

ΜΟΝΑΔΑ ΑΡΙΣΤΕΙΑΣ
ΑΝΟΙΧΤΟΥ ΛΟΓΙΣΜΙΚΟΥ

ΧΑΡΟΚΟΠΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ



ΜΟΝΑΔΕΣ ΑΡΙΣΤΕΙΑΣ
ΑΝΟΙΧΤΟΥ ΛΟΓΙΣΜΙΚΟΥ



WordPress

Wordpress Plugins

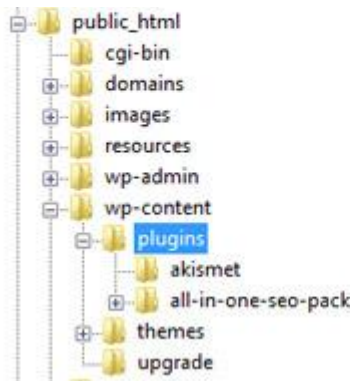


WP Folder Structure

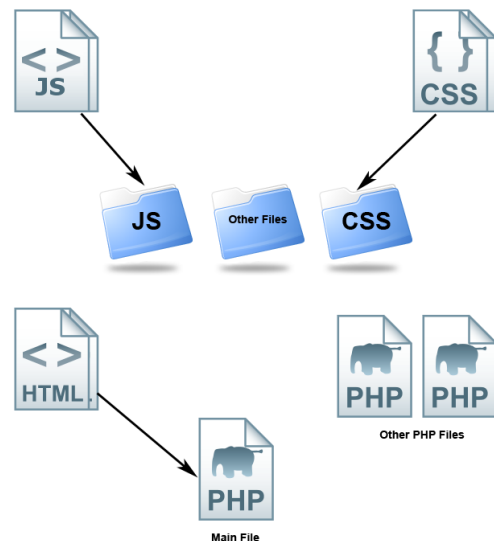
Η δομή των καταλόγων του wordpress

Τα plugins τοποθετούνται στον καταλογο
wp-content/plugins

- απλό plugin: ένα ρηρ αρχείο
- πιο σύνθετο: κατάλογος (προτείνεται)
- καλό θα ήταν να υπάρχει και ένα readme.txt αρχείο
- κατάλογος και βασικό αρχείο καλό είναι να έχουν το ίδιο όνομα



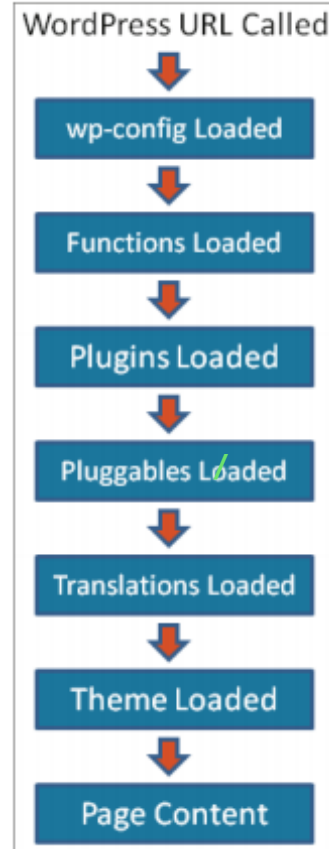
WordPress Plugin Structure



Διαδικασία Φόρτωσης

Τα plugins «φορτώνονται» πολύ νωρίς
τυπική δομή:

```
/unique-plugin-name – (no spaces or special characters)  
unique-plugin-name.php – Primary plugin PHP file  
  uninstall.php – The uninstall file for your plugin  
  /js – Folder for JavaScript files  
  /css – Folder for stylesheet files  
  /includes – Folder for other PHP includes  
  /images – Folder for plugin images
```



Γράφοντας ένα Plugin

- Δημιουργία ενός php αρχείου με όνομα το επιθυμητό όνομα του plugin
- Οι πρώτες γραμμές πρέπει να είναι συγκεκριμένα σχόλια που αφορούν πληροφορίες για το plugin.

```
<?php
/* Plugin Name: 1WD Slider
Plugin URI: http://1stwebdesigner.com/
Description: Slider Component for WordPress
Version: 1.0
Author: Rakhitha Nimesh
Author URI: http://1stwebdesigner.com/
License: GPLv2 or later
*/
?>
```

WordPress Naming Conventions And Best Practices

- Στις μεταβλητές και τις συναρτήσεις καλό είναι να έχετε το δικό σας πρόθεμα (το πρόθεμα του plugin σας)
- Χρησιμοποιήστε τα κανονικά tags της php `<?php ... ?>`
- Χρησιμοποιήστε `{}` σε όλα τα blocks
- Χρησιμοποιήστε `'` και `"` αποδοτικά και αποφύγετε τους escape characters στα `'` μέσα σε strings
- Αν κλείνετε τα php statements φροντίστε να μην αφήνετε κενά μετά. Αλλιώς μην κλείνετε τα php tags στο τέλος των αρχείων.
- Αφήνετε κενά μετά τα `,` και στους τελεστές

Filters And Actions

- Βασικά στοιχεία του wordpress API
- **Filters:** functions στις οποίες το wordpress περνά δεδομένα, σε συγκεκριμένα σημεία εκτέλεσης, πριν γίνει κάποια ενέργεια στα δεδομένα. “Κάθονται” ανάμεσα στη βάση και τον browser.
- **Actions:** Ενεργοποιούνται από συγκεκριμένα γεγονότα που συμβαίνουν στο wordpress. Απαντούν σε αυτά τα γεγονότα εκτελώντας μια συνάρτηση.



Filters

"overwrite code"

- Μπορούμε να αλλάξουμε μια μεταβλητή του wordpress ή το αποτέλεσμα μιας function
- Επιδρά σε μεταβλητές στο page content
 - **the_content** {content of your posts and pages}
 - **the_tags** {applied to the tags retrieved from the database, prior to printing on the screen}
 - **the_title** {applied to the post title retrieved from the database, prior to printing on the screen}
 - **wp_title** {applied to the blog page title before sending to the browser in the wp_title function}
 -

{τι να αλλάξω} {τι να εκτελεσω} {προτεραιότητα} {# παραμέτρων}

```
add_filter( $tag, $function_to_add, $priority, $accepted_args );
```

```
apply_filters($tag, $value)
```

Actions (hooks)

“add code”

- Μοιάζει με τα filters, αλλά εφαρμόζεται όταν γίνουν συγκεκριμένες ενέργειες:
 - **publish_post** – called when a post is published or when status is changed into “published”
 - **save_post** – called when a post/page is created from start or updated
 - **wp_head** – called when the template is loaded and runs the wp_head() function
 - **loop_end** – called immediately after the final post has been processed through the WordPress loop
 - **trackback_post** – called whenever a new trackback is added
 - ...

```
do_action( 'wp_head' );
```

```
add_action( 'action_hook', 'your_function_name' );
```



Παράδειγμα

```
<?php
function hk_trim_content( $limit ) {
    $content = explode( ' ', get_the_content(), $limit );

    if ( count( $content ) >= $limit ) {
        array_pop( $content );
        $content = implode(" ", $content). '...';
    } else {
        $content = implode(" ", $content);
    }

    $content = preg_replace('/\[.+\\]/', '', $content);
    $content = apply_filters('the_content', $content);

    return $content;
}
?>
```

```
has_action( $tag, $function_to_check );
```

```
<?php echo hk_trim_content(55);
// display page content limited at 55 chars
?>
```

Activation / Deactivation

Μπορούμε να κάνουμε ενέργειες κατά την ενεργοποίηση / απενεργοποίηση του plugin

- Δημιουργία πινάκων στη βάση δεδομένων και διαγραφή τους κατά την απενεργοποίηση
- Καθορισμός επιλογών κατά την ενεργοποίηση και διαγραφή τους κατά την απενεργοποίηση
- Έλεγχος για άλλα προαπαιτούμενα plugin

register_activation_hook()

This hook allows you to create a function that runs when your plugin is activated. It takes the path to your main plugin file as the first argument, and the function that you want to run as the second argument. You can use this to check the version of your plugin, do some upgrades between versions, check for the correct PHP version and so on.

register_deactivation_hook()

The name says it all. This function works like its counterpart above, but it runs whenever your plugin is deactivated. I suggest using the next function when deleting data; use this one just for general housekeeping.

register_uninstall_hook()

This function runs when the website administrator deletes your plugin in WordPress' back end. This is a great way to remove data that has been lying around, such as database tables, settings and what not. A drawback to this method is that the plugin needs to be able to run for it to work; so, if your plugin cannot uninstall in this way, you can create an uninstall.php file. Check out this function's documentation for more information.

Things to do...

- localization
- options
- styles
- debug http://codex.wordpress.org/Debugging_in_WordPress
- widgets

Menus

Menus and submenus

```
< ?php add_menu_page( page_title, menu_title, capability,  
menu_slug, function,  
icon_url, position ); ? >  
  
< ?php add_submenu_page( parent_slug, page_title,  
menu_title, capability,  
menu_slug, function ); ? >
```

The `add_menu_page()` function accepts the following parameters:

- `page_title` — The title of the page as shown in the `<title>` tags
- `menu_title` — The name of your menu displayed on the dashboard
- `capability` — Minimum capability required to view the menu
- `menu_slug` — Slug name to refer to the menu; should be a unique name
- `function`: Function to be called to display the page content for the item
- `icon_url` — URL to a custom image to use as the Menu icon
- `position` — Location in the menu order where it should appear

Widgets

Widgets are a great way to give the users of your plugin an easy method to display your plugin information or data.

WordPress features a Widgets API for creating and interacting with widgets.

In this section you explore how to create widgets, add and save widget options, and display plugin information in the widget.



Code

<https://drive.google.com/file/d/0B8IPE7CZ3z8ENEgyWUxveEFiQkE/view?usp=sharing>

HTTP API

wp_remote_Funcions

- wp_remote_get()
- wp_remote_post()
- wp_remote_head()

PARAMETER	SIGNIFICATION
'method'	Either 'GET', 'POST', or 'HEAD'. Some transports (the HTTP or the CURL extension for instance) may accept other rarely used methods such as 'PUT' or 'TRACE', but should not be assumed.
'timeout'	A number of seconds: how long the connection should stay open before failing when no response.
'user-agent'	The user-agent used to identify “who” is performing the request. Defaults to "WordPress/" followed by the version of WordPress running and the URL of the blog issuing the request.
'headers'	An array of additional headers.
'cookies'	An array of cookie values passed to the server.
'body'	The body of the request, either a string or an array, which is data submitted to the URL.

HTTP API response

wp_remote_ Companion Functions

- **wp_remote_retrieve_response_code()** — Returns just the response code (for example, 200) of an HTTP response
- **wp_remote_retrieve_response_message()** — Returns just the response message (for example, “OK”)
- **wp_remote_retrieve_body()** — Returns the body of the response
- **wp_remote_retrieve_headers()** — Returns all the headers of a server response
- **wp_remote_retrieve_header()** — Returns just one particular header from a server response

Παράδειγμα

```
<?php
$url = 'http://www.example.com/bleh';
// Send GET request
$response = wp_remote_get( $url );
// Check for server response
if (is_wp_error($response))
{
    $code = $response->get_error_message();
    wp_die('Requests could not execute. Error was: ' . $code);
}
// Check that the server sent a "404 Not Found" HTTP status code
if (wp_remote_retrieve_response_code($response) == 404)
{
    wp_die( 'Link not found' );
}
// So far, so good echo 'Link found';
?>
```

Shortcodes



[shortcode]

```
// Extended subscription function with subscription type
variable
function subscribe_multilink_shortcode( $atts ) {
    extract( shortcode_atts( array(
        'subtype' => 'RSS',
        'subtypeurl' => 'http://feeds.feedburner.
com/ElegantThemes',
    ), $atts, 'multilink' ) );

    return sprintf( 'Be sure to subscribe to future Elegant
Themes updates <a href=%1$s>by %2$s</a>.',
        esc_url( $subtypeurl ),
        esc_html( $subtype )
    );
}
add_shortcode( 'subscribe', 'subscribe_multilink_shortcode' );
```

```
[subscribe]
[subscribe subtype="Twitter" subtypeurl="http://www.
twitter.com/elegantthemes/"]
[subscribe subtype="Facebook" subtypeurl="http://www.
facebook.com/elegantthemes/"]
[subscribe subtype="Google" subtypeurl="http://plus.
google.com/+elegantthemes/"]
```