□□□□□

□□

- □□□
  - canInsertBlockInWidgetArea
  - getEditedWidgetAreas
  - getIsWidgetAreaOpen
  - getParentWidgetAreaBlock
  - getReferenceWidgetBlocks
  - getWidget
  - getWidgetAreaForWidgetId
  - getWidgetAreas
  - getWidgets
  - isInserterOpened
  - isListViewOpened
  - isSavingWidgetAreas
  - isWidgetSavingLocked
- □□
  - closeGeneralSidebar
  - lockWidgetSaving
  - moveBlockToWidgetArea
  - persistStubPost
  - saveEditedWidgetAreas
  - saveWidgetArea
  - saveWidgetAreas
  - setIsInserterOpened
  - setIsListViewOpened
  - setIsWidgetAreaOpen
  - setWidgetAreasOpenState
  - setWidgetIdForClientId
  - unlockWidgetSaving

命名空间： core/edit-widgets。

选择器

canInsertBlockInWidgetArea
如果可以将区块插入到小部件区域，则返回 true。
参数
state Array: 小部件区域的打开状态。
blockName string: 要插入的区块名称。
返回值
boolean: 如果区块可以插入到小部件区域，则为 true。

getEditedWidgetAreas
返回所有已编辑的小部件区域实体记录。
返回值
Object[]: 已编辑的小部件区域实体记录列表。

getIsWidgetAreaOpen
获取小部件区域是否已打开。
参数
state Array: 小部件区域的打开状态。
clientId string: 小部件区域的 clientId。
返回值
boolean: 如果小部件区域已打开，则为 true。

getParentWidgetAreaBlock

给定一个子客户端 ID，返回父级小部件区域块。
参数
clientId string: 小部件区域中某个块的客户端 ID。
返回值
WPBlock: 小部件区域块。

getReferenceWidgetBlocks
返回所有表示引用小部件的块。
参数
referenceWidgetName string：可选。如果提供，则仅返回具有此名称的引用小部件。
返回值
Array：表示引用小部件的所有块的列表

getWidget
返回指定小部件 ID 的 API 小部件数据。
参数
id number: 小部件 ID。
返回
Object: 指定小部件 ID 的 API 小部件数据。

getWidgetAreaForWidgetId
返回包含给定 widgetId 所标识区块的 widgetArea。
参数
widgetId string: 小部件的 ID。
返回

Object: 包含小部件区域的对象。

getWidgetAreas
返回所有 API 小部件区域。
返回
Object[]: API 小部件区域列表。

getWidgets
返回所有 API 小部件。
返回
Object[]: API 小部件列表。

isInserterOpened
如果插入器已打开，则返回 true。
参数
state Object: 全局应用程序状态。
返回值
boolean: 插入器是否已打开。

isListViewOpened
如果列表视图已打开，则返回 true。
参数
state Object: 全局应用程序状态。
返回值
boolean: 列表视图是否已打开。

isSavingWidgetAreas

□□□□□□□□□□□□□□□□□□□□□□ true□

□□□

- `boolean`: □□□□□□□□□□□□□□□□□□□□ true□□□□ false□

isWidgetSavingLocked

□□□□□□□□□□□□□

□□

```
import { __ } from '@wordpress/i18n';
import { store as widgetStore } from '@wordpress/edit-widgets';
import { useSelect } from '@wordpress/data';

const ExampleComponent = () => {
□const isSavingLocked = useSelect(
□□( select ) => select( widgetStore ).isWidgetSavingLocked(),
□□[]
□);

□return isSavingLocked ? (
□□<p>{ __( 'Widget saving is locked' ) }</p>
□) : (
□□<p>{ __( 'Widget saving is not locked' ) }</p>
□);
};
```

□□

- *state* `Object`: □□□□□□□

□□□

- `boolean`: □□□□□□

操作

closeGeneralSidebar

返回一个操作对象，表示用户关闭了侧边栏。

返回值

- `Object`：操作创建器。

lockWidgetSaving

返回一个用于表示小部件保存已锁定的操作对象。

用法

```
import { store as widgetStore } from '@wordpress/edit-widgets';
import { useDispatch } from '@wordpress/data';

const ExampleComponent = () => {
	const { lockWidgetSaving } = useDispatch( widgetStore );
	return (
		<Button onClick={ () => lockWidgetSaving( 'lockName' ) }>
			{ __( 'Lock Widget Saving' ) }
		</Button>
	);
};
```

参数

- *lockName* `string`：锁名称。

返回值

- `Object`：操作对象

moveBlockToWidgetArea

用于在挂件区域之间移动区块的操作

参数
clientId string: 要移动区块的客户端 ID。
widgetAreaId string: 要将区块移动到的挂件区域 ID。

persistStubPost
将具有给定 ID 的存根文章持久化到核心数据存储中。该文章仅存在于内存中，不应通过 API 保存。参数
id string: 文章 ID。
blocks Array: 文章应包含的区块。
返回值
Object: 文章对象。

saveEditedWidgetAreas
将已编辑的小部件区域中的所有区块转换为小部件，并提交批量请求一次性保存所有内容。
无论成功或失败，都会创建一个 Snackbar 通知。
返回值
Function: 一个动作创建器。

saveWidgetArea
将指定 ID 的小工具区域中的所有区块转换为小工具，并提交批量请求一次性保存所有内容。
参数
widgetAreaId string: 要处理的小工具区域的 ID。
返回
Function: 一个动作创建器。

saveWidgetAreas

将指定小部件区域中的所有区块转换为小部件，并提交批量请求以一次性保存所有内容。

参数

- *widgetAreas* `Object[]`：要保存的小部件区域。

返回值

- `Function`：一个动作创建器。

setIsInserterOpened

返回一个用于打开 /关闭插入器的操作对象。

参数

- *value* `boolean|Object`：插入器应打开 (true) 还是关闭 (false)。要指定插入点，请使用对象。
- *value.rootClientId* `string`：要插入的根客户端 ID。
- *value.insertionIndex* `number`：要插入的索引。

返回值

- `Object`：操作对象。

setIsListViewOpened

返回一个用于打开 /关闭列表视图的操作对象。

参数

- *isOpen* `boolean`：一个布尔值，表示是否应打开列表视图。

返回值

- `Object`：操作对象。

setIsWidgetAreaOpen

设置小部件区域的打开状态。

参数

- *clientId* `string`：小部件区域的 clientId。
- *isOpen* `boolean`：小部件区域是否应被打开。

□□□

- `Object`: □□□

setWidgetAreasOpenState

□□□□□□□□□□□□□□□

□□

- *widgetAreasOpenState* `Object` □□□□□□□□□□□□□□□

□□□

- `Object`□□□□

setWidgetIdForClientId

□□□□□ widgetId □□□ clientId□

□□

- *clientId* `number`: □□□ ID□
- *widgetId* `number`: □□ ID□

□□□

- `Object`: □□□□□

unlockWidgetSaving

□□□□□□□□□□□□□□□□□□□□□□

□□

```
import { store as widgetStore } from '@wordpress/edit-widgets';
import { useDispatch } from '@wordpress/data';

const ExampleComponent = () => {
□const { unlockWidgetSaving } = useDispatch( widgetStore );
□return (
□□<Button onClick={ () => unlockWidgetSaving( 'lockName' ) }>
```

```
□□□{ __( 'Unlock Widget Saving' ) }
□□</Button>
□);
};
```

参数

- *lockName* `string`：锁名称。

返回值

- `Object`：操作对象

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