区块包装器

目录

Editor markup

保存标记

动态渲染标记

区块编辑器中的每个区块都包含在一个 HTML
包装器内，该包装器必须具备特定属性才能在编辑器和前端正常运行。作为开发者，我们可以直接操作此标记
，WordPress 提供了诸如 useBlockProps() 等工具来修改添加到区块包装器的属性。

在使用自定义样式或 区块支持 等功能时，确保区块包装器具有正确的属性尤为重要。

WordPress 中的区块可以通过三种不同类型的标记来定义，每种标记都扮演着独特的角色：

编辑器标记： 这是区块在区块编辑器中的视觉呈现。当区块通过 registerBlockType
在客户端注册时，使用 Edit React 组件定义。
保存标记： 此标记是在保存区块内容时存入数据库的内容。它通过 save
函数指定，该函数同样在区块注册时提供给 registerBlockType
。如果区块不使用动态渲染，则此保存的标记将用于前端显示。
动态渲染标记： 当区块内容需要动态生成时，此标记发挥作用。它在服务器端定义，可以通过
register_block_type 中的 render_callback 函数，或 block.json 中指定的 render.php
文件来定义。如果存在，此标记将覆盖任何已保存的标记，并用于区块的前端显示。

对于 Edit 组件和 save 函数 ，重要的是使用标准 DOM 元素（如 <div>）或将所有附加属性传递给原生
DOM 元素的 React 组件作为包装器元素。使用 React Fragments (<Fragment>) 或 <ServerSideRender> 组件作为这些包装器是无效的。

Editor markup
The useBlockProps() hook, provided by the @wordpress/block-editor package, is used to define the outer markup of a block in the Edit component.

This hook simplifies several tasks, including:

Assigning a unique id to the block's HTML structure.

- Adding various accessibility and `data-` attributes for enhanced functionality and information.
- Incorporating classes and inline styles that reflect the block's custom settings. By default, this includes:
  - The `wp-block` class for general block styling.
  - A block-specific class that combines the block's namespace and name, ensuring unique and targeted styling capabilities.

In the following example, the Editor markup of the block is defined in the `Edit` component using the `useBlockProps()` hook.

```
const Edit = () => <p { ...useBlockProps() }>Hello World - Block Editor</p>;

registerBlockType( ..., {
edit: Edit
} );
```

See the full block example of the code above.

The markup of the block in the Block Editor could look like this, where the classes and attributes are applied automatically:

```
<p
    tabindex="0"
    id="block-4462939a-b918-44bb-9b7c-35a0db5ab8fe"
    role="document"
    aria-label="Block: Minimal Gutenberg Block ca6eda"
    data-block="4462939a-b918-44bb-9b7c-35a0db5ab8fe"
    data-type="block-development-examples/minimal-block-ca6eda"
    data-title="Minimal Gutenberg Block ca6eda"
    class="
        block-editor-block-list__block
        wp-block
        is-selected
        wp-block-block-development-examples-minimal-block-ca6eda
    "
>Hello World - Block Editor</p>
```

In a block's `Edit` component, use the `useBlockProps()` hook to include additional classes and attributes by passing them as arguments. (See example)

When you enable features using the `supports` property, any corresponding classes or attributes are included in the object returned by `useBlockProps` automatically.

保存标记

将标记保存到数据库时，务必将 `useBlockProps.save()` 返回的属性添加到区块的包装元素上。`useBlockProps.save()` 不仅能确保区块类名正确渲染，还能包含由区块支持 API 注入的任何 HTML 属性。

请看以下在客户端注册区块的代码。注意它如何定义编辑区块时和将区块保存到数据库时应使用的标记。

```
const Edit = () => <p { ...useBlockProps() }>Hello World - Block Editor</p>;
const save = () => <p { ...useBlockProps.save() }>Hello World - Frontend</p>;

registerBlockType( ..., {
	edit: Edit,
	save,
} );
```

查看 上方代码 的 完整区块示例 。

前端区块的标记可能如下所示，其中类名会自动应用：

```
<p class="wp-block-block-development-examples-minimal-block-ca6eda">Hello World – Frontend</p>
```

如果要在区块的 `save` 函数中添加任何额外的类或属性，应将其作为 `useBlockProps.save()` 的参数传递。（参见 示例）

当为任何功能添加 `supports` 时，相应的类会被添加到 `useBlockProps.save()` 钩子返回的对象中。在下面的示例中，段落区块已添加了文本和背景颜色类。

```
<p class="
    wp-block-block-development-examples-block-supports-6aa4dd
    has-accent-4-color
    has-contrast-background-color
    has-text-color
    has-background
">Hello World</p>
```

生成此 HTML 的 示例区块 可在 区块开发示例 仓库中找到。

动态渲染标记

在动态区块中，前端标记是通过服务器端渲染的，你可以使用 `get_block_wrapper_attributes()` 函数来输出必要的类和属性，就像在 `save` 函数中使用 `useBlockProps.save()` 一样。（参见 示例）

```
<p <?php echo get_block_wrapper_attributes(); ?>>
	<?php esc_html_e( 'Block with Dynamic Rendering – hello!!!', 'block-development-examples' );
	?>
</p>
```

版本号 #2
由 壳壳虫 创建于 2026-03-26 17:08:43 UTC
由 壳壳虫 更新于 2026-03-26 17:08:43 UTC