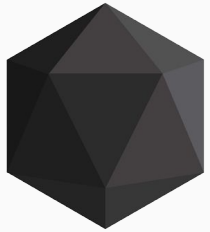
Mes premiers blocs WordPress

Présentation

Arnaud Banvillet - Développeur Web chez Matière Noire



Attention ce meetup n’est pas un meetup sur :

- (le cyclimse)
- La prise en main de l’interface
- Gutenberg côté thème

Etat des lieux du projet Gutenberg

WordPress 5.0 en béta (19 novembre 2018)

Gutenberg devient le **block editor** en comparaison au **classic editor**

Techniquement :

- modif du bootstrap php pour la page d'édition
- Ajout de dépendances au `package.json` du core

Le plugin continue d'exister sur Github pour la phase 2

Création de notre plugin

- wp-config.php
  - `define( 'SCRIPT_DEBUG', true );`

- wp-content/plugins/gut-meetup/
  - .babelrc
  - gut-meetup.php
  - package.json

Autre possibilités

- create guten block : https://github.com/ahmadawais/create-guten-block
- wp scaffold block (création des fichiers de block ES5 et pas de babel)

Ne pas créer de nouveau block: Les styles de block

```php
// gut-demo.php
add_action('enqueue_block_editor_assets', 'gut_demo_enqueue_block_editor_assets');
function gut_demo_enqueue_block_editor_assets(){
	wp_enqueue_script(‘demo-style', plugins_url( 'dist/index.js', __FILE__ ), array( 'wp-blocks'));
}
```

```js
// src/index.js
wp.blocks.registerBlockStyle( 'core/quote', {
	name: 'fancy-quote',
	label: 'Fancy Quote'
} )
```

Création de blocs

```php
// gut-demo.php

add_action( 'init', 'gut_demo_register_block' )

function gut_demo_register_block() {
	wp_register_script( 'gut-demo-block',
		plugins_url( 'dist/index.js', __FILE__ ),
		array( 'wp-element' )
	);

	register_block_type( 'gut-demo-block/hello-world', array(
		'editor_script'		=> 'gut-demo-block',
		// 'script'			=> Block type front end script handle
		// 'editor_style'	=> Block type editor style handle
		// 'style'			=> Block type front end style handle
		// 'render_callback'	=> Block type render callback
	) );
}
```

```
// src/index.js

const { createElement } = wp.element
const { registerBlockType } = wp.blocks

registerBlockType("gut-demo-block/hello-world", {
        title: "Hello World",
        category: "common",
        description: "Just another Hello World block",
        icon: "admin-site",

        edit: function() { return <p>Hello Editor</p> },

        save: function() { return <p>Hello Frontend</p> }
})
```

registerBlockType(args)

- Title
- Description
- Category [common | formatting | layout | widgets | embed] (filter “block_categories”)
- Icon (Dashicon, SVG)
- Keywords
- Attributes
- Transforms
- Parent
- Supports [align,alignWide,anchor,customClassName,className,html,inserter,multiple]

Suite

- Deprecated
- Styles

```
styles: [
	{ name: 'regular', label: __( 'Regular' ), isDefault: true },
	{ name: 'stripes', label: __( 'Stripes' ) },
]
```

Edit

Component, class ou fonction

```
const { Component } = wp.element
```

On récupère en props

```
{ attributes, setAttributes, className, isSelected }
```

Save

- HTML sauvegarder en base
- Parser pour les attributs
- Si changement, utiliser : `deprecated` dans `registerBlockType`

Attributs et fonctionnement

Ils seront accessibles dans les props de votre composant

Récupéré en parsant le HTML sauvegardé

Plusieurs sources possible

Sources des attributs

Sources :

- attribute
- text : inner text
- html : inner HTML
- query : array of values from markup, ex
- meta : attention doit être exposé dans l’API et correctement typé

```
attributes: {
	id: {
		type: 'number',
	},
	fileName: {
		source: 'html',
		selector: 'a:not([download])',
	},
	// Differs to the href when the block is configured to link to the attachment page
	textLinkHref: {
		type: 'string',
		source: 'attribute',
		selector: 'a:not([download])',
		attribute: 'href',
	},
	showDownloadButton: {
		type: 'boolean',
		default: true,
	},
},
```

@wordpress/components

Éléments d’interface à utiliser

Pas vraiment de doc, chaque composant à un readme :
https://github.com/WordPress/gutenberg/tree/master/packages/components/src

- PanelBody, Toolbar
- TextControl, Button, SelectControl, CheckboxControl, RadioControl, RangeControle
- ColorPicker, Spinner, DateTimePicker, ContrastChecker

Deux zones pour les paramétrages

```
edit: function() {
	return (
		<Fragment>
			<BlockControls>
				<!-- ici block controls -->
			</BlockControls>
			<InspectorControls>
				<PanelBody title={ 'Map Settings' }></PanelBody>
			</InspectorControls>
			<ServerSideRender block="gut-meetup/dynamic-block"  />
		</Fragment>
		);
}
```

Deux autre types de blocs

- Inner blocks
- Dynamic blocks

Inner blocks

```
const { createElement } = wp.element
const { registerBlockType } = wp.blocks
const { InnerBlocks } = wp.editor
registerBlockType("gut-meetup/inner-blocks", {
        //…
        edit({className}) {
                return (
                        <div className={className}>
                                <InnerBlocks template={[
                                        ["core/paragraph", {placeholder: "First content..."}],
                                        ["core/image"],
                                        ["core/paragraph", {placeholder: "Second Content"}]
                                    ]}
                                            templateLock="all"
                                />
                        </div>
                        )
         },
        save() {
                return (<div><InnerBlocks.Content/></div>)
                 }
        }
)
```

Dynamic blocks

```php
// gut-demo.php

add_action( 'init', 'gut_demo_register_block' );

function gut_demo_register_block() {
	wp_register_script( 'gut-demo-block',
		plugins_url( 'dist/index.js', __FILE__ ),
		array( 'wp-element' )
	);

	register_block_type( 'gut-demo/dynamic-block', array(
		'editor_script'		=> 'gut-demo-block',
		// 'script'			=> Block type front end script handle
		// 'editor_style'		=> Block type editor style handle
		// 'style'			=> Block type front end style handle
		 'render_callback'	=> 'gut_demo_render_block' // return pas echo
	) );
}
```

```
// src/index.js
const { createElement } = wp.element;
const { registerBlockType } = wp.blocks;
const { ServerSideRender } = wp.components;

registerBlockType( 'gut-demo/dynamic-block', {
	title: 'Dynamic Block',

	edit: function( props ) {
		return (
			<ServerSideRender
				block="gut-demo/dynamic-block"
				attributes={ props.attributes }
			/>  );
	},

	 save() {
		// Rendering in PHP
		return null;
	 },
} );
```

Avez vous des questions ?