Block Style Variations

- What are block styles?
- PHP-based block styles
  - Registering blocks styles with PHP
  - Unregistering block styles with PHP
- JavaScript-based block styles
  - Registering block styles with JavaScript
  - Unregistering block styles with JavaScript
- Customizing block styles via theme.json

Block style variations (block styles, for short) let you create alternate styles for individual blocks. Registered styles appear in the user interface, allowing users to quickly switch between the default style and any alternates.

Many features, unfortunately, use the term “variations” in their name. This can be confusing at times. The biggest thing to note is that block style variations are not the same thing as global style variations and block variations.

Often, you will see block style variations referred to as “block styles,” as they are in this article, but this should also not be confused with block stylesheets. The two features can be used to work together, but they are not the same thing.

What are block styles?

Under the hood, block styles are nothing more than CSS classes added to a block’s wrapping element with the name of `.is-style-{name}`. This allows you to add custom CSS to alter the block’s design in some way.

In this screenshot, you can see multiple block styles registered for the Image block under the **Styles** panel, and the core **Rounded** option is selected:

WordPress post editor showing an Image block with a rounded style in the content canvas.

All that is happening here is that core WordPress and the theme have both registered block styles for the Image block, and the user has selected one of those styles. Then the CSS for that registered style is applied to a block.

Ultimately, block styles are just a standard way to add a class to a block and customize it with CSS.

Users can only apply one style to a block at a time. They cannot combine them using the standard design tools. If you need to allow more options, you might consider building custom design tools.

Read the Beyond Block Styles tutorial series on the WordPress Developer Blog to learn how.

PHP-based block styles

Most theme authors will want to register block styles with PHP. Skip ahead to the “JavaScript-based block styles” section if you need or want to work with them via JavaScript.

Registering blocks styles with PHP

To register a block style, use the `register_block_style()` PHP function. The function’s signature looks like this:

```
register_block_style(
	string $block_name,
	array $style_properties
): bool
```

The function accepts two parameters and returns `true` or `false`, depending on whether the registration was successful:

- **`$block_name`:** The name of the block, including both the namespace and slug (e.g., `core/image`).
- **`$style_properties`:** An array of arguments that can be used to configure the style:
  - **`name`:** *(Required)* A unique identifier/slug, which is used to generate the class (e.g., `is-style-{name}`).
  - **`label`:** *(Required)* An human-readable label, which may be translated.
  - **`inline_style`:** Inline CSS to be printed when the style is in use.
  - **`style_handle`:** A handle for a registered stylesheet to load for the style.
  - **`is_default`:** Whether the style should be selected as the default for the block (defaults to `false`).

`style_handle` currently has a bug where it only enqueues the stylesheet in the editor but not on the front end. Until that linked ticket is addressed, its use is not recommended.

Let’s try creating a block style that gives the Image block a “hand drawn” look so that it looks like this when selected in the editor:

WordPress post editor showing an Image block with a hand-drawn style in the content canvas.

To register this block style, you need to configure it via the `register_block_style()` function and add it as a callback on the `init` hook.

Add the following code to your theme’s `functions.php` file to test it out:

```
add_action( 'init', 'themeslug_register_block_styles' );

function themeslug_register_block_styles() {
	register_block_style( 'core/image', array(
		'name'         => 'hand-drawn',
		'label'        => __( 'Hand Drawn', 'themeslug' ),
		'inline_style' => '.wp-block-image.is-style-hand-drawn img {
			border: 2px solid currentColor;
			overflow: hidden;
			box-shadow: 0 4px  10px 0 rgba( 0, 0, 0, 0.3 );
			border-radius: 255px 15px 225px 15px/15px 225px 15px 255px !important;
		}'
    ) );
}
```

If your `inline_style` argument is more than a few lines of CSS code, you may want to add your CSS to a custom block stylesheet instead.

Unregistering block styles with PHP

To unregister a block style that has been registered with PHP, use the `unregister_block_style()` function. Here is a look at its signature:

```
unregister_block_style(
	string $block_name,
	string $block_style_name
): bool
```

The function accepts two parameters and returns `true` or `false`, depending on whether the registration was successful:

- **`$block_name`:** The name of the block, including both its namespace and slug (e.g., `core/image`).
- **`$block_style_name`:** The name/slug of a block style that was registered via PHP.

To unregister the `hand-drawn` block style you registered earlier, you need to add an action function to the `init` hook (note the action call has a later priority of `99` so that it runs after the earlier registration function, which defaults to `10`):

```
add_action( 'init', 'themeslug_register_block_styles', 999 );

function themeslug_register_block_styles() {
	unregister_block_style( 'core/image', 'hand-drawn' );
}
```

You can only unregister block styles with PHP if they have been registered via PHP. If the block style was registered with JavaScript, you must also use JavaScript to unregister it. For more information, see the “Unregistering block styles with JavaScript” section below.

JavaScript-based block styles

To register or unregister block styles via JavaScript, you need to load a JavaScript file on the `enqueue_block_editor_assets` hook.

First, create an `/assets/js/block-editor.js` file in your theme (you can change this name or add it anywhere, but the example below assumes this name and location).

Then add this code to your theme’s `functions.php` file to enqueue your script:

```
add_action( 'enqueue_block_editor_assets', 'themeslug_block_editor_assets' );

function themeslug_block_editor_assets() {
	wp_enqueue_script(
		'themeslug-block-editor',
		get_theme_file_uri( 'assets/js/block-editor.js' ),
		array(
			'wp-blocks',
			'wp-dom-ready',
			'wp-edit-post'
		)
	);
```

```
}
```

Registering block styles with JavaScript

To register a block style via JavaScript, you'll use the `registerBlockStyle()` function, which works similarly to the `register_block_style()` PHP function. It just uses JavaScript syntax.

Let's use the same example from earlier and register a "hand drawn" Image block style. Add this code to your `/assets/js/block-editor.js` file:

```
wp.blocks.registerBlockStyle( 'core/image', {
	name: 'hand-drawn',
	label: 'Hand Drawn'
} );
```

You can also set the `isDefault` argument to `true` if this is meant to be the default block style (it's `false` by default).

For adding your custom CSS, you'll need to load a custom block stylesheet or add your CSS by some other means.

Unregistering block styles with JavaScript

There is also an `unregisterBlockStyle()` JavaScript function that is equivalent to the `unregister_block_style()` PHP function. You can use this to unregister any block styles that have been registered via JavaScript.

To avoid a race condition where there's a conflict between when a block style is registered versus when it is unregistered, you need to ensure that you unregister a block style after it has already been registered. The best way to do this is to add it inside a function callback on `wp.domReady`, which ensures that you unregister after the DOM has loaded.

To unregister the `hand-drawn` block style that you registered via JavaScript, add this to your `/assets/js/block-editor.js` file:

```
wp.domReady( function () {
	wp.blocks.unregisterBlockStyle( 'core/image', 'hand-drawn' );
} );
```

You can only unregister block styles with JavaScript if they have been registered via JavaScript. If the block style was registered with PHP, you must also use PHP to unregister it. For more

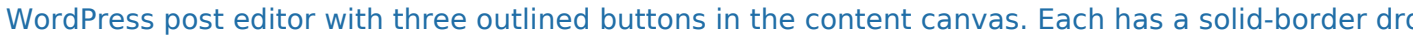
information, see the “Unregistering block styles with PHP” section above.

Customizing block styles via theme.json

It’s possible to customize the design of the core WordPress block styles. This means that you might not need to add custom CSS at all to get the look you’re after.

It is also recommended to use this method if possible for your customizations because they appear in the **Appearance > Editor > Styles** interface. Plus, your theme users can make their own changes to them if desired.

Let’s try to modify the Outline style of core Button block so that it has a drop-shadow that creates a double-border effect:



To do this, open your `theme.json` file. You’ll need to target the `styles.blocks.core/button.variations.outline` property. It works just like any other style that you can add via `theme.json`.

Try this code snippet in your `theme.json` file:

```
{
□"version": 2,
□"styles": {
□□"blocks": {
□□□"core/button": {
□□□□"variations": {
□□□□□"outline": {
□□□□□□"border": {
□□□□□□□"color": "var:preset|color|black",
□□□□□□□"radius": "0",
□□□□□□□"style": "solid",
□□□□□□□"width": "3px"
□□□□□□},
□□□□□□"shadow": "var:preset|shadow|outlined",
□□□□□□"spacing": {
□□□□□□□"padding": {
```

```
□□□□□□□□□"top": "0.5rem",
□□□□□□□□□"bottom": "0.5rem",
□□□□□□□□□"left": "1.5rem",
□□□□□□□□□"right": "1.5rem"
□□□□□□□□}
□□□□□□□}
□□□□□□}
□□□□□}
□□□□}
□□}
□}
}
```

Feel free to adjust that and customize it to your liking. Then have fun modifying other block styles.

It is not currently possible to customize your custom-registered block styles via `theme.json`. You can only style those registered by core WordPress at the moment. For more information, check out the related feature request on GitHub.

The following are the core WordPress blocks and their styles that you can customize via `theme.json` :

- **`core/button`**: `outline`, `fill`
- **`core/image`**: `rounded`
- **`core/quote`**: `plain`
- **`core/site-logo`**: `rounded`
- **`core/separator`**: `wide`, `dots`
- **`core/social-links`**: `logos-only`, `pill-shape`
- **`core/table`**: `stripes`
- **`core/tag-cloud`**: `outline`

For a deeper dive into styling core blocks, read Customizing core block style variations via theme.json on the WordPress Developer Blog.

□□□ #2
□ □□□ □□□ 2026-08-29 15:36:32 UTC
□ □□□ □□□ 2026-08-29 15:36:32 UTC