@wordpress/server-side-render

- Installation
- API
  - default
  - ServerSideRender
  - useServerSideRender
- Output
- API Endpoint
- Contributing to this package

ServerSideRender is a component used for server-side rendering a preview of dynamic blocks to display in the editor. Server-side rendering in a block's `edit` function should be limited to blocks that are heavily dependent on existing PHP rendering logic that is heavily intertwined with data, particularly when there are no endpoints available.

ServerSideRender may also be used when a legacy block is provided as a backward compatibility measure, rather than needing to re-write the deprecated code that the block may depend on.

ServerSideRender should be regarded as a fallback or legacy mechanism, it is not appropriate for developing new features against.

New blocks should be built in conjunction with any necessary REST API endpoints, so that JavaScript can be used for rendering client-side in the `edit` function. This gives the best user experience, instead of relying on using the PHP `render_callback`. The logic necessary for rendering should be included in the endpoint, so that both the client-side JavaScript and server-side PHP logic should require a minimal amount of differences.

> This package is meant to be used only with WordPress core. Feel free to use it in your own project but please keep in mind that it might never get fully documented.

Installation

Install the module

```
npm install @wordpress/server-side-render --save
```

*This package assumes that your code will run in an **ES2015+** environment. If you're using an environment that has limited or no support for such language features and APIs, you should include the polyfill shipped in `@wordpress/babel-preset-default` in your code.*

API

default

> **Deprecated** Use `ServerSideRender` non-default export instead.

A compatibility layer for the `ServerSideRender` component when used with `wp` global namespace.

Usage

```
import ServerSideRender from '@wordpress/server-side-render';
```

ServerSideRender

A component that renders server-side content for blocks.

Note: URL query will include the current post ID when applicable. This is useful for blocks that depend on the context of the current post for rendering.

Usage

```
import { ServerSideRender } from '@wordpress/server-side-render';
// Legacy import for WordPress 6.8 and earlier
// import { default as ServerSideRender } from '@wordpress/server-side-render';
```

```
function Example() {
    return (
        <ServerSideRender
            block="core/archives"
            attributes={ { showPostCounts: true } }
            urlQueryArgs={ { customArg: 'value' } }
            className="custom-class"
        />
    );
}
```

Parameters

- *props* `ServerSideRenderWithPostIdProps`: Component props.
- *props.urlQueryArgs* `ServerSideRenderWithPostIdProps[ 'urlQueryArgs' ]`: Additional query arguments to append to the request URL.

Returns

- The rendered server-side content.

useServerSideRender

A hook for server-side rendering a preview of dynamic blocks to display in the editor.

Handles fetching server-rendered previews for blocks, managing loading states, and automatically debouncing requests to prevent excessive API calls. It supports both GET and POST requests, with POST requests used for larger attribute payloads.

Usage

Basic usage:

```
import { RawHTML } from '@wordpress/element';
import { useServerSideRender } from '@wordpress/server-side-render';

function MyServerSideRender( { attributes, block } ) {
    const { content, status, error } = useServerSideRender( {
        attributes,
        block,
    } );
```

```
    if ( status === 'loading' ) {
        return <div>Loading...</div>;
    }

    if ( status === 'error' ) {
        return <div>Error: { error }</div>;
    }

    return <RawHTML>{ content }</RawHTML>;
}
```

Parameters

- *args* `UseServerSideRenderArgs`: The hook configuration object.

Returns

- `ServerSideRenderResponse`: The server-side render response object.

Output

Output uses the block's `render_callback` function, set when defining the block.

API Endpoint

The API endpoint for getting the output for ServerSideRender is `/wp/v2/block-renderer/:block`. It will use the block's `render_callback` method.

If you pass `attributes` to `ServerSideRender`, the block must also be registered and have its attributes defined in PHP.

```
register_block_type(
    'core/archives',
    array(
        'api_version' => 3,
        'attributes'      => array(
            'showPostCounts'    => array(
                'type'      => 'boolean',
```

```
                'default'   => false,
            ),
            'displayAsDropdown' => array(
                'type'      => 'boolean',
                'default'   => false,
            ),
        ),
        'render_callback' => 'render_block_core_archives',
    )
);
```

Contributing to this package

This is an individual package that’s part of the Gutenberg project. The project is organized as a monorepo. It’s made up of multiple self-contained software packages, each with a specific purpose.

The packages in this monorepo are published to npm and used by WordPress as well as other software projects.

To find out more about contributing to this package or Gutenberg as a whole, please read the project’s main contributor guide.

□□□ #2
□ □□□ □□□ 2026-10-31 18:20:23 UTC
□ □□□ □□□ 2026-10-31 18:20:25 UTC