acf_add_options_page()

- Description
- Parameters
  - $settings
- Return
- Examples
  - Default options page
    - functions.php
  - Customized options page
    - functions.php
  - Creating sub pages
    - functions.php
- Notes
  - Priority
  - WordPress 6.8

Description

Adds an options page to the admin menu.

Options pages are used to store global settings. These settings are not tied to a specific post, but are instead stored in the `wp_options` table.

Once registered, your page will appear in the admin menu. You can then assign fields to your page via the “Options Page” location rule when editing a field group.

Parameters

```
acf_add_options_page( [$settings] );
```

$settings

(array) (Optional) Array of arguments for registering an options page. – **page_title**
(string) The title displayed on the options page. Defaults to ‘Options’.

```
‘page\_title’ => \_\_(‘My Options’),
```

- **menu_title**
  (string) The title displayed in the admin menu. Defaults to ‘page_title’ value.

  ```
  'menu_title' => __('My Options'),
  ```

- **menu_slug**
  (string) The URL slug used to uniquely identify this options page. Defaults to a url friendly version of menu_title.

  ```
  'menu_slug' => 'my-options',
  ```

- **capability**
  (string) The capability required for this menu to be displayed to the user. Defaults to ‘edit_posts’. Read more about Roles and Capabilities.

  ```
  'capability' => 'edit_posts',
  ```

- **position**
  (int|string) The position in the menu order where this menu should appear. WARNING: if two menu items use the same position, one of the items may be overwritten. Risk of conflict can be reduced by using decimal instead of integer values, e.g. ‘63.3’ instead of 63 (must use quotes). Defaults to bottom of utility menu items.

  ```
  'position' => '',
  ```

- **parent_slug**
  (string) The slug of another WP admin page. If set, this will become a child page of that parent.

  ```
  'parent_slug' => 'my-parent-page',
  ```

- **icon_url**
  (string) The icon class for this menu. Defaults to default WordPress gear. Read more about Dashicons.

```
'icon_url' => '',
```

- **redirect**
  (bool) If set to true, this options page will redirect to the first child page (if a child page exists). If set to false, this parent page will appear alongside any child pages as its own page. Defaults to true.

```
'redirect' => true,
```

- **post_id**
  (int|string) The `$post_id` to save and load values from. Can be set to a numeric post ID (123), or a string ('user_2'). Read more about the available post_id values. Defaults to 'options'. Added in v5.2.7.

```
'post_id' => 'options',
```

- **autoload**
  (bool) Data saved in the wp_options table is given an "autoload" identifier. When set to true, WP will automatically load these values within a single SQL query which can improve page load performance. Defaults to false. Added in v5.2.8.

```
'autoload' => false,
```

- **update_button**
  (string) The text displayed on the options page submit button. Added in v5.3.7.

```
'update_button' => __('Update', 'acf'),
```

- **updated_message**
  (string) The message shown above the form after updating the options page. Added in v5.6.0.

```
'updated_message' => __("Options Updated", 'acf'),
```

Return

(array) The validated and final page settings. This is useful to find the page's menu_slug and use it later when adding child options pages.

Examples

Default options page

This example shows how to create the default options page.

functions.php

```
add_action('acf/init', function() {
  if( function_exists('acf_add_options_page') ) {
    acf_add_options_page();
  }
} );
```

Customized options page

This example shows how to create a customized options page and store the data in a variable for later use.

functions.php

```
add_action('acf/init', 'my_acf_op_init');
function my_acf_op_init() {

    // Check function exists.
    if( function_exists('acf_add_options_page') ) {

        // Register options page.
        $option_page = acf_add_options_page(array(
            'page_title'    => __('Theme General Settings'),
            'menu_title'    => __('Theme Settings'),
            'menu_slug'     => 'theme-general-settings',
            'capability'    => 'edit_posts',
            'redirect'      => false
        ));
    }
}
```

Creating sub pages

This example shows how to create an options sub page using the *parent_slug* attribute.

functions.php

```
add_action('acf/init', 'my_acf_op_init');
function my_acf_op_init() {

    // Check function exists.
    if( function_exists('acf_add_options_page') ) {

        // Add parent.
        $parent = acf_add_options_page(array(
            'page_title'  => __('Theme General Settings'),
            'menu_title'  => __('Theme Settings'),
            'redirect'    => false,
        ));

        // Add sub page.
        $child = acf_add_options_page(array(
            'page_title'  => __('Social Settings'),
            'menu_title'  => __('Social'),
            'parent_slug' => $parent['menu_slug'],
        ));
    }
}
```

Notes

Priority

This function must be used before the action `admin_menu` (priority 99) as this is when admin pages are registered in WordPress core. We advise using the “acf/init” action.

WordPress 6.8

Since WordPress 6.8, any function that causes ACF to initialise early such as this one will trigger an PHP notice, if you call it before the `acf/init` action:

```
Notice: Function _load_textdomain_just_in_time was called incorrectly. Translation loading for the acf domain was triggered too early. This is usually an indicator for some code in the plugin or theme running too early. Translations should be loaded at the init action or later.
```

To avoid this, you must make sure you don’t call any ACF functions before the `acf/init` action. Our code samples above show how to wrap these calls appropriately.

□□□ #1
□ □□□ □□□ 2026-08-29 15:44:03 UTC
□ □□□ □□□ 2026-08-29 15:44:03 UTC