Using Presets

- What are presets?
- Applying presets as styles
  - Referencing custom presets
- Available presets

In the `theme.json` Settings documentation, you learned how to register custom presets, such as a color palette, font sizes, a spacing scale, and more. In the Styles documentation, you've learned how to apply CSS styles via `theme.json`.

Now it's time to combine these two concepts by applying styles using the presets that you've registered.

In the styles documentation, you will often see examples using hard-coded CSS values. This is primarily for the sake of demonstration. You will almost always use registered presets instead. This makes it much easier to manage everything from a single location.

What are presets?

Essentially, presets are options that you register—generally for users to select from the UI—that map to CSS custom properties.

For example, when you learned how to register custom font sizes, you added your presets to `settings.typography.fontSizes`. WordPress takes each of those font sizes and creates a CSS custom property with the name of `--wp--preset--font-size--{$slug}`.

WordPress itself, your theme, plugins, and even users can register presets for the various features that are supported. And WordPress creates CSS custom properties for all registered presets with the name of `--wp--preset--{$feature}-{$slug}`.

Let's look at a basic example of a custom color palette with three colors:

```
{
	"version": 2,
	"settings": {
		"color": {
			"palette": [
				{
					"color": "#ffffff",
					"name": "Base",
					"slug": "base"
				},
				{
					"color": "#000000",
					"name": "Contrast",
					"slug": "contrast"
				},
				{
					"color": "#89cff0",
					"name": "Primary",
					"slug": "primary"
				}
			]
		}
	}
}
```

This creates three CSS custom properties, which WordPress will output in the editor and on the front end:

```
body {
	--wp--preset--color--base: #ffffff;
	--wp--preset--color--contrast: #000000;
	--wp--preset--color--primary: #89cff0;
}
```

At the end of the day, presets are merely a standardized method of creating options in the interface and generating CSS custom properties.

Applying presets as styles

Because presets are registered as settings, they are available to select in the user interface. But you must apply them as styles if you want to use them as part of your theme's default design.

Let's build off the custom color palette from the previous section. Suppose you wanted to apply these styles using your registered presets:

- The site's background color should use the `base` preset.
- The site's primary text color should use the `contrast` preset.
- Link colors should use the `primary` preset.

To reference presets in `theme.json`, there is a special syntax that you can use: `var:preset|$feature|$slug`. So `base` color in this instance would be `var:preset|color|base`.

With that plan in mind and using what you've already learned from this chapter, try your hand at recreating this in `theme.json`. Your code should look like this:

```
{
□"version": 2,
□"settings": {
□□"color": {
□□□"palette": [
□□□□{
□□□□□"color": "#ffffff",
□□□□□"name": "Base",
□□□□□"slug": "base"
□□□□},
□□□□{
□□□□□"color": "#000000",
□□□□□"name": "Contrast",
□□□□□"slug": "contrast"
□□□□},
□□□□{
□□□□□"color": "#89cff0",
□□□□□"name": "Primary",
□□□□□"slug": "primary"
□□□□}
□□□]
□□}
□},
□"styles": {
□□"color": {
□□□"text": "var:preset|color|contrast",
```

```
□□□"background": "var:preset|color|base"
□□},
□□"elements": {
□□□"link": {
□□□□"color": {
□□□□□"text": "var:preset|color|primary"
□□□□}
□□□}
□□}
□}
}
```

If you test your site on the front end or via the editor, you should see that your colors have correctly applied:

WordPress post editor with a couple of demo paragraphs. The image reflects the black text color an

The next step now is to figure out which elements and blocks you want to apply other presets to. Depending on the complexity of your theme, this can be as simple as a few custom colors to hundreds of lines of JSON code. Really, what you do from here is entirely up to you.

Technically, you can reference presets using the CSS syntax of `var( --wp--preset--{$feature}--{$slug} )`. But the WordPress `var:preset|$feature|$slug` syntax works much better and always appears correctly throughout the interface in the WordPress admin. Save the CSS syntax for when you are actually writing CSS.

Referencing custom presets

In the Custom Settings documentation, you learned how to create “custom” presets. These are non-standard CSS custom properties that you can register, and WordPress generates the CSS output for you.

These use a different naming convention in comparison to standard presets. Essentially, whenever you used the term `preset` in your code for standard presets, you would exchange that for `custom` when dealing with custom presets.

Let’s walk through an example. Assume that you wanted to register some CSS custom properties for handling line heights in your theme design. You would add this to your `theme.json` file:

```
{
□"version": 2,
□"settings": {
```

```
□□"custom": {
□□□"lineHeight": {
□□□□"xs": "1",
□□□□"sm": "1.25",
□□□□"md": "1.5",
□□□□"lg": "1.75"
□□□}
□□}
□}
}
```

WordPress will generate these custom line heights as CSS custom properties in the editor and on the front end:

```
body {
□--wp--custom--line-height--xs: 1;
□--wp--custom--line-height--sm: 1.25;
□--wp--custom--line-height--md: 1.5;
□--wp--custom--line-height--lg: 1.75;
}
```

To reference these as styles in `theme.json`, you would use the `var:custom` syntax instead of `var:preset`.

Suppose you wanted to apply the registered `md` line-height size as the default line height at the root/global level. For that, you would need to target `styles.typography.lineHeight`.

Here is what the full code would look like in `theme.json`:

```
{
□"version": 2,
□"settings": {
□□"custom": {
□□□"lineHeight": {
□□□□"xs": "1",
□□□□"sm": "1.25",
□□□□"md": "1.5",
□□□□"lg": "1.75"
□□□}
□□}
□},
```

```
"styles": {
"typography": {
"lineHeight": "var:custom|line-height|md"
}
}
}
```

Of course, you could use your other custom line-height presets for styling other elements and blocks.

Available presets

WordPress has several features that you can register presets for. You can find these presets in their corresponding settings docs (the specific properties are noted in parentheses):

- Color (`palette`)
- Shadow (`presets`)
- Spacing (`spacingScale`, `spacingSizes`)
- Typography (`fontSizes`, `fontFamily`)
- Custom

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