

Creare Blocchi per Gutenberg. Da Dove Iniziare?



Carlo Daniele

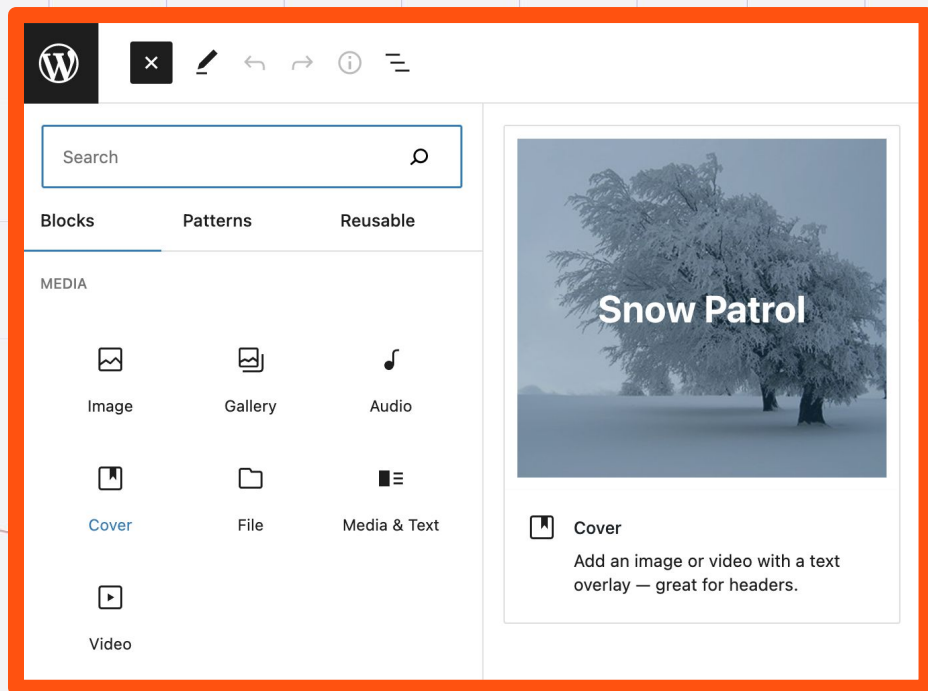


Business Development Manager Italy @Kinsta

Cosa è Gutenberg

Gutenberg è una **Single Page Application** (SPA) basata su **React** che permette agli utenti di WordPress di creare, modificare e cancellare contenuti in WordPress.

Gutenberg non è un normale editor WYSIWYG, ma uno strumento che ridefinisce l'intera esperienza di editing in WordPress.

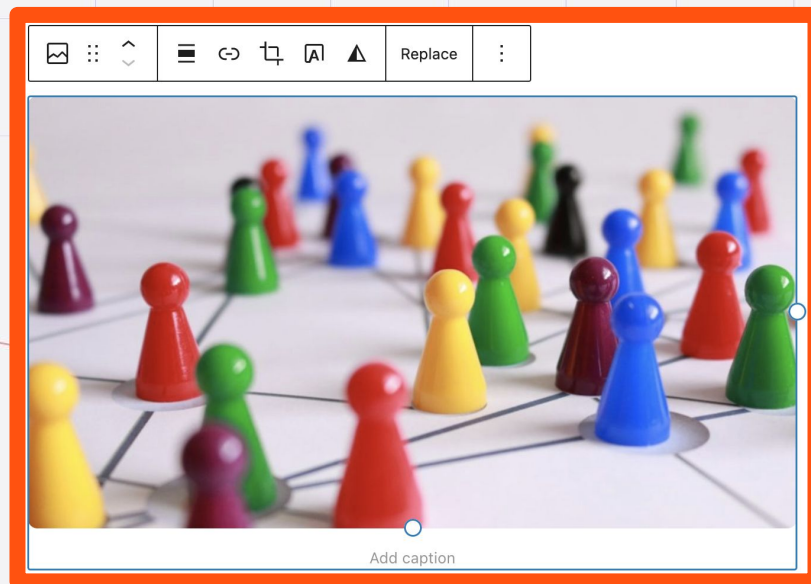


I Blocchi di Gutenberg

In Gutenberg, il contenuto è diviso in **blocchi**, che sono i “mattoni” che gli utenti possono utilizzare per creare articoli e pagine o i loro interi siti web.

“Blocco” è il termine astratto utilizzato per descrivere **unità di markup** che, composte insieme, formano il contenuto o il layout di una pagina web.

Titoli, paragrafi, colonne, immagini, fino ai controlli della barra degli strumenti dei blocchi, sono **componenti React**.



I Componenti React

I componenti sono **blocchi di codice indipendenti e riutilizzabili**.

Hanno lo stesso scopo delle funzioni JavaScript, ma lavorano in modo isolato e **restituiscono codice HTML**.

```
/**
 * The edit function describes the structure of your block in the context of the
 * editor. This represents what the editor will render when the block is used.
 *
 * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-edit-save/#edit
 *
 * @return {WPElement} Element to render.
 */
export default function Edit() {
  return (
    <p { ...useBlockProps() }>
      { __( 'My block  hello from the editor!', 'my-first-block' ) }
    </p>
  );
}
```

Il Codice Generato da un Blocco

I post creati con Gutenberg sono sempre memorizzati nella tabella `wp_posts`.

Ma in un post creato con Gutenberg, nella tabella vengono memorizzate altre informazioni che non sono presenti nei post creati tramite l'editor classico.

Queste informazioni sono racchiuse nei **delimitatori di blocco**.

```
<!-- wp:image {"id":1531,"sizeSlug":"large","linkDestination":"none"} -->
<figure class="wp-block-image size-large"></figure>
<!-- /wp:image -->
```

Configurazione dell'Ambiente di Sviluppo

1. Installazione di **Node.js** e **npm**
2. Installazione dell'**Ambiente di Sviluppo**
3. Installazione del **Plugin del Blocco**

node

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Node.js* is an open-source, cross-platform JavaScript runtime environment.

Download for macOS (x64)

16.18.0 LTS
Recommended For Most Users

18.11.0 Current
Latest Features

Other Downloads | Changelog | API Docs | Other Downloads | Changelog | API Docs

For information about supported releases, see the [release schedule](#).

Copyright OpenJS Foundation and Node.js contributors. All rights reserved. The OpenJS Foundation has registered trademarks and uses trademarks. For a list of trademarks of the OpenJS Foundation, please see our [Trademark Policy](#) and [Trademark List](#). Trademarks and logos not indicated on the list of OpenJS Foundation trademarks are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by them.

The OpenJS Foundation | [Terms of Use](#) | [Privacy Policy](#) | [Bylaws](#) | [Code of Conduct](#) | [Trademark Policy](#) | [Trademark List](#) | [Cookie Policy](#) | [Edit On GitHub](#)

npm Docs

Search npm Docs

npmjs.com Status Support

npm Docs

Documentation for the npm registry interface

- About npm
- Getting started
- Packages and modules
- Integrations
- Organizations
- Policies
- Threats and Mitigations
- npm CLI

DevKinsta

Features Documentation FAQ Forum

Download

Your Free Local WordPress Development Suite

Design, develop, and deploy WordPress sites from the comfort of your local machine. DevKinsta is free forever, and available for macOS, Windows, and Ubuntu. Used by 28,700+ developers, web designers, and freelancers.

Download DevKinsta

@wordpress/create-block

4.3.0 • Public • Published 10 days ago

Readme Explore **BETA** 13 Dependencies 2 Dependents 70 Versions

Create Block

Create Block is an officially supported tool for scaffolding WordPress plugins with blocks. It generates PHP, JS, CSS code, and everything you need to start the project. It integrates a modern build setup with no configuration.

Install

```
npm i @wordpress/create-block
```

Repository

github.com/WordPress/gutenberg

Homepage

github.com/WordPress/gutenberg...

Weekly Downloads

NodeJS & npm

Node.js è un **runtime JavaScript** costruito sul motore JavaScript V8 di Chrome.

npm, comunemente conosciuto come il **gestore di pacchetti** di Node, è considerato “il più grande registry di software del mondo.”



node

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Node.js® is an open-source, cross-platform JavaScript runtime environment.

Download for macOS (x64)

16.18.0 LTS
Recommended For Most Users

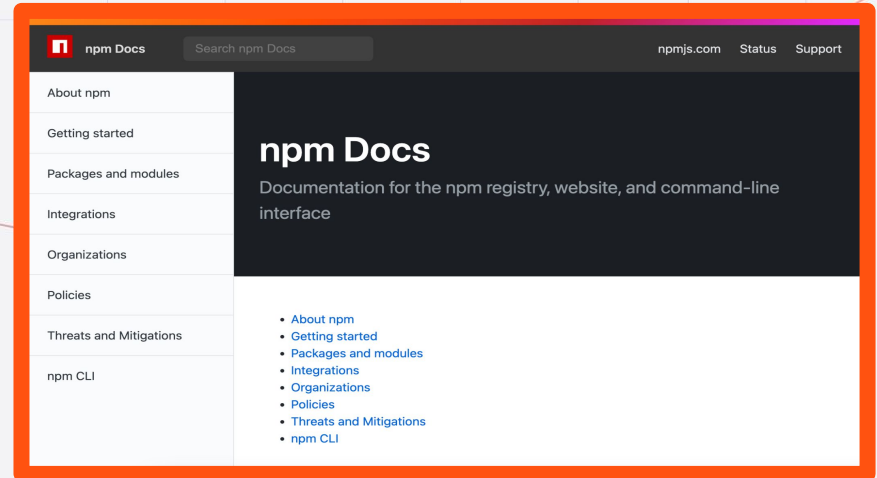
18.11.0 Current
Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#) | [Other Downloads](#) | [Changelog](#) | [API Docs](#)

For information about supported releases, see the [release schedule](#).

Copyright OpenJS Foundation and Node.js contributors. All rights reserved. The OpenJS Foundation has registered trademarks and uses trademarks. For a list of trademarks of the OpenJS Foundation, please see our [Trademark Policy](#) and [Trademark List](#). Trademarks and logos not indicated on the list of OpenJS Foundation trademarks are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by them.

[The OpenJS Foundation](#) | [Terms of Use](#) | [Privacy Policy](#) | [Bylaws](#) | [Code of Conduct](#) | [Trademark Policy](#) | [Trademark List](#) | [Cookie Policy](#) | [Edit On GitHub](#)



npm Docs Search npm Docs npmjs.com Status Support

About npm

Getting started

Packages and modules

Integrations

Organizations

Policies

Threats and Mitigations

npm CLI

npm Docs

Documentation for the npm registry, website, and command-line interface

- [About npm](#)
- [Getting started](#)
- [Packages and modules](#)
- [Integrations](#)
- [Organizations](#)
- [Policies](#)
- [Threats and Mitigations](#)
- [npm CLI](#)

Ambiente di Sviluppo

- **MAMP** - <https://www.mamp.info/>
- **XAMPP** - <https://www.apachefriends.org/>
- **DevKinsta** - <https://kinsta.com/it/devkinsta/>
- **wp-env** - <https://www.npmjs.com/package/@wordpress/env>



Apache Friends Scarica Componenti aggiuntivi Hosting Comunità Chi siamo Cerca...

XAMPP Apache + MariaDB + PHP +

Cos'è XAMPP?

XAMPP è il più popolare ambiente di sviluppo PHP. XAMPP è una distribuzione di Apache completamente gratuita e semplice da installare, contenente MySQL, PHP e Perl. Il pacchetto open source XAMPP è stato creato per essere estremamente facile da installare e utilizzare.

Scarica

- XAMPP per Windows 8.1.10 (PHP 8.1.10)
- XAMPP per Linux 8.1.10 (PHP 8.1.10)
- XAMPP per macOS 8.1.6 (PHP 8.1.10)



MAMP PRO MAMP NAMO More products... Downloads Support

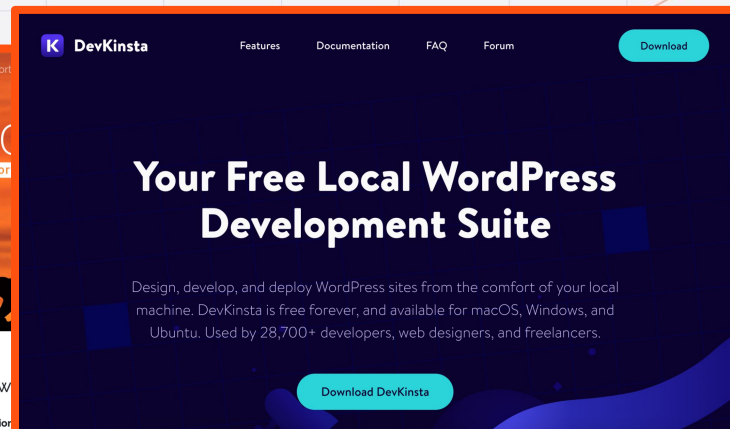
MAMP & MAMP PRO

Your local web development solution PHP & Support

Free Download Try Now Buy Now

MAMP PRO for Windows

MAMP PRO is the commercial, professional version of the classic local server environment: MAMP. With MAMP PRO you can create a separate host for each of your web



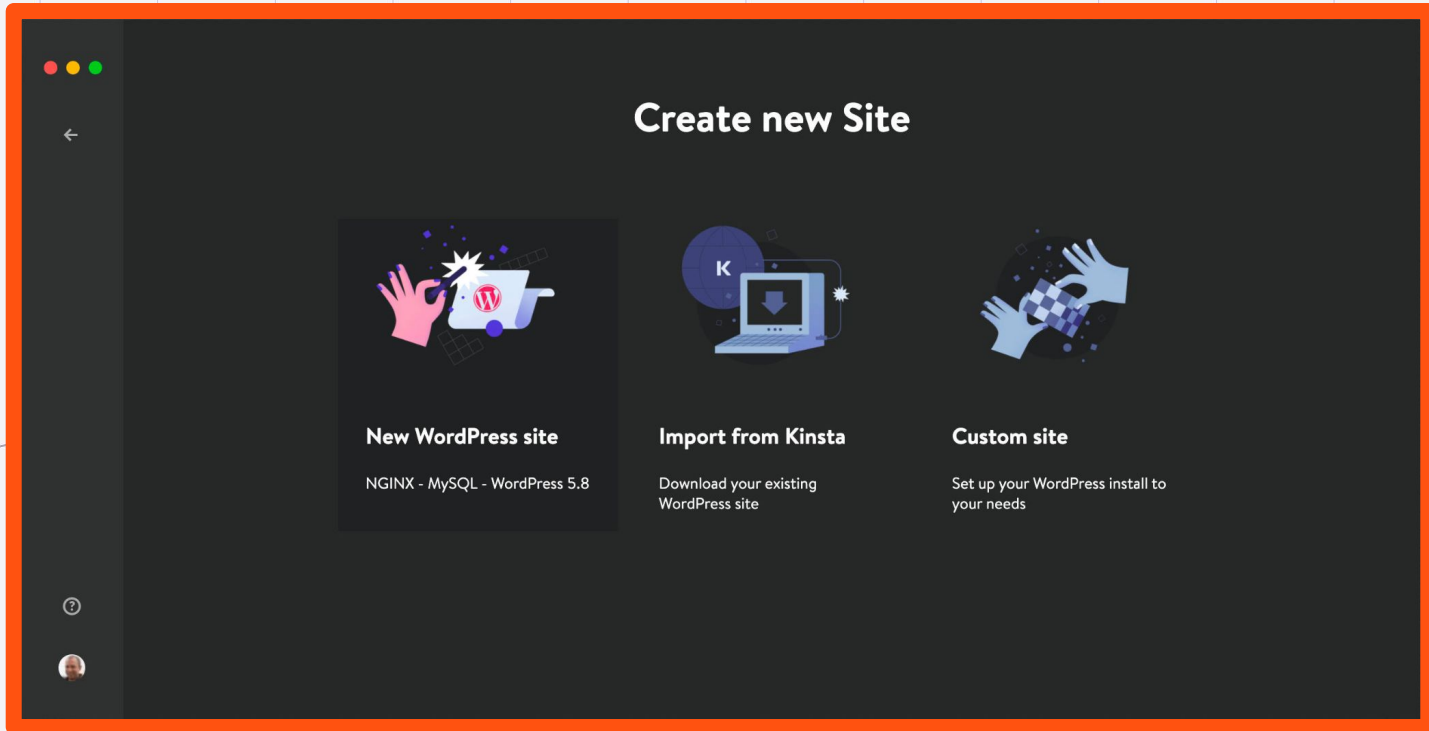
DevKinsta Features Documentation FAQ Forum Download

Your Free Local WordPress Development Suite

Design, develop, and deploy WordPress sites from the comfort of your local machine. DevKinsta is free forever, and available for macOS, Windows, and Ubuntu. Used by 28,700+ developers, web designers, and freelancers.

Download DevKinsta

DevKinsta



@wordpress/create-block

Dalla directory `/wp-content/plugins/` eseguire questo comando:

```
npx @wordpress/create-block my-first-block
```

@wordpress/create-block

Builds the code for production.

```
$ npm run format  
Formats files.
```

```
$ npm run lint:css  
Lints CSS files.
```

```
$ npm run lint:js  
Lints JavaScript files.
```

```
$ npm run packages-update  
Updates WordPress packages to the latest version.
```

To enter the folder type:

```
$ cd my-first-block
```

You can start development with:

```
$ npm start
```

```
Code is Poetry  
192:plugins carlodaniele$ █
```

@wordpress/create-block in Modalità Interattiva

Dalla directory `/wp-content/plugins/` eseguire questo comando:

```
npx @wordpress/create-block
```

@wordpress/create-block in Modalità Interattiva

```
[192:plugins carlodaniele$ npx @wordpress/create-block  
Let's customize your WordPress plugin with blocks:  
? The template variant to use for this block: static  
? The block slug used for identification (also the output folder name): ka-examp  
[le-block  
? The internal namespace for the block name (something unique for your products)  
[: ka-example-block  
? The display title for your block: Kinsta Academy Block  
? The short description for your block (optional): An example block for Kinsta A  
[cademy students  
? The dashicon to make it easier to identify your block (optional): superhero-al  
t  
? The category name to help users browse and discover your block: widgets  
? Do you want to customize the WordPress plugin? Yes  
? The home page of the plugin (optional). Unique URL outside of WordPress.org: h  
[ttps://kinsta.com/  
? The current version number of the plugin: 0.1.0  
? The name of the plugin author (optional). Multiple authors may be listed using  
[commas: Kinsta Students  
? The short name of the plugin's license (optional): GPL-2.0-or-later  
? A link to the full text of the license (optional): https://www.gnu.org/license  
[s/gpl-2.0.html  
? A custom domain path for the translations (optional): █
```

@wordpress/create-block

Builds the code for production.

```
$ npm run format  
Formats files.
```

```
$ npm run lint:css  
Lints CSS files.
```

```
$ npm run lint:js  
Lints JavaScript files.
```

```
$ npm run packages-update  
Updates WordPress packages to the latest version.
```

To enter the folder type:

```
$ cd my-first-block
```

You can start development with:

```
$ npm start
```

```
Code is Poetry  
192:plugins carlodaniele$ █
```

Il Mio Primo Block Plugin

The screenshot shows the WordPress Plugins management interface. At the top, there are 'Screen Options' and 'Help' dropdown menus. Below them is the 'Plugins' title and an 'Add New' button. A filter bar shows 'All (3) | Active (1) | Inactive (2) | Auto-updates Disabled (3)' and a search box for installed plugins. A 'Bulk actions' dropdown and an 'Apply' button are visible. The main content is a table of installed plugins. The 'My First Block' plugin is highlighted in light blue. Below the table, another 'Bulk actions' dropdown and 'Apply' button are present.

Plugins [Add New](#) Screen Options ▾ Help ▾

All (3) | [Active \(1\)](#) | [Inactive \(2\)](#) | [Auto-updates Disabled \(3\)](#)

Bulk actions ▾ [Apply](#) 3 items

<input type="checkbox"/>	Plugin	Description	Automatic Updates
<input type="checkbox"/>	Akismet Anti-Spam Activate Delete	Used by millions, Akismet is quite possibly the best way in the world to protect your blog from spam . It keeps your site protected even while you sleep. To get started: activate the Akismet plugin and then go to your Akismet Settings page to set up your API key. Version 4.1.12 By Automattic View details	Enable auto-updates
<input type="checkbox"/>	Hello Dolly Activate Delete	This is not just a plugin, it symbolizes the hope and enthusiasm of an entire generation summed up in two words sung most famously by Louis Armstrong: Hello, Dolly. When activated you will randomly see a lyric from Hello, Dolly in the upper right of your admin screen on every page. Version 1.7.2 By Matt Mullenweg View details	Enable auto-updates
<input type="checkbox"/>	My First Block Deactivate	Example block written with ESNext standard and JSX support – build step required. Version 0.1.0 By The WordPress Contributors	
<input type="checkbox"/>	Plugin	Description	Automatic Updates

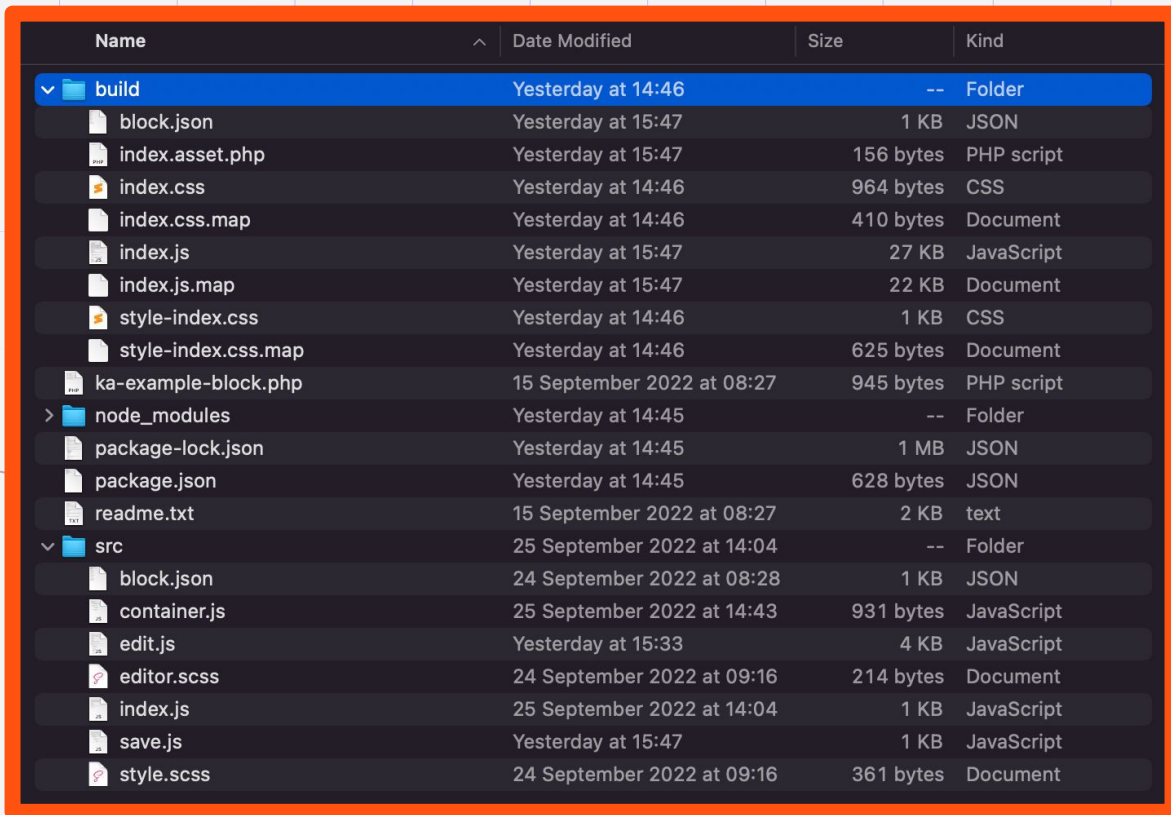
Bulk actions ▾ [Apply](#) 3 items

Il Mio Primo Blocco

The screenshot displays the WordPress Gutenberg editor interface. At the top left, there is a search bar with the text "Search" and a magnifying glass icon. Below the search bar, there are two tabs: "Blocks" (which is active) and "Patterns". Under the "Blocks" tab, there are three columns of block options: "Custom HTML", "Latest Comments", and "Latest Posts". The "Custom HTML" column contains "Page List", "Shortcode", and "Kinsta Academy Block". The "Latest Comments" column contains "RSS" and "My First Block". The "Latest Posts" column contains "Search" and "Tag Cloud".

In the main editor area, there is a large grey box with the text "No Preview Available." and a smaller white box below it containing a smiley face icon and the text "My First Block" followed by "Example block scaffolded with Create Block tool." Below this, there is a toolbar with a smiley face icon, a grid icon, an up arrow icon, and a down arrow icon. At the bottom of the editor area, there is a blue bar with the text "My First Block – hello from the editor!".

L'Impalcatura del Blocco (Block Scaffolding)

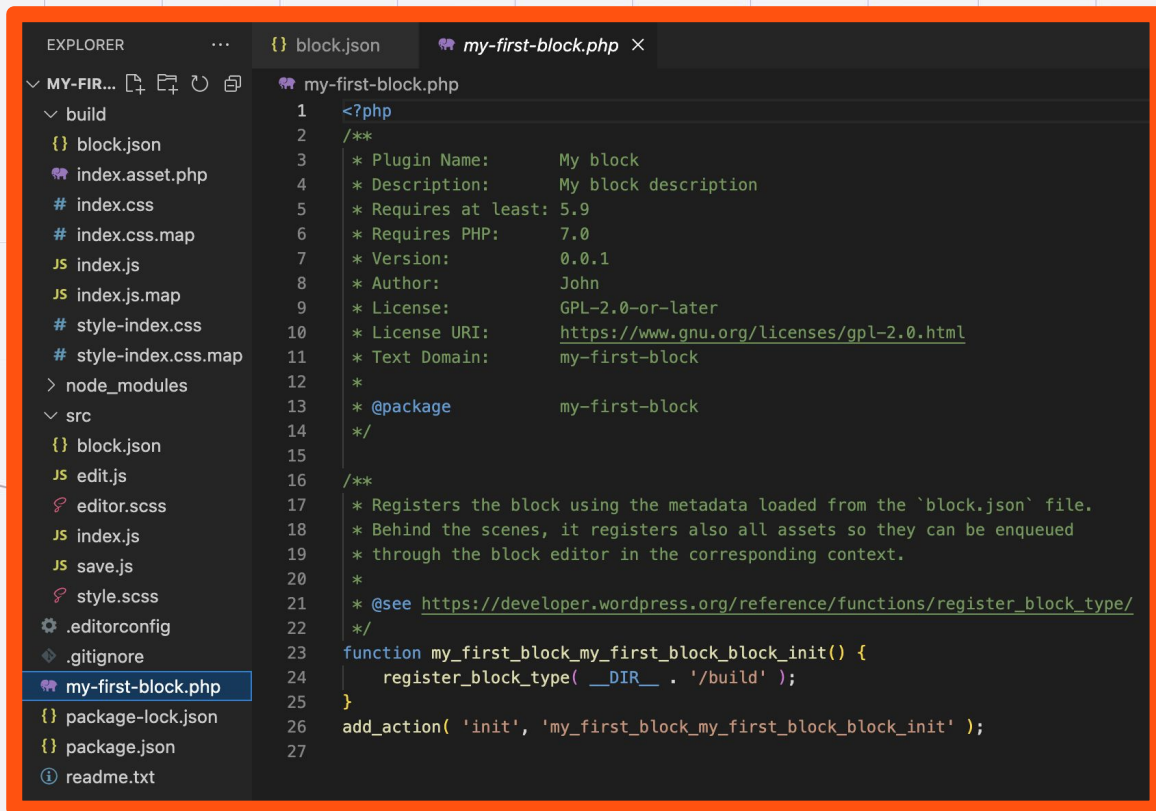


Name	Date Modified	Size	Kind
build	Yesterday at 14:46	--	Folder
block.json	Yesterday at 15:47	1 KB	JSON
index.asset.php	Yesterday at 15:47	156 bytes	PHP script
index.css	Yesterday at 14:46	964 bytes	CSS
index.css.map	Yesterday at 14:46	410 bytes	Document
index.js	Yesterday at 15:47	27 KB	JavaScript
index.js.map	Yesterday at 15:47	22 KB	Document
style-index.css	Yesterday at 14:46	1 KB	CSS
style-index.css.map	Yesterday at 14:46	625 bytes	Document
ka-example-block.php	15 September 2022 at 08:27	945 bytes	PHP script
node_modules	Yesterday at 14:45	--	Folder
package-lock.json	Yesterday at 14:45	1 MB	JSON
package.json	Yesterday at 14:45	628 bytes	JSON
readme.txt	15 September 2022 at 08:27	2 KB	text
src	25 September 2022 at 14:04	--	Folder
block.json	24 September 2022 at 08:28	1 KB	JSON
container.js	25 September 2022 at 14:43	931 bytes	JavaScript
edit.js	Yesterday at 15:33	4 KB	JavaScript
editor.scss	24 September 2022 at 09:16	214 bytes	Document
index.js	25 September 2022 at 14:04	1 KB	JavaScript
save.js	Yesterday at 15:47	1 KB	JavaScript
style.scss	24 September 2022 at 09:16	361 bytes	Document

Il File PHP

Il file PHP del plugin

registra il blocco sul server



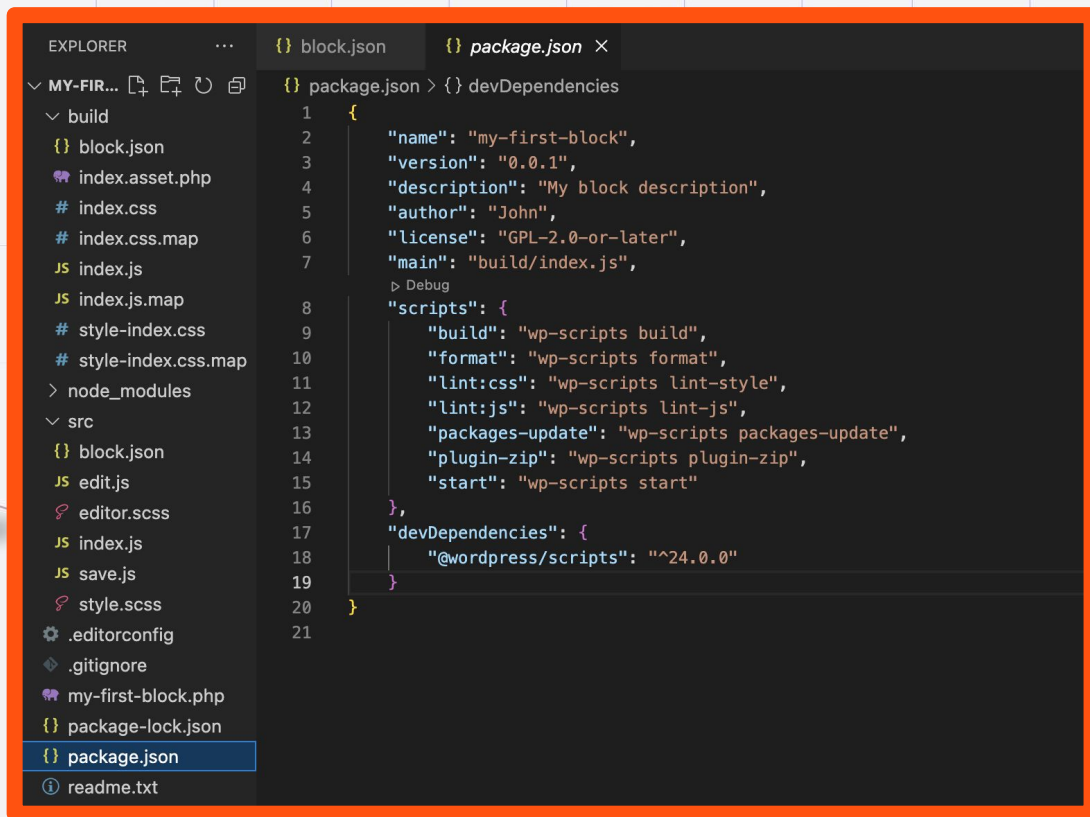
```
EXPLORER  ...  {} block.json  my-first-block.php x
v MY-FIR...  {} block.json
  v build
    {} block.json
    index.asset.php
    # index.css
    # index.css.map
    JS index.js
    JS index.js.map
    # style-index.css
    # style-index.css.map
  > node_modules
  v src
    {} block.json
    JS edit.js
    editor.scss
    JS index.js
    JS save.js
    editorconfig
    .editorconfig
    .gitignore
    my-first-block.php
    {} package-lock.json
    {} package.json
    readme.txt

1  <?php
2  /**
3   * Plugin Name:      My block
4   * Description:     My block description
5   * Requires at least: 5.9
6   * Requires PHP:    7.0
7   * Version:         0.0.1
8   * Author:         John
9   * License:        GPL-2.0-or-later
10  * License URI:    https://www.gnu.org/licenses/gpl-2.0.html
11  * Text Domain:    my-first-block
12  *
13  * @package        my-first-block
14  */
15
16 /**
17  * Registers the block using the metadata loaded from the `block.json` file.
18  * Behind the scenes, it registers also all assets so they can be enqueued
19  * through the block editor in the corresponding context.
20  *
21  * @see https://developer.wordpress.org/reference/functions/register_block_type/
22  */
23 function my_first_block_my_first_block_block_init() {
24     register_block_type( __DIR__ . '/build' );
25 }
26 add_action( 'init', 'my_first_block_my_first_block_block_init' );
27
```

Il File package.json

Il file package.json

definisce proprietà e script
utilizzati nel progetto

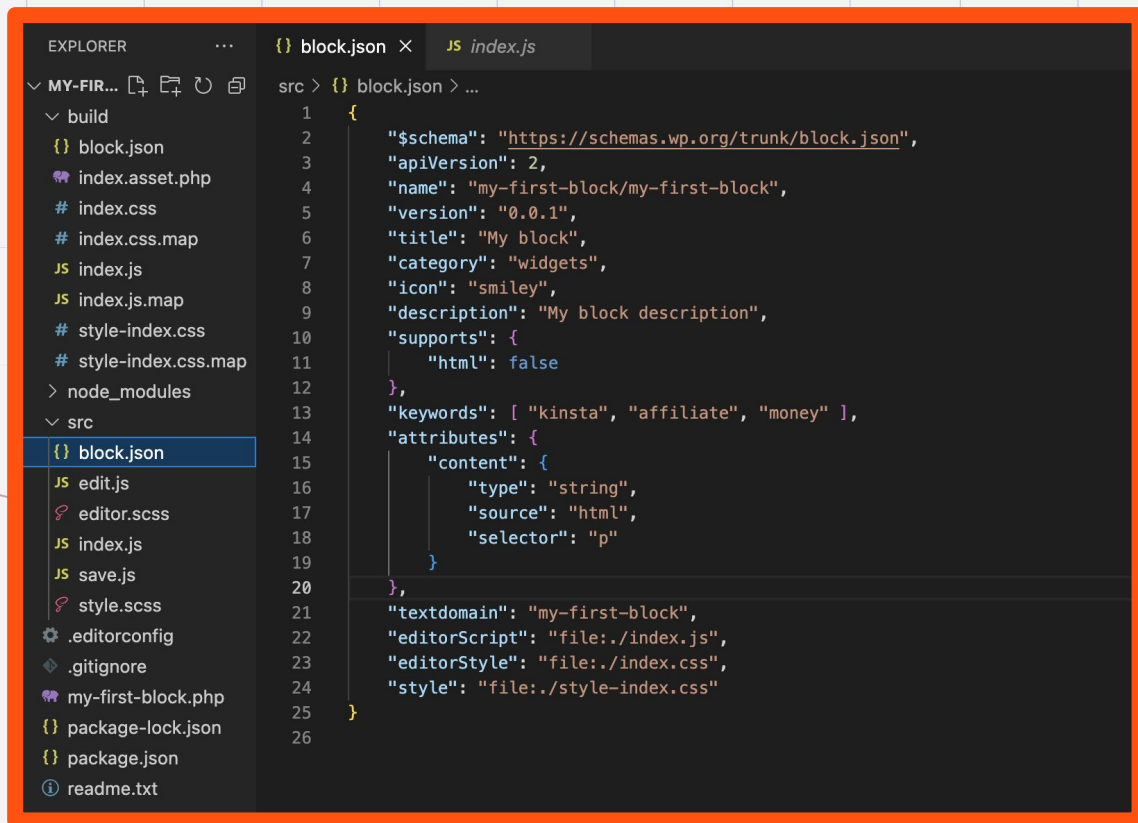


```
EXPLORER  ...  {} block.json  {} package.json ×
└─ MY-FIR...  {} package.json > {} devDependencies
  └─ build
    {} block.json
    index.asset.php
    # index.css
    # index.css.map
    JS index.js
    JS index.js.map
    # style-index.css
    # style-index.css.map
  > node_modules
  └─ src
    {} block.json
    JS edit.js
    editor.scss
    JS index.js
    JS save.js
    editorconfig
    .editorconfig
    .gitignore
    my-first-block.php
    {} package-lock.json
    {} package.json
    readme.txt

1  {
2    "name": "my-first-block",
3    "version": "0.0.1",
4    "description": "My block description",
5    "author": "John",
6    "license": "GPL-2.0-or-later",
7    "main": "build/index.js",
8    "scripts": {
9      "build": "wp-scripts build",
10     "format": "wp-scripts format",
11     "lint:css": "wp-scripts lint-style",
12     "lint:js": "wp-scripts lint-js",
13     "packages-update": "wp-scripts packages-update",
14     "plugin-zip": "wp-scripts plugin-zip",
15     "start": "wp-scripts start"
16   },
17   "devDependencies": {
18     "@wordpress/scripts": "^24.0.0"
19   }
20 }
21
```

Il File block.json

Il file block.json è il file che contiene i metadati del progetto e costituisce la modalità canonica di registrazione dei tipi di blocco



The image shows a screenshot of a code editor (VS Code) with a dark theme. On the left, the Explorer sidebar shows a file tree for a project named 'MY-FIR...'. The 'src' directory is expanded, and 'block.json' is selected. The main editor area shows the content of 'block.json' with line numbers 1 through 26. The JSON content is as follows:

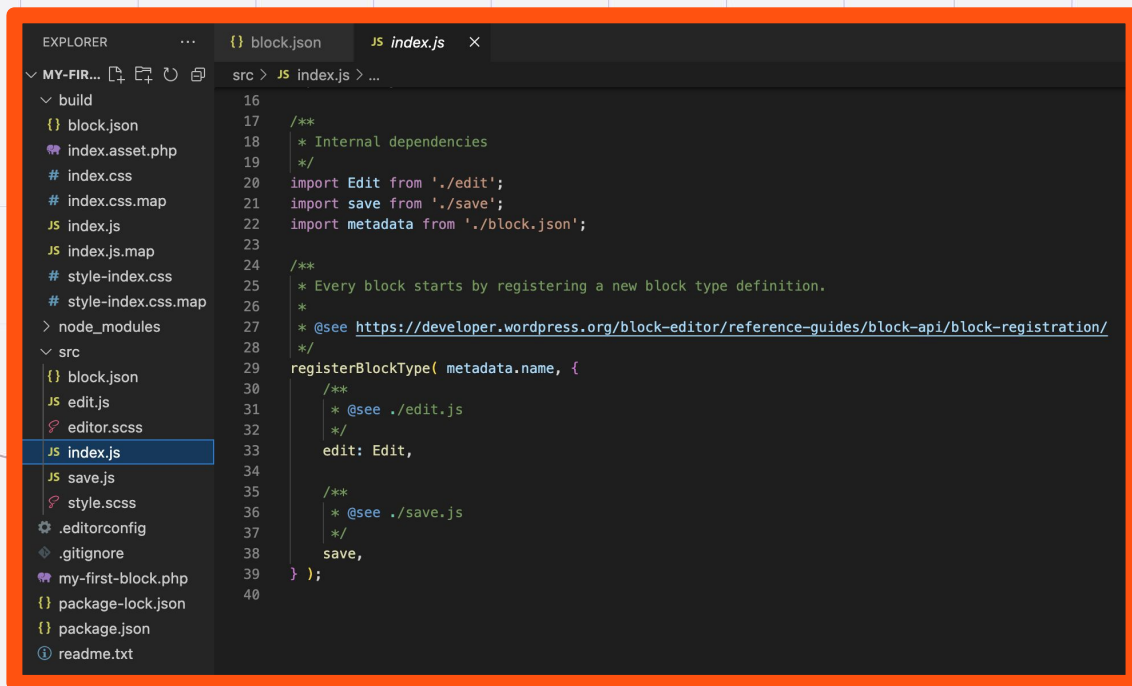
```
1 {
2   "$schema": "https://schemas.wp.org/trunk/block.json",
3   "apiVersion": 2,
4   "name": "my-first-block/my-first-block",
5   "version": "0.0.1",
6   "title": "My block",
7   "category": "widgets",
8   "icon": "smiley",
9   "description": "My block description",
10  "supports": {
11    "html": false
12  },
13  "keywords": [ "kinsta", "affiliate", "money" ],
14  "attributes": {
15    "content": {
16      "type": "string",
17      "source": "html",
18      "selector": "p"
19    }
20  },
21  "textdomain": "my-first-block",
22  "editorScript": "file:./index.js",
23  "editorStyle": "file:./index.css",
24  "style": "file:./style-index.css"
25 }
26
```

Il File block.json - Definizione degli Attributi del Blocco

```
"attributes": {
  "content": {
    "type": "array",
    "source": "children",
    "selector": "p"
  },
  "align": {
    "type": "string",
    "default": "none"
  },
  "link": {
    "type": "string",
    "default": "https://example.com"
  }
},
```

Il File index.js

È il file da cui vengono importate le dipendenze ed è responsabile della registrazione del blocco sul client

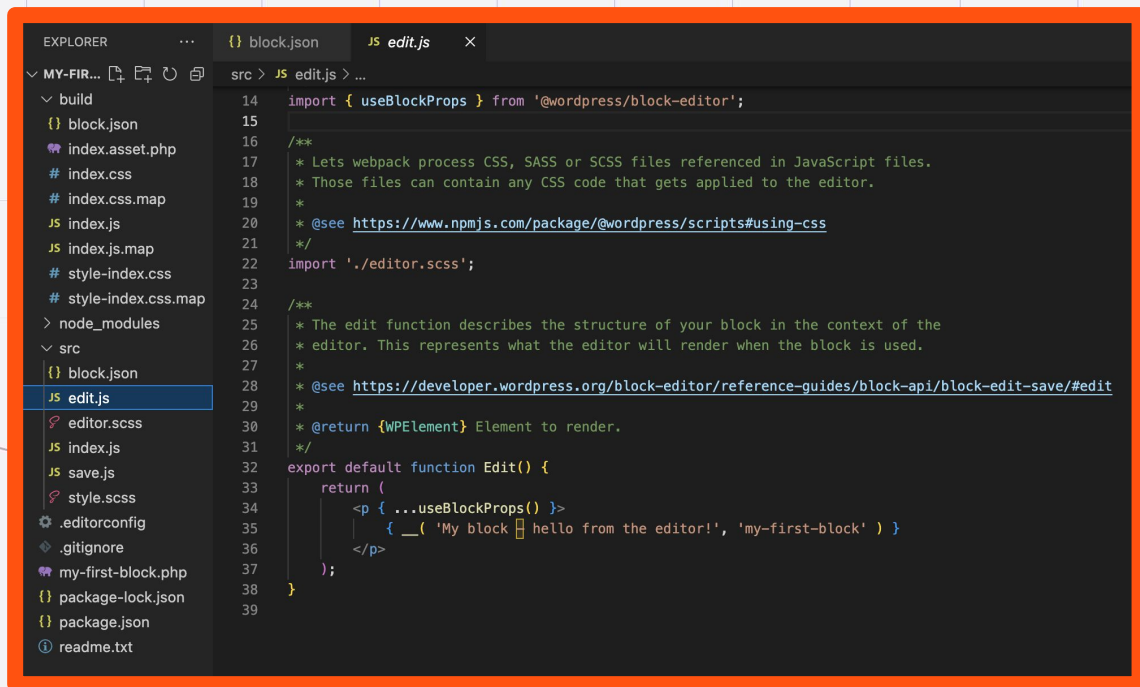


```
EXPLORER  ...  {} block.json  JS index.js  X
MY-FIR...  {} block.json
  build
  {} block.json
  index.asset.php
  # index.css
  # index.css.map
  JS index.js
  JS index.js.map
  # style-index.css
  # style-index.css.map
  > node_modules
  src
  {} block.json
  JS edit.js
  editor.scss
  JS index.js
  JS save.js
  style.scss
  .editorconfig
  .gitignore
  my-first-block.php
  {} package-lock.json
  {} package.json
  readme.txt

src > JS index.js > ...
16
17  /**
18   * Internal dependencies
19   */
20  import Edit from './edit';
21  import save from './save';
22  import metadata from './block.json';
23
24  /**
25   * Every block starts by registering a new block type definition.
26   *
27   * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-registration/
28   */
29  registerBlockType( metadata.name, {
30    /**
31     * @see ./edit.js
32     */
33    edit: Edit,
34
35    /**
36     * @see ./save.js
37     */
38    save,
39  } );
40
```

Il File edit.js

È il file che genera l'interfaccia di editing del blocco

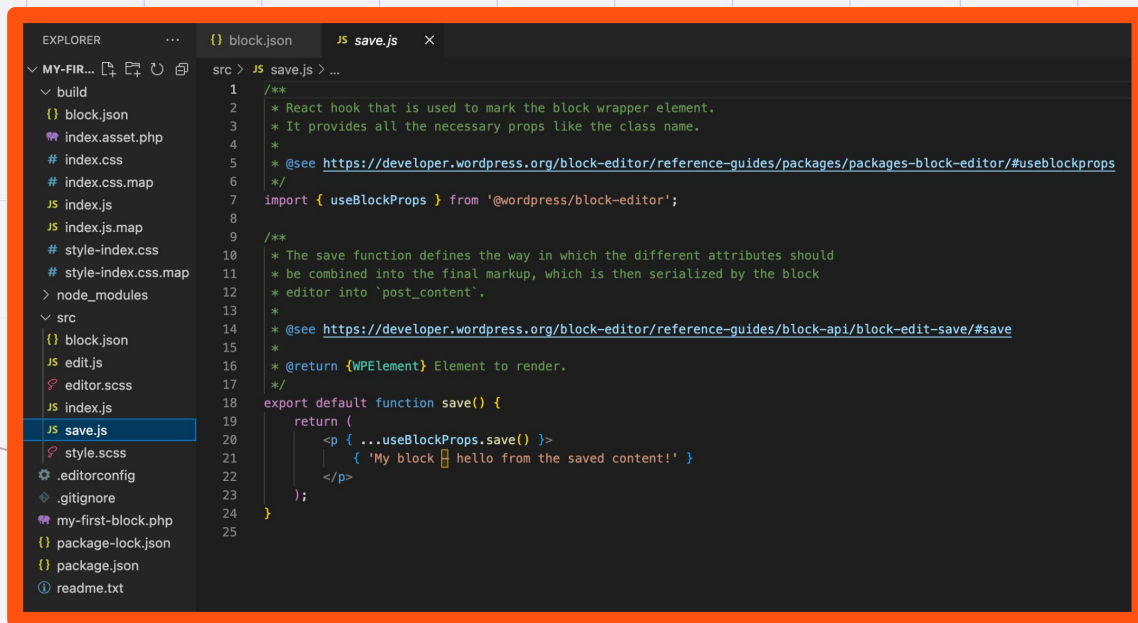


The screenshot shows a code editor window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a 'src' directory containing 'block.json' and 'edit.js'. The 'edit.js' file is selected and highlighted. The code editor shows the following code:

```
14 import { useBlockProps } from '@wordpress/block-editor';
15
16 /**
17  * Lets webpack process CSS, SASS or SCSS files referenced in JavaScript files.
18  * Those files can contain any CSS code that gets applied to the editor.
19  *
20  * @see https://www.npmjs.com/package/@wordpress/scripts#using-css
21  */
22 import './editor.scss';
23
24 /**
25  * The edit function describes the structure of your block in the context of the
26  * editor. This represents what the editor will render when the block is used.
27  *
28  * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-edit-save/#edit
29  *
30  * @return {WPElement} Element to render.
31  */
32 export default function Edit() {
33   return (
34     <p { ...useBlockProps() }>
35       { __( 'My block', 'my-first-block' ) }
36     </p>
37   );
38 }
39
```

Il File save.js

È il file che genera la struttura del blocco che viene serializzata e salvata nel database



```
1  /**
2  * React hook that is used to mark the block wrapper element.
3  * It provides all the necessary props like the class name.
4  *
5  * @see https://developer.wordpress.org/block-editor/reference-guides/packages/packages-block-editor/#useblockprops
6  */
7  import { useBlockProps } from '@wordpress/block-editor';
8
9  /**
10 * The save function defines the way in which the different attributes should
11 * be combined into the final markup, which is then serialized by the block
12 * editor into 'post_content'.
13 *
14 * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-edit-save/#save
15 *
16 * @return {WPElement} Element to render.
17 */
18 export default function save() {
19   return (
20     <p { ...useBlockProps.save() }>
21       { 'My block ' + hello from the saved content!' }
22     </p>
23   );
24 }
25
```


Avviare l'Ambiente di Sviluppo

Tornare al terminale e spostarsi nella directory

my-first-block:

```
Builds the code for production.

$ npm run format
Formats files.

$ npm run lint:css
Lints CSS files.

$ npm run lint:js
Lints JavaScript files.

$ npm run packages-update
Updates WordPress packages to the latest version.

To enter the folder type:

$ cd my-first-block

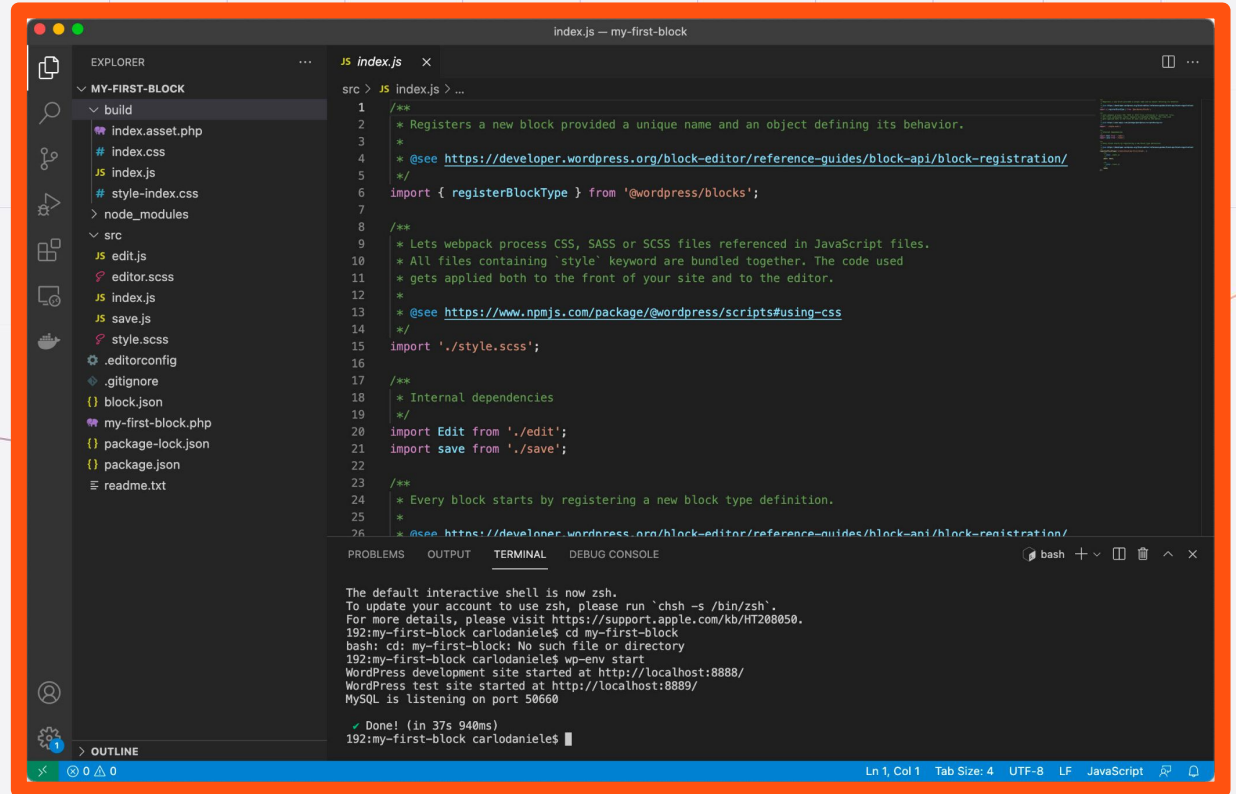
You can start development with:

$ npm start

Code is Poetry
192:plugins carlodaniele$
```

```
cd my-first-block
```

Avviare il Terminale in Visual Studio Code



The screenshot shows the Visual Studio Code interface with a terminal window open. The Explorer view on the left shows a project named 'MY-FIRST-BLOCK' with a 'src' directory containing files like 'index.js', 'edit.js', and 'save.js'. The main editor displays the content of 'index.js', which is a JavaScript file for registering a WordPress block. The terminal window shows the following output:

```
src > JS index.js > ...
1 /**
2  * Registers a new block provided a unique name and an object defining its behavior.
3  *
4  * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-registration/
5  */
6 import { registerBlockType } from '@wordpress/blocks';
7
8 /**
9  * Lets webpack process CSS, SASS or SCSS files referenced in JavaScript files.
10 * All files containing 'style' keyword are bundled together. The code used
11 * gets applied both to the front of your site and to the editor.
12 *
13 * @see https://www.npmjs.com/package/@wordpress/scripts#using-css
14 */
15 import './style.scss';
16
17 /**
18 * Internal dependencies
19 */
20 import Edit from './edit';
21 import save from './save';
22
23 /**
24 * Every block starts by registering a new block type definition.
25 *
26 * @see https://developer.wordpress.org/block-editor/reference-guides/block-api/block-registration/
```

The terminal output shows the following commands and results:

```
bash
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
192:my-first-block carlodaniele$ cd my-first-block
bash: cd: my-first-block: No such file or directory
192:my-first-block carlodaniele$ wp-env start
WordPress development site started at http://localhost:8888/
WordPress test site started at http://localhost:8889/
MySQL is listening on port 50660

Done! (in 37s 940ms)
192:my-first-block carlodaniele$
```

The status bar at the bottom indicates the current file is 'index.js' in the 'src' directory, with a tab size of 4, UTF-8 encoding, and LF line endings.

npm

Avviare l'ambiente di sviluppo:

```
npm start
```

Creare gli script di produzione:

```
npm run build
```

Aggiungere un Componente ad un Blocco

Per aggiungere un componente ad un blocco è necessario:

1. Importare i componenti necessari da un pacchetto WordPress
2. Includere gli elementi corrispondenti nel codice JSX
3. Definire gli attributi necessari nel file `block.json`
4. Definire gli event handler
5. Salvare i dati

1. Importare il Componente da un Pacchetto WordPress

Nel file `edit.js`:

```
import { useBlockProps, RichText } from '@wordpress/block-editor';
```

2. Includere gli Elementi Corrispondenti nel Codice JSX

Nel file `edit.js`:

```
export default function Edit( { attributes, setAttributes } ) {  
  const blockProps = useBlockProps();  
  return (  
    <RichText  
      { ...blockProps }  
      tagName="p"  
      onChange={ onChangeContent }  
      allowedFormats={ [ 'core/bold', 'core/italic' ] }  
      value={ attributes.content }  
      placeholder={ __( 'Write your text...' ) }  
    />  
  );  
}
```

3: Definire gli Attributi nel File block.json

Nel file **block.json**:

```
"attributes": {  
  "content": {  
    "type": "string",  
    "source": "html",  
    "selector": "p"  
  }  
},
```

4: Definire gli Event Handlers

Nel file `edit.js`:

```
export default function Edit( { attributes, setAttributes } ) {  
  const blockProps = useBlockProps();  
  
  const onChangeContent = ( newContent ) => {  
    setAttributes( { content: newContent } )  
  }  
  
  return (  
    <RichText ... />  
  );  
}
```

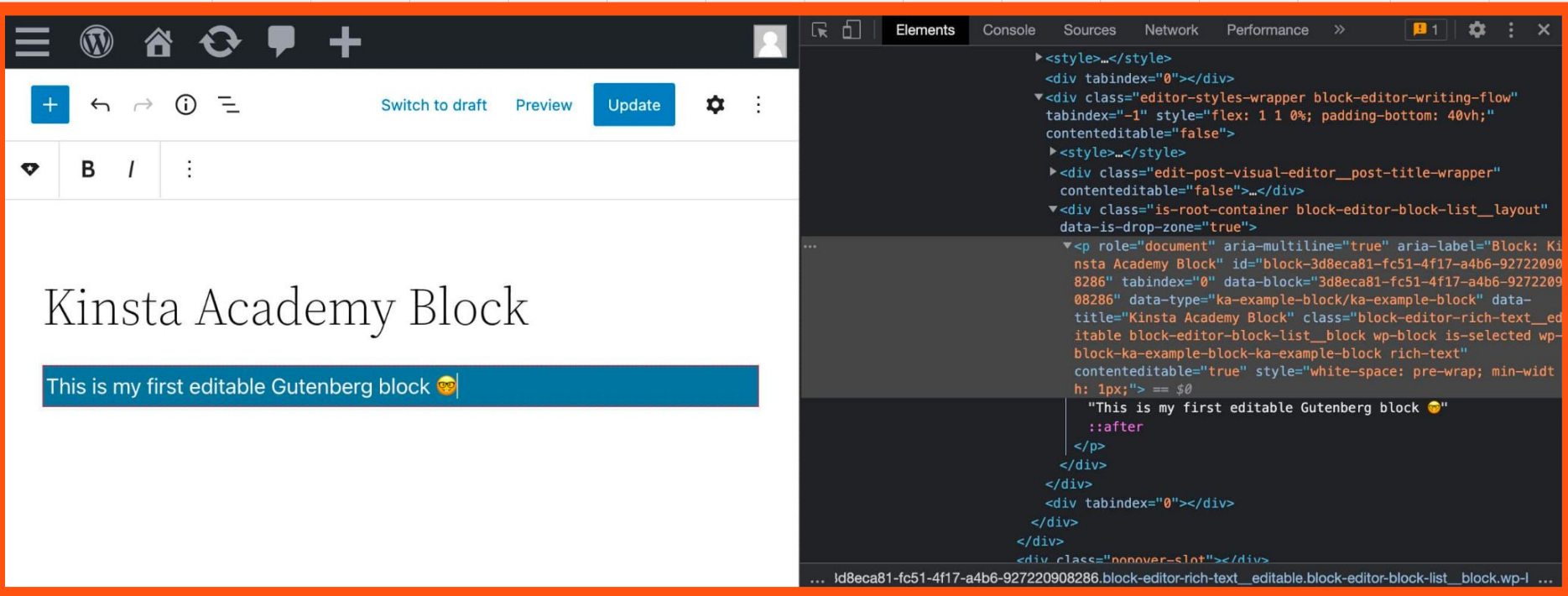

5: Salvare i Dati

Nel file `save.js`:

```
import { __ } from '@wordpress/i18n';
import { useBlockProps, RichText } from '@wordpress/block-editor';

export default function save( { attributes } ) {
  const blockProps = useBlockProps.save();
  return (
    <RichText.Content
      { ...blockProps }
      tagName="p"
      value={ attributes.content }
    />
  );
}
```

Il Blocco nel Front End



The image shows a screenshot of the WordPress Gutenberg editor interface. The top navigation bar includes icons for home, refresh, chat, and a plus sign, along with a user profile icon. Below the navigation bar, there are buttons for "Switch to draft", "Preview", and "Update". The main editing area displays a custom block titled "Kinsta Academy Block" with the text "This is my first editable Gutenberg block 🤖". The right-hand side of the image shows the developer tools, specifically the "Elements" panel, which displays the HTML structure of the block. The HTML includes a `<div>` with a `tabindex="0"` attribute, a `<div>` with a `class="editor-styles-wrapper block-editor-writing-flow"` attribute, and a `<div>` with a `class="edit-post-visual-editor__post-title-wrapper"` attribute. The main content is wrapped in a `<p>` tag with a `role="document"` attribute and an `aria-multiline="true"` attribute. The text content is "This is my first editable Gutenberg block 🤖".

Kinsta Academy Block

This is my first editable Gutenberg block 🤖

```
<style>...</style>
<div tabindex="0"></div>
<div class="editor-styles-wrapper block-editor-writing-flow"
tabindex="-1" style="flex: 1 1 0%; padding-bottom: 40vh;"
contenteditable="false">
  <style>...</style>
  <div class="edit-post-visual-editor__post-title-wrapper"
contenteditable="false">...</div>
  <div class="is-root-container block-editor-block-list__layout"
data-is-drop-zone="true">
...
  <p role="document" aria-multiline="true" aria-label="Block: Ki
nsta Academy Block" id="block-3d8eca81-fc51-4f17-a4b6-92722090
8286" tabindex="0" data-block="3d8eca81-fc51-4f17-a4b6-9272209
08286" data-type="ka-example-block/ka-example-block" data-
title="Kinsta Academy Block" class="block-editor-rich-text__ed
itable block-editor-block-list__block wp-block is-selected wp-
block-ka-example-block-ka-example-block rich-text"
contenteditable="true" style="white-space: pre-wrap; min-widt
h: 1px;"> == $0
  "This is my first editable Gutenberg block 🤖"
  ::after
</p>
</div>
</div>
<div tabindex="0"></div>
</div>
<div class="nonover-slot"></div>
... ld8eca81-fc51-4f17-a4b6-927220908286.block-editor-rich-text__editable.block-editor-block-list__block.wp-l ...
```

Risorse



Creare Blocchi Gutenberg Personalizzati: Guida Completa

<https://kinsta.com/it/blog/blocchi-gutenberg/>

Sviluppare Blocchi per Gutenberg: Videocorso Gratuito

<https://kinsta.com/academy/course/gutenberg-block-development/>