=""></select>

**select:disabled, select:d**

<select name="" id="" disabled="disabled"></select>

**option, opt** <option value=""></option>

**textarea**

<textarea name="" id="" cols="30" rows="10"></textarea>

**Marquee** <marquee behavior="" direction=""></marquee>

**menu:context, menu:c** <menu type="context"></menu>

**menu:toolbar, menu:t** <menu type="toolbar"></menu>

**video** <video src=""></video>

**audio** <audio src=""></audio>

**html:xml** <html [xmlns="http://www.](http://www/) w3.org/1999/xhtml"></html>

**keygen** <keygen />

**command** <command />

**button:submit, button:s, btn:s** <button type="submit"></button>

**button:reset, button:r, btn:r** <button type="reset"></button>

**button:disabled, button:d, btn:d**

<button disabled="disabled"></ button>

**fieldset:disabled, fieldset:d, fset:d, fst:d**

<fieldset disabled="disabled"></ fieldset>