□□□

□□

- □□□ □□
  - getFormatType
  - getFormatTypeForBareElement
  - getFormatTypeForClassName
  - getFormatTypes
- □□ □□

□□□□□ `core/rich-text` □

□□□

getFormatType

□□□□□□□□□□□

□□

```
import { __, sprintf } from '@wordpress/i18n';
import { store as richTextStore } from '@wordpress/rich-text';
import { useSelect } from '@wordpress/data';

const ExampleComponent = () => {
   const { getFormatType } = useSelect(
       ( select ) => select( richTextStore ),
       []
   );

   const boldFormat = getFormatType( 'core/bold' );

   return boldFormat ? (
```

```
        <ul>
            { Object.entries( boldFormat )?.map( ( [ key, value ] ) => (
                <li>
                    { key } : { value }
                </li>
            ) ) }
        </ul>
    ) : (
        __( 'Not Found' )
    ;
};
```

□□

- *state* `Object`: □□□□□
- *name* `string`: □□□□□□□

□□□

- `?Object`: □□□□□

getFormatTypeForBareElement

Gets the format type, if any, that can handle a bare element (without a data-format-type attribute), given the tag name of this element.

Usage

```
import { __, sprintf } from '@wordpress/i18n';
import { store as richTextStore } from '@wordpress/rich-text';
import { useSelect } from '@wordpress/data';

const ExampleComponent = () => {
□const { getFormatTypeForBareElement } = useSelect(
□□( select ) => select( richTextStore ),
□□[]
□);

□const format = getFormatTypeForBareElement( 'strong' );

□return format && <p>{ sprintf( __( 'Format name: %s' ), format.name ) }</p>;
```

```
};
```

Parameters

- *state* `Object`: Data state.
- *bareElementTagName* `string`: The tag name of the element to find a format type for.

Returns

- `?Object`: Format type.

getFormatTypeForClassName

Gets the format type, if any, that can handle an element, given its classes.

Usage

```
import { __, sprintf } from '@wordpress/i18n';
import { store as richTextStore } from '@wordpress/rich-text';
import { useSelect } from '@wordpress/data';

const ExampleComponent = () => {
	const { getFormatTypeForClassName } = useSelect(
		( select ) => select( richTextStore ),
		[]
	);

	const format = getFormatTypeForClassName( 'has-inline-color' );

	return format && <p>{ sprintf( __( 'Format name: %s' ), format.name ) }</p>;
};
```

Parameters

- *state* `Object`: Data state.
- *elementClassName* `string`: The classes of the element to find a format type for.

Returns

- `?Object`: Format type.

getFormatTypes

□□□□□□□□□□□□

□□

```
import { __, sprintf } from '@wordpress/i18n';
import { store as richTextStore } from '@wordpress/rich-text';
import { useSelect } from '@wordpress/data';

const ExampleComponent = () => {
□const { getFormatTypes } = useSelect(
□□( select ) => select( richTextStore ),
□□[]
□);

□const availableFormats = getFormatTypes();

□return availableFormats ? (
□□<ul>
□□□{ availableFormats?.map( ( format ) => (
□□□□<li>{ format.name }</li>
□□□) ) }
□□</ul>
□) : (
□□__(□□□□□' )
□);
};
```

□□

- *state* `Object`: □□□□□

□□□

- `Array`: □□□□□

□□

□□□□□□□□

□□□ #2
□ □□□ □□□ 2026-03-26 17:08:43 UTC
□ □□□ □□□ 2026-03-26 17:08:43 UTC