What's new for developers

What's new for developers? (August 2025)

It's already August, and we have a freshly printed Roadmap to 6.9. Everything from evolving the site editor to refining content creation to performance improvements is on the docket. That means the last few months of 2025 are bound to be exciting ones.

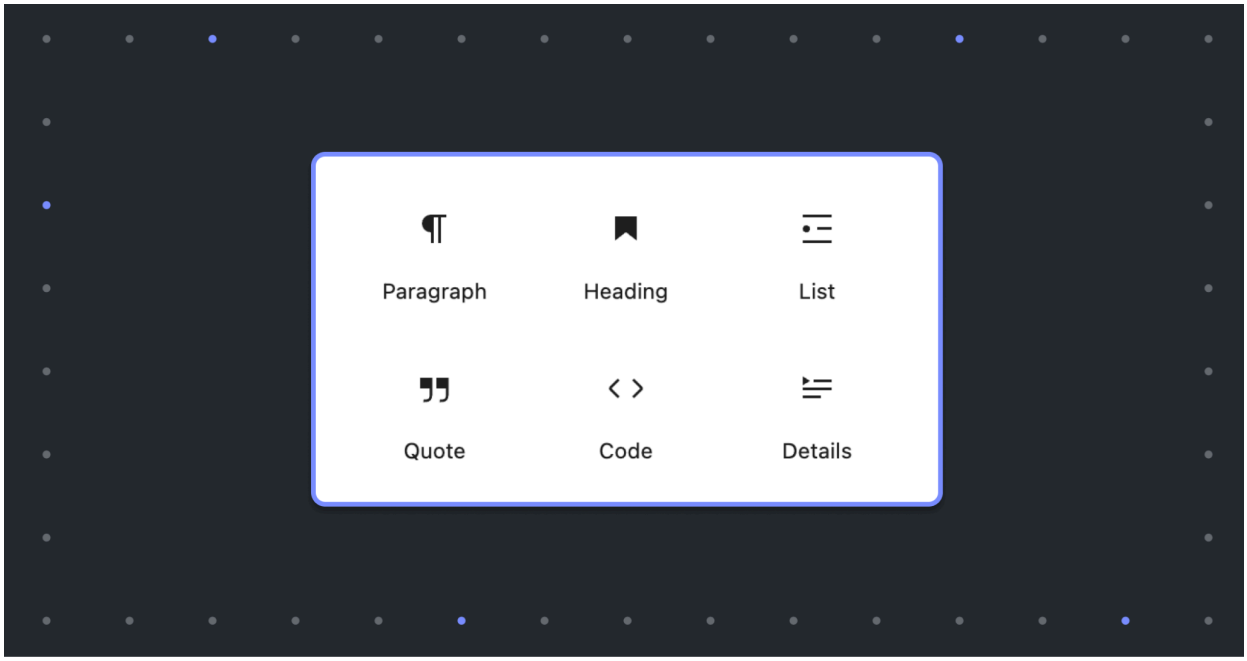
If you're up for some *light* reading before jumping into the rest of this month's news, I encourage you to at least peruse:

- Admin Materials and Surfaces: A discussion on the structure of the future admin and mapping out the main elements that compose it.
- Abilities & Workflows Overview: An overview of the elements needed to unify admin commands, shortcuts, and AI-assisted workflows.
- Admin Redesign: A solid routing foundation: A tracking document of the tasks for building the routing foundation for the WordPress admin redesign.

— INSERT TABLE OF CONTENTS —

Highlights

Expanding the Core block library? It might be in the cards



There are always discussions of new blocks for Core happening. Just hop over to read the [current] 63 tickets labeled New Block in the Gutenberg repository.

In the past, there has seemed to be a hardline stance against anything that seemed too niche to include in Core, leaving third-party developers to fill the gaps. This has been a good system over the last two decades, but blocks are unlike other plugins. Theme authors rely on having a full toolkit to create unique designs without making users jump through hoops.

The good news is that the previous stance may be softening. Gutenberg lead architect Matías Ventura stated in the ticket New Block additions for the Block Library:

> *I think not having these blocks in the core library severely limits the expressiveness that theme builders (and users) can depend upon to build great designs and it can fracture the overall experience, forcing people to go for more fully equipped block libraries instead or desist altogether if they plan to release something broadly. One of the promises of blocks is that we can finally make themes interoperable without dulling down expression. A separate plugin for these blocks won't help much if a person needs to release a theme or a pattern in the directory, cannot rely on the presence of the block, or is excluded because it doesn't use core blocks.*
>
> *My opinion is that we should add a lot more blocks and revise a stance that I think harms adoption in this day and age.*

We're still a few steps away from completely opening the floodgates, but there are already several tickets and PRs linked in the GitHub issue that make for possible block candidates. I wouldn't doubt it if the Icon block proposal has the most support. At the very least, it has stirred up a lot of discussion in recent weeks.

What do you think? The more blocks in Core the merrier? Or leave it to plugins?

Proposing more theme.json settings control

In the last week, a new ticket landed in the Gutenberg repository that would allow deeper control over the block editor's user interface and experience from `theme.json`. The ticket also pushes for such control via post type registration.

This is another idea that's not a done deal yet, but it is an exciting proposal that could have some real momentum behind it. Currently, `theme.json` control of the user experience is a far call from being feature complete. But there is hope. Now that the UI for all the Core blocks has been standardized via the `ToolsPanel` component, contributors have taken the first step toward making this secondary proposal a reality.

The ticket outlines the next two steps that must be taken to make this level of control possible:

- Audit all block tool panels, ensuring that all inspector controls and toolbar options can be enabled/disabled via `theme.json`. It would require all Core blocks to adhere to those standards.
- Ensure this new mechanism can be used both globally and at the granular level.

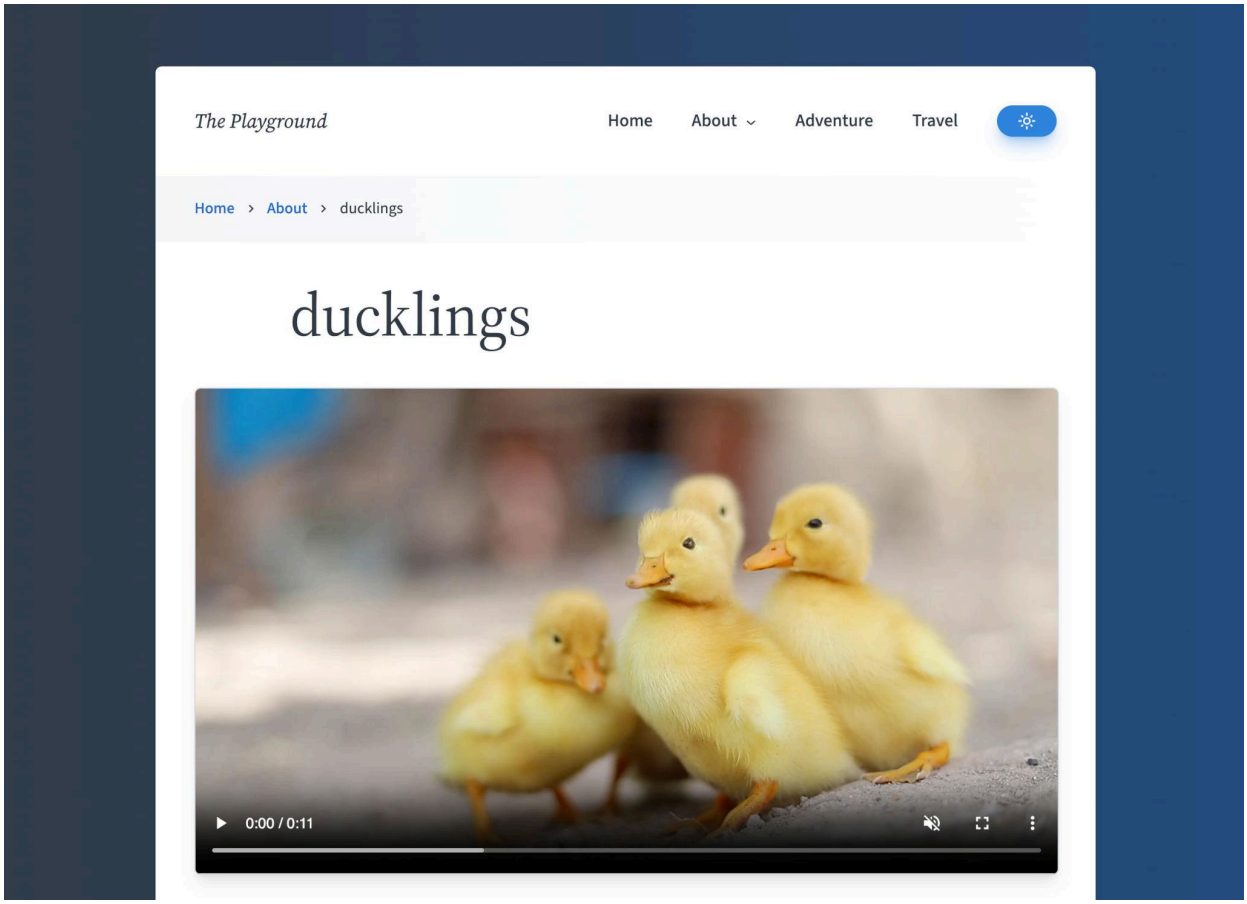
As a theme author, I'm excited about the prospect of enacting more control over the user experience. It would give me the freedom to implement some of the ideas I've had in these last few years but have lacked Core support for.

Customize Block Bindings support

I've rattled on enough about things that *might* happen at some point. Let me dive into a feature that is about as much of a sure thing as it gets.

With the latest WordPress trunk active, you can customize which blocks and their attributes can be bound. As of WordPress 6.8, this support has been limited to the Heading, Paragraph, and Button blocks. But now things are wide open with the new `block_bindings_supported_attributes_{$block_type}` filter hook.

It didn't take me long to tinker with this once the patch was committed in WordPress trunk. My first test was making the Video block's `src` attribute bindable, as shown in this screenshot (yep, that's really a custom binding under the hood):



Want to test your own bindings? Here's the filter I used for making the Video block's `src` attribute bindable:

```php
add_filter( 'block_bindings_supported_attributes_core/video', 'devblog_bindable_video_attrs' );

function devblog_bindable_video_attrs( array $attrs ): array
{
	return array_merge( $attrs, [ 'src' ] );
}
```

I'll let you take the rest from there. Be sure to check out the Block Bindings API tutorials right here on the Developer Blog if you need to freshen up your knowledge.

The only problem I have run into is not having enough time to try out all the ideas I've had for the last year or so since the API graced us all with its initial featureset. *I'm going to have so much fun with this!*

Plugins and tools

Keep a close eye on

In last month's edition of the monthly roundup, I noted a ticket that WordPress co-founder Matt Mullenweg opened: Phased releases and roll-outs of plugins. Over the last couple of weeks, contributors have steadily pushed new code to WordPress.org to make this possible. There's been no official word via the Make Plugins P2 yet, but it's worth keeping track of if you have plugins in the official directory.

A discussion on "composite" blocks is in the early stages on the Gutenberg repository. The idea is that there should be an easier way for blocks that contain specific nested blocks to surface those children for styling in the UI but be granular enough to style separately through `theme.json`. The Details block was noted as a suitable candidate for the concept (i.e., styling its `<summary>` element).

It's still too early to tell where the discussion leads, but it could potentially cater to both theme authors with more granular design control and users with a simpler interface.

AI building blocks

The Core AI team published its AI Building Blocks for WordPress post, which gave a quick overview of the four building blocks with dedicated posts detailing each one:

- PHP AI Client SDK
- Abilities API
- MCP Adapter
- AI Experiments Plugin

If you're interested in bringing these tools to Core, join the discussions in the #core-ai Slack channel.

WordPress Coding Standards 3.2 released

The latest iteration of the WPCS landed in the last month with several new additions and quite a few changes. Some of these include:

- Stricter meta usage checks
- Sniffs for PHP heredoc syntax
- Improved PHP 8.1+ callable support

- Expanded deprecated feature detection to WP 6.8.1,

And that's just to name a few items. The release also includes improved documentation and other optimizations to make your code better. Check out the release changelog for details on all the updates.

New date calendar components

Gutenberg 21.3 added the `DateCalendar` and `DateRangeCalendar` components, a needed improvement over the current `DatePicker` component, which doesn't support date ranges.

The new components create a unified approach to calendars that supports both single dates and date ranges. They also provide more flexible layout options.

Data Views progress

Contributors merged several notable tickets on Data Views over the past month:

- The View config has a new `perPageSizes` option for controlling the available sizes in the Items Per Page panel.
- Over two dozen new fields, controls, and operators were added in one giant ticket to help customize how Data Views and Data Forms work for a plethora of new use cases.

- Grouping functionality is now supported via the `groupByField` property.
- A new date field type was introduced with sorting and validation functions.

Playground

WordPress Playground has had a big month, especially when you consider that 39 pages of its documentation was translated into five languages: Gujarati, Spanish, Japanese, Filipino, and Portuguese.

There's also a new Playground CLI doc, an updated Web Instances doc, and new video resources.

General updates

Several general updates landed in the last month:

- PHP 8.3 is now the default version for playground.wordpress.net and Playground CLI.
- Added support to query monitor in the SQLite integration plugin.
- Added OPCache support.
- New multi-worker support for Node.js Asyncify builds.
- Playground now defaults to CORS proxy in embedded instances.
- Cleaned up and improved data flows for error logging.

CLI additions

A few nice additions to the Playground CLI include:

- Blueprints v2 added via the `--experimental-blueprints-v2-runner` flag.
- An `--internal-cookie-store` option if you need Playground to own the cookie session in your workflow.
- Stop printing the stack trace if the `--debug` flag isn't set and reject unrecognized options.

Xdebug support updates

Experimental Xdebug support had already shipped during the last monthly roundup, but contributors continued improving it by:

- Bridging DBGP sessions with CDP servers.
- Enabling the `--experimental–devtools` and `--xdebug` CLI options.

Themes

Poster image for Cover block videos

Over the last few years, several theme authors have implemented custom solutions for implementing the `poster` attribute for the Video block. Typically, these would fall back to using the attachment's featured image if

it existed. But most users wouldn't make the jump to uploading the image since the UI for it is removed from the block inspector controls.

Gutenberg 21.3 introduced a dedicated control for uploading a poster image for Video blocks.

The downside of the PR is that the image is only stored on the block level and not added to the media attachment itself. So you might still need that custom script as a fallback in cases where the user doesn't add their own poster.

Button element inherits typography

As of Gutenberg 21.3, the button element defined in the default `theme.json` inherits typographical styles from its parent. The full styles for the element now look like this:

```JSON
{
	"styles": {
		"elements": {
			"button": {
				"typography": {
					"fontSize": "inherit",
					"fontFamily": "inherit",
					"fontStyle": "inherit",
					"fontWeight": "inherit",
					"letterSpacing": "inherit",
```

```
                        "textTransform": "inherit",
                        "lineHeight": "inherit",
                        "textDecoration": "none"
                    }
                }
            }
        }
    }
```

This should not negatively affect most themes, but it's worth testing your design to ensure compatibility. If it does impact your design, it's likely that button elements were not consistent between the front end and editor in your themes in the past.

HTML tag selector for the Post Content block

Gutenberg 21.3 also added support for selecting a tag via the **Advanced → HTML Element** dropdown for the Post Content block, giving you more flexibility over the document markup. The block now supports four selectable elements:

- `<div>` (default)
- `<main>`
- `<section>`
- `<article>`

Welcome bug fixes

Contributors addressed several theme-related issues over the last couple of Gutenberg releases:

- Line breaks are now preserved in the caption for Image blocks.
- Core will decode HTML entities in the RSS block's feed titles before displaying them.
- Padding for the Archives block was removed to remain consistent with the List block.
- You can now style the button element in the File block via `theme.json` because the specificity of the block's CSS has been lowered.

Notable user changes

As usual, there are some changes that are more specific to users but are noteworthy in this development-related roundup because they directly impact your users.

Both the Gallery and Site Logo blocks now use the `ToolsPanel` component for their settings. This is a part of a larger effort to bring consistency to the block inspector across all Core blocks. It's also possible that this effort leads to making all block settings configurable via `theme.json`.

The Video block also has support for adding multiple tracks at once. This eliminates the need to repeat the process for each file.

Resources and news

There is one additional item from the Developer Blog in the past month. It's a tutorial that showcases how to style block themes without `theme.json`.

This section is a bit light this month. And while I think the pace of new articles will pick up pretty soon, the Developer Blog still needs you! Consider joining us and sharing your knowledge with the community. It can be a rewarding experience.

Props to @yournamehere, ...

Notes

Notes

Reviewers, this document section is not a part of the final post. Please ignore.

Resources:

- Keeping up with Gutenberg: Index 2022
- A Week in Core (nothing since March 2024)
    - Trac changesets (starts June 10 going back 61 days to April 10)
- Gutenberg plugin release posts
- Core editor chat (nothing since 2023), dev chat, performance team meeting summaries
- Dev Notes
- #design-share (nothing since Dec 2023)
- What's new in Gutenberg?
- https://github.com/WordPress/gutenberg/pulse

Key:

- ⭐ Highlights
- ❗ Deprecations and warnings
- 👍 Intro mention
- 🐛 Bugs
- ❓ Needs more investigation

Plugins & Tools

- Keep track of phased plugin rollouts: https://meta.trac.wordpress.org/ticket/8009
- Block Bindings: Add filter to set supported block attributes.
- DataViews: Introduce perPageSizes to control the available sizes of the items per page. (70604)
- More fields, controls and operators. (70567)
- Add groupByField support to grid layout. (70752)
- DataViews: Add date field type. (70657)
- Add DateCalendar and DateRangeCalendar components. (70578)
- Add ValidateControls components. (70620)
- wp-env: Add wp-cli configuration when creating environment. (70661)
- Notable tickets:
    - Explore a concept for composite blocks (a container block with a fixed set of inner blocks, with some UX tweaking) #70994
- JRF released v3.2 of the WordPress Coding Standards, PHP_CodeSniffer rules (sniffs) to enforce WordPress coding conventions. This release adds stricter meta usage checks, new sniffs for heredoc, better PHP 8.1+ callable support, expanded deprecated feature detection to WP 6.8.1, improved documentation, higher PHPCS requirements, and various optimizations. These help developers ensure cleaner, more up-to-date, and future-proof code.

WordPress Playground

At the Playground project, we implemented several performance improvements, including OPCache, and supported multiple Workers. On error handling, we included better messaging on PHP error logging and error reporting with the CLI, improving some flows by rejecting unrecognized options. At playground.wordpress.net, and the CLI, PHP 8.3 is the new default version.

About the Xdebug support, it is already available as an experimental feature with a lot of cool integrations, bridging DBGP sessions with CDP servers, and enabling Xdebug via CLI flags. Another important experimental feature is that Blueprints V2 is available via a flag.

Finishing with the documentation, more content has been translated into other languages, and now the documentation has content in Filipino, together with Gujarati, Spanish, Portuguese, and Japanese. A new page explaining how to use the new Playground CLI was added to the documentation.

General updates

- Added support to query monitor in the SQLite integration plugin #2415
- Added OPCache support #2400
- Makes PHP 8.3 the default version for playground.wordpress.net and Playground CLI. #2371

- Support multiple workers for NODEFS /wordpress mounts – Asyncify #2317
- Use CORS proxy in embedded Playgrounds #2369
- Improve error logging #2357

Adding experimental support to Xdebug

- Bridge DBGP session with CDP server with Xdebug #2402
- Introduces an --experimental-devtools option in the Playground CLI to start the Xdebug-to-CDP bridge process. #2411.
- Add --xdebug option in php-wasm CLI and wp-playground CLI #2346.

Playground CLI

- Add --internal-cookie-store option for Playground CLI #2323: Some workflows need Playground to own cookie persistence, e.g., VS Code webview does not store cookies, so any Blueprint with autologin enabled results in an infinite redirection loop.
- Improve error reporting #2401: Stop printing the stack trace if the --debug flag isn't set and reject unrecognized options.
- Expose Blueprints v2 runner in Playground CLI: Adds Blueprints v2 support to Playground CLI via the --experimental-blueprints-v2-runner flag. #2394

Documentation

- Added a new page about the playground CLI #2337, updated Playground Web instance page #2365, and new video content

related to Playground in Polish, Spanish, Japanese, and English was added to the resources page. #2348.

- Translations: 39 new pages of the documentation have been translated into four languages: Gujarati, Spanish, Japanese, Filipino, and Portuguese. #2448, #2343, #2358, #2391, #2340, #2352, #2386, #2434, #2438, #2431, #2351, #2450, and another 15 translated pages were updated.

Themes

- Navigation Link: Add "Open in new tab" toggle to navigation block sidebar. (67262)
- Image Block: Preserve line breaks in media caption. (70476)
- RSS Block: Decode HTML entities in feed titles before display. (70491)
- Cover: Enable support for adding posters over video. (70816)
- Navigation Submenu Link: Add Open in new tab toggle to navigation block sidebar. (70687)
- Post Content Block: Add tagName selector. (70698)
- Archives: Update padding for archives block to remain consistent with list. (69008)
- Fix: Preserve hasParallax when switching from video to image in Cover block. (70703)
- Core/File: Reduce specificity of pseudo-selector styles for better override support via theme.json. (70358)

- Make Button element inherit all typography styles on the frontend. (70676)
- Notable tickets:
    - New Block additions for the Block Library #71026
    - Controlling the editor: Theme.json settings to disable/enable all block controls. #71013

User

- Gallery: Refactor "Settings" panel of Gallery block to use ToolsPanel instead of PanelBody. (67904)
- Site Logo: Refactor the media panel to use ToolsPanel. (70599)
- Video: Enable support for adding multiple tracks. (70689)
-

Resources

- General:
    - Roadmap to 6.9
    - Admin Materials and Surfaces,
    - Abilities & Workflows Overview,
    - Admin Redesign: A solid routing foundation,
    - Core AI team published their AI Building Blocks for WordPress post, briefly mention the four projects, their GitHub repositories, and that interested parties can join the #core-ai Slack channel

- Developer Blog:
    -