add_editor_style(array|string $stylesheet = ‘editor-style.css’)

Adds callback for custom TinyMCE editor stylesheets.

Description

The parameter $stylesheet is the name of the stylesheet, relative to the theme root. It also accepts an array of stylesheets.
It is optional and defaults to ‘editor-style.css’.

This function automatically adds another stylesheet with -rtl prefix, e.g. editor-style-rtl.css.
If that file doesn’t exist, it is removed before adding the stylesheet(s) to TinyMCE.
If an array of stylesheets is passed to add_editor_style() , RTL is only added for the first stylesheet.

Since version 3.4 the TinyMCE body has .rtl CSS class.
It is a better option to use that class and add any RTL styles to the main stylesheet.

Parameters

`$stylesheet` `array|string` optional
Stylesheet name or array thereof, relative to theme root.
Defaults to `'editor-style.css'`

Default: `'editor-style.css'`

More Information

Allows theme developers to link a custom stylesheet file to the TinyMCE visual editor. The function tests for the existence of the relative path(s) given as the `$stylesheet` argument against the current theme directory and links the file(s) on success. If no `$stylesheet` argument is specified, the function will test for the existence of the default editor stylesheet file, `editor-style.css`, against the current theme directory, and link that file on success.

If a child theme is used, both the current child and parent theme directories are tested and both the files with the same relative path are linked with this single call if they are found.

To link a stylesheet file from a location other than the current theme directory, such as under your plugin directory, use a filter attached to the mce_css hook instead.

Source

wp-includes/theme.php

Expand code Copy

```
function add_editor_style( $stylesheet = 'editor-style.css' ) {
```

```
    global $editor_styles;

    add_theme_support( 'editor-style' );

    $editor_styles = (array) $editor_styles;
    $stylesheet    = (array) $stylesheet;

    if ( is_rtl() ) {
        $rtl_stylesheet = str_replace( '.css', '-rtl.css', $stylesheet[0] );
        $stylesheet[]   = $rtl_stylesheet;
    }
```

View all references·View on Trac·View on GitHub

Related

Uses

add_theme_support()	wp-includes/theme.php
is_rtl()	wp-includes/l10n.php

Changelog

Version	Description
3.0.0	Introduced.

User Contributed Notes

Codex 8 years ago — 16

Basic Example

Add the following to the functions.php file of your theme.

Copy

```
/**
 * Registers an editor stylesheet for the theme.
 */
function wpdocs_theme_add_editor_styles() {
    add_editor_style( 'custom-editor-style.css' );
}
add_action( 'admin_init', 'wpdocs_theme_add_editor_styles' );
```

Next, create a file named custom-editor-style.css in your themes root directory. Any CSS rules added to that file will be reflected within the TinyMCE visual editor. The contents of the file might look like this:

Copy

```
body#tinymce.wp-editor {
    font-family: Arial, Helvetica, sans-serif;
    margin: 10px;
}

body#tinymce.wp-editor a {
    color: #4CA6CF;
}
```

Show feedback (2)Log in to add feedback

Code Muffin3 years ago 14

Gutenberg

This works with Gutenberg. Your CSS will be added as inline styles. All selectors will be prefixed with: `.editor-styles-wrapper` (except `body`, see below).

All the info below is related to using this function with Gutenberg.

Body Styles

Styles targeting the body will be changed to target `.editor-styles-wrapper` instead. This lets you to customise the Gutenberg editor pane directly, e.g. to change its padding. You can also set defaults (like font family, color, etc) by adding styles to the body class.

Example: `body { color: red }` becomes `.editor-styles-wrapper { color: red }`

CSS Resets

If you're using the same stylesheet for both your frontend and editor styles, be careful about using a CSS reset. It will affect all elements within the editor, including admin controls for blocks — such as TinyMCE's GUI buttons in the Classic block, or ACF block fields in edit mode.

You may prefer to move your reset of your main styles and enqueue it independently on the frontend instead, perhaps with a dedicated, customized reset for the editor.

Using @import

I've seen developers report that using `@import` in your CSS causes issues. If this happens to you, it would be better to add the imported file by other means, either as an additional array item in `add_editor_style`, or by enqueuing it as an admin style on just editor pages (see other user notes that use `$pagenow` for this).

Local Development + SSL

If you're working locally, be aware that using an absolute URL (eg. via get_template_directory_uri) won't work if you're using HTTPS with a self-signed certificate. This is due to wp_remote_get being used to retrieve your stylesheet, which fails (silently) if a requested URL's certificate isn't included in WordPress' internal store of allowed certificates.

Workarounds include: Using a path relative to your theme root instead (ideal), using an absolute HTTP URL instead of HTTPS, and disabling the SSL verification with https_ssl_verify (not recommended for security reasons).

Code Context

The code that retrieves the styles uses either wp_remote_get for absolute URLs, or `file_get_contents` for files relative to the theme root. You can find this code in block-editor.php, at the function `get_block_editor_theme_styles()`. It's called from edit-form-blocks.php.

Show feedback (1)Log in to add feedback

Tony Hayes8 years ago 9

If you want to add styles dynamically (eg. from theme mods) you can use the tiny_mce_before_init filter and add them to the content_style key.

Copy

```
add_filter('tiny_mce_before_init','wpdocs_theme_editor_dynamic_styles');
function wpdocs_theme_editor_dynamic_styles( $mceInit ) {
    $styles = 'body.mce-content-body { background-color: #' . get_theme_mod( 'background-color', '#
    if ( isset( $mceInit['content_style'] ) ) {
        $mceInit['content_style'] .= ' ' . $styles . ' ';
    } else {
        $mceInit['content_style'] = $styles . ' ';
    }
    return $mceInit;
}
```

Note that any new lines or double quotes should be removed or double escaped in your CSS.

Log in to add feedback

Codex8 years ago 2

Using Google Fonts

Google Fonts API provides a single URL for a CSS file that can include multiple variants of a type face, separated by commas. Commas in a URL need to be encoded before the string can be passed to add_editor_style.

Copy

```
/**
 * Registers an editor stylesheet for the current theme.
 */
function wpdocs_theme_add_editor_styles() {
    $font_url = str_replace( ',', '%2C', '//fonts.googleapis.com/css?family=Lato:300,400,700' );
    add_editor_style( $font_url );
}
add_action( 'after_setup_theme', 'wpdocs_theme_add_editor_styles' );
```

Log in to add feedback

Codex8 years ago 2

Choosing Styles Based on Post Type

To link a custom editor stylesheet file based on the post type being edited, you can use the following in the functions.php file of your theme. This assumes the stylesheet files with names in the form of editor-style-{post_type}.css are present directly under your theme directory.

Expand code Copy

```
/**
 * Registers an editor stylesheet for the current theme.
 *
 * @global WP_Post $post Global post object.
 */
function wpdocs_theme_add_editor_styles() {
    global $post;

    $my_post_type = 'posttype';

    // New post (init hook).
    if ( false !== stristr( $_SERVER['REQUEST_URI'], 'post-new.php' )
            && ( isset( $_GET['post_type'] ) === true && $my_post_type == $_GET['post_type'] )
```

Note that the pre_get_posts action hook is used to ensure that the post type is already determined but, at the same time, that TinyMCE has not been configured yet. That hook is not run when creating new posts, that is why we need to use it in combination with the init hook to achieve a consistent result.

Log in to add feedback

Brad Davis 6 years ago 1

If you are keeping your style files in a sub-directory, eg, css, you add the editor style with:

Copy

```
/**
 * Registers an editor stylesheet in a sub-directory.
 */
function add_editor_styles_sub_dir() {
    add_editor_style( trailingslashit( get_template_directory_uri() ) . 'css/editor-style.css' );
}
add_action( 'after_setup_theme', 'add_editor_styles_sub_dir' );
```

Log in to add feedback

goulvench 1 year ago 0

Using Gutenberg, calling `add_theme_support( 'editor-styles' );` is required before calling `add_editor_style()`, even though the function is supposed to add theme support for editor styles.

So if you want to add your own styles to the Gutenberg editor, be sure to call both functions:

Copy

```
add_action( 'admin_init', 'wpdocs_add_editor_styles' );
function wpdocs_add_editor_styles() {
  add_theme_support( 'editor-styles' );
  add_editor_style( 'editor-style.css' );
}
```

Show feedback (1)Log in to add feedback

cweiser10 months ago 0

Note that the Block Editor is unable to parse any file containing CSS @layers. Using CSS layers will prevent the entire file from being applied to the editor. Ex:

Copy

```
/* some-component.scss */
.wp-button__link {
  border: solid 1px green;
}
@layer theme {
  .wp-button__link {
    color: red !important;
  }
}
/* Neither of these styles will be applied due to usage of @layer */
```

Show feedback (1)Log in to add feedback

BigupJeff5 months ago 0

To add styles from a plugin use a URL in place of a path relative to the theme root.

Copy

```
// Will NOT work in plugin.
add_editor_style( 'build/css/editor.css' );

// WILL work in plugin.
add_editor_style( 'https://mywebsite.com/wp-content/plugins/myplugin/build/css/editor.css' );
```

Log in to add feedback

olik97 years ago -1

The example above can be rewritten simpler:

Expand code Copy

```
function value( $ar, $key, $default = '' ) {
    if ( is_array( $ar ) && isset( $ar[ $key ] ) ) { return $ar[ $key ]; }
    //else

    return $default;
}

/**
 * Registers an editor stylesheet for the current theme.
 *
 * @global WP_Post $post Global post object.
 */
function wpdocs_theme_add_editor_styles() {
```

Log in to add feedback

Codex 8 years ago -14

Reusing Your Theme Styles

You can reuse the styles from your theme stylesheet file in your custom editor stylesheet file using the @import CSS rule. Working on the previous example, put the following instead into the custom-editor-style.css file.

Copy

```
@import url( 'style.css' );

/* Add overwrites as needed so that the content of the editor field is attractive and not broken */
body { padding: 0; background: #fff; }
```

If necessary, change 'style.css' to the path to your theme stylesheet, relative to the custom-editor-style.css file.

Log in to add feedback

You must log in before being able to contribute a note or feedback.

WordPress.tv ↗
Get Involved
Events
Donate ↗
Swag Store ↗
WordPress.com ↗
Matt ↗
bbPress ↗
BuddyPress ↗

CODE IS POETRY