

Creating fields with code



- [Registering custom fields with PHP](#)
 - [Field group settings](#)
 - [Contexts](#)
 - [Fields](#)
 - [Field types](#)
 - [Field settings](#)
 - [Field-specific settings](#)
 - [Shorthand syntax](#)
 - [Video tutorial](#)
 - [FAQ](#)
-

```
import LiteYouTubeEmbed from 'react-lite-youtube-embed'; import 'react-lite-youtube-embed/dist/LiteYouTubeEmbed.css'; import Tabs from '@theme/Tabs'; import TabItem from '@theme/TabItem'; import FAQ from '@site/src/components/FAQ';
```

Creating custom fields with code is suitable if you want to keep everything in your themes or plugins. This way, you can re-use them in many websites and put them under a version control like Git.

Registering custom fields with PHP

To create custom fields, you will need to use the filter `rwmb_meta_boxes` to register field groups. This filter accepts one parameter - the array of field groups:

```
add_filter( 'rwmb_meta_boxes', function ( $meta_boxes ) {  
    $meta_boxes[] = [  
        'title'      => 'Event details',  
        'post_types' => 'event',  
        'fields'     => [  
            [  
                'name' => 'Date and time',
```

```

        'id'    => 'datetime',
        'type' => 'datetime',
    ],
    [
        'name' => 'Location',
        'id'    => 'location',
        'type' => 'text',
    ],
    [
        'name'          => 'Map',
        'id'             => 'map',
        'type'           => 'osm',
        'address_field' => 'location',
    ],
],
];

// Add more field groups if you want
// $meta_boxes[] = ...

return $meta_boxes;
} );

```

Each field group has several settings and a list of fields, which we will cover below.

Field group settings

Each field group has several settings for the location or appearance. Please see the list below:

Name	Description
<code>id</code>	ID, must be unique. Optional. If it is absent, it will be generated from the title.
<code>title</code>	The field group title. Required.
<code>post_types</code>	Custom post types which the field group is for. It can be a string or an array of slugs. Must be in lowercase (like the slug). Optional. Default: <code>post</code> .
<code>context</code>	Where the field group is displayed. See below for a list of field group contexts. Optional.

Name	Description
style	Whether to keep the default WordPress field group style (<code>default</code>) or remove the wrapper box and display the fields seamlessly (<code>seamless</code>).
closed	Whether to collapse the field group when the page loads? Optional. Default: <code>false</code> .
priority	Priority within the context where the box is displayed (<code>high</code> or <code>low</code>). Optional. Default: <code>high</code> .
default_hidden	Hide the field group by default (<code>true</code> or <code>false</code>)? The field group can be toggled using the checkbox option in screen Help (on the top right). Optional. Default <code>false</code> .
autosave	Auto save the custom fields' values (like post content and title)? Optional. Default: <code>false</code> .
media_modal	Add custom fields to media modal when viewing/editing an attachment. Works only when <code>post_types</code> is or contains <code>attachment</code> . Optional. Default <code>false</code> .
class	Custom CSS class for the field group wrapper. Optional.

:::info Media modal limitation

Only simple fields such as text, select, radio, checkbox work in the media modal. Other fields that require custom JavaScript do not work.

:::

Contexts

The plugin supports the following contexts (locations) where a field group can appear:

Name	Description
normal	Below the post editor. This is the default value.
advanced	Below the <code>normal</code> section.
side	On the right sidebar.
form_top	Top of the post form, before the post title
after_title	After post title
after_editor	After the post content editor, but before <code>normal</code> section
before_permalink	Before permalink

:::warning Gutenberg

Gutenberg editor only supports `normal` and `side` contexts. Other extra contexts are not supported.

Fields

Fields are added to a field group via the key `fields`. Each field is an array of settings.

Meta Box supports more than 40 field types. They share some common settings but also offer unique settings per field type.

Field types

When adding a field, you need to know what type it is and how it works. Understanding that helps you choose the right type of field and, therefore, the right type of data that you want to add to your posts.

Below is the list of supported field types in alphabetical order with a brief description. The field type key is used for reference in code. For how they look like and how to use them, please see details in the [Field types](#) menu.

These are the basic and most used field types that do not require any extra library. The UI of these fields is WordPress-native.

Type	Key	Description
Checkbox	<code>checkbox</code>	A simple checkbox, usually used for Yes/No question
Checkbox list	<code>checkbox_list</code>	A list of checkboxes where you can select multiple choices
Radio	<code>radio</code>	Radio input where you can select only one choice
Select	<code>select</code>	Select dropdown where you can select one or multiple choice
Text	<code>text</code>	A single-line text input
Textarea	<code>textarea</code>	A paragraph text input

These are the advanced field types that usually need an additional library to provide a user-friendly UI.

Type	Key	Description
Background	<code>background</code>	Set background properties

Type	Key	Description
Button	<code>button</code>	A simple button, usually used for JavaScript triggers
Button group	<code>button_group</code>	Select one or multiple choices by enabling button(s) from a group
Color picker	<code>color</code>	Color picker
Custom HTML	<code>custom_html</code>	Output custom HTML content
Date picker	<code>date</code>	Date picker
Datetime picker	<code>datetime</code>	Date and time picker
Hidden	<code>hidden</code>	For storing a default hidden value
Image Select	<code>image_select</code>	Select a choice with images
Key Value	<code>key_value</code>	Add an unlimited group of key-value pairs
Google maps	<code>map</code>	Google Maps
oEmbed	<code>oembed</code>	Input for media from Youtube, Vimeo, and all supported sites by WordPress
Open Street Maps	<code>osm</code>	Open Street Maps
Password	<code>password</code>	For entering a password
Select advanced	<code>select_advanced</code>	Beautiful select dropdown using select2 library
Slider	<code>slider</code>	jQuery UI slider
Switch	<code>switch</code>	On/off switch with iOS style
Time picker	<code>time</code>	Time picker
WYSIWYG editor	<code>wysiwyg</code>	WordPress editor

Besides, some field types that are rarely used and we would not recommend using them because their UI is not as good and native as other fields.

Type	Key	Description
Autocomplete	<code>autocomplete</code>	Text input that uses an autocomplete library to suggest user input. Not recommended. Use the Select or Select advanced field type instead.
Fieldset text	<code>fieldset_text</code>	Group of text inputs. Not recommended. Use the Group field type instead.

Type	Key	Description
Text list	<code>text_list</code>	Group of text inputs. Similar to Fieldset text, but has a different UI. Not recommended. Use the Group field type instead.

:::warning

These are field types that use the built-in browser UI without extra libraries. The UI might be different across operating systems and browsers. Please use with care.

:::

Type	Key	Description
Email	<code>email</code>	For entering an email address with browser validation
Number	<code>number</code>	For entering a number with browser validation
Range	<code>range</code>	A slider for selecting a number
URL	<code>url</code>	An input for URL with browser validation

These are field types that help you select a WordPress object.

Type	Key	Description
Post	<code>post</code>	For selecting posts
Sidebar	<code>sidebar</code>	For selecting sidebars
Taxonomy	<code>taxonomy</code>	For selecting taxonomy terms. Does not save term IDs in post meta, but sets post terms.
Taxonomy advanced	<code>taxonomy_advanced</code>	For selecting taxonomy terms and saving term IDs in post meta as a comma-separated string. It does not set post terms.
User	<code>user</code>	For selecting users

These are field types that help you upload media files.

Type	Key	Description
File	<code>file</code>	Simple file upload with default UI like <code><input type="file" /></code> . Not recommended. Use File advanced instead.

Type	Key	Description
File advanced	<code>file_advanced</code>	Multiple file uploads with WordPress media popup
File input	<code>file_input</code>	A text input for entering a file URL with the ability to select a file from the Media Library
File upload	<code>file_upload</code>	Multiple file uploads with a drag and drop area
Image	<code>image</code>	Simple image upload with default UI like <code><input type="file" /></code> . Not recommended. Use Image advanced instead.
Image advanced	<code>image_advanced</code>	Multiple image uploads with WordPress media popup, usually used for a gallery
Image upload	<code>image_upload</code>	Multiple image uploads with a drag and drop area
Single image	<code>single_image</code>	Single image upload with WordPress media popup
Video	<code>video</code>	Multiple video uploads with WordPress media popup

These are field types that help you organize and improve the UI of the fields.

Type	Key	Description
Divider	<code>divider</code>	Simple horizontal line
Heading	<code>heading</code>	Heading text
Group	<code>group</code>	For creating nesting groups of fields. Requires MB Group extension.
Tab	<code>tab</code>	For organizing fields in tabs. Requires MB Tabs extension.

Field settings

Each field contains settings to determine where and how data is loaded and saved. All fields share some common settings, but also offer unique settings per field type. There are also settings from extensions which are explained on each extension docs.

Below is the list of settings with a brief description. The keys are for reference in code.

general settings

Name	Key	Description
Label	name	Field label. Optional. If empty, the field input is 100% width.
ID	id	Field ID. Required and must be unique. It will be used as meta_key when saving to the database. Use only numbers, letters, and underscores (and rarely dashes).
Type	type	Field type. Required.
Label description	label_description	Label description, displayed below the field label. Optional.
Input description	desc	Field description, displayed below the field input. Optional.
Default value	std	Default value. Optional.
Placeholder	placeholder	Placeholder text for the input or select box. Optional.
Required	required	Whether the field is required (true or false). Optional. Default false.
Disabled	disabled	Whether the field is disabled (true or false). Optional. Default false.
Read only	readonly	Whether the field is read only (true or false). Optional. Default false.
Multiple	multiple	Does the field have multiple values (like the select field)? Optional. Default false.
Cloneable	clone	Is the field clonable (repeatable)? true or false. Optional. Default false.
Sortable	sort_clone	Ability to drag-and-drop reorder clones (true or false). Optional. Default false.
Clone default value	clone_default	Clone the default value of fields? true or false (default).
Clone as multiple	clone_as_multiple	Whether to store clone values in multiple rows in the database? Optional. Default false.
Max number of clones	max_clone	Maximum number of clones. Optional. Default 0 (unlimited).
Min number of clones	min_clone	Minimum number of clones. Optional. Default 0.
Add more text	add_button	The text for Add more clone button. Optional. Default "+ Add more".

advanced settings

Name	Key	Description
Before	<code>before</code>	Custom HTML outputted before field's HTML.
After	<code>after</code>	Custom HTML outputted after field's HTML.
Custom CSS class	<code>class</code>	Custom CSS class, in case you want to customize the field. Optional.
Custom sanitize callback	<code>sanitize_callback</code>	Custom PHP callback for sanitizing field value before saving into the database. Set it to <code>none</code> to bypass the sanitization. See more details .
Save field value	<code>save_field</code>	Whether to save field value. Optional. Default <code>true</code> . This option does not work in the block editor (Gutenberg).
Hide from block bindings?	<code>hide_from_block_bindings</code>	Hide this field from the block editor bindings UI. Optional. Default <code>false</code> . See more details .
Custom HTML5 attributes	<code>attributes</code>	Custom attributes for inputs. See more details .
Validation	<code>validation</code>	Validation rules for fields. Optional. See more details .
Custom settings	N/A	Custom field settings, useful when you want to add your settings to fields.

:::tip Field ID prefix

You can add a prefix to field IDs to prevent using the same ID with other scripts. If you want to hide the fields in the default WordPress **Custom Fields** meta box, use underscore (`_`) as the prefix.

:::

Field-specific settings

Besides all common settings, each field type can have its own settings. Please see more details for each field type on the left menu.

:::tip Code examples

To save time reading and writing settings for fields, we have already prepared some code examples that you can get from [Meta Box Code Snippet Library](#).

Shorthand syntax

For **text** field type, you can omit the `type` settings, like this:

```
add_filter( 'rwmb_meta_boxes', function ( $meta_boxes ) {
    $meta_boxes[] = [
        'title'      => 'Event details',
        'post_types' => 'event',
        'fields'      => [
            [
                'name' => 'Date and time',
                'id'   => 'datetime',
                'type' => 'datetime',
            ],
            // highlight-start
            [
                'name' => 'Location',
                'id'   => 'location',
            ],
            // highlight-end
            [
                'name'      => 'Map',
                'id'        => 'map',
                'type'       => 'osm',
                'address_field' => 'location',
            ],
        ],
    ];

    return $meta_boxes;
} );
```

And if the `id` of the field can be auto-generated from the field name, you can omit it and **define the field as a simple string**:

```
add_filter( 'rwmb_meta_boxes', function ( $meta_boxes ) {
    $meta_boxes[] = [
```

```

    'title'      => 'Event details',
    'post_types' => 'event',
    'fields'     => [
        [
            'name' => 'Date and time',
            'id'   => 'datetime',
            'type' => 'datetime',
        ],
        // highlight-next-line
        'Location',
        [
            'name'      => 'Map',
            'id'        => 'map',
            'type'      => 'osm',
            'address_field' => 'location',
        ],
    ],
];

return $meta_boxes;
} );

```

Video tutorial

This video shows you all the field types and field settings:

FAQ

The mechanism of `std` in Meta Box works only if the **field group has not been saved before**. It means all fields in that field group, not just the specific field that you set the `std` for. So if there is any field that already has value, then `std` will not work for other fields, even new fields you have just added.

Examples:

When you create a new post, then no fields have values (of course), then `std` works for all fields.

When you edit an existing post that has a field group, then some fields might have values. Therefore, `std` does not work for all fields. In this case, if you edit the field group and add a new field, `std` still does not work for that new field (even it has no value before), because the field group has been saved before.

There are some situations where the context does not work as expected. That is probably because you have dragged and dropped the field groups to reorder them. If you have, then WordPress will save the position/location of them and use the saved position instead of the value in the `context` parameter. The order of field groups is saved in the user meta `meta-box-order_{screen id}` as follows:

meta box order

In this case, deleting this user meta from the database will make the context work again.

		#1	
			2026-08-29 23:36:17 UTC
			2026-08-29 23:36:17 UTC