@wordpress/create-block

- Quick start
- Usage
  - Interactive Mode
  - namespace
  - slug
  - options
    - --template
    - --variant
    - --no-plugin
    - --wp-env
    - --textdomain
    - --help
- Available commands in the scaffolded project
- External Project Templates
- Contributing to this package

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

It is largely inspired by create-react-app. Major kudos to @gaearon, the whole Facebook team, and the React community.

> **Blocks are the fundamental elements of modern WordPress sites**.
>
> Introduced in WordPress 5.0, they allow page and post builder-like functionality to every up-to-date WordPress website.
>
> *Learn more about the Block API at the Gutenberg HandBook.*

Quick start

```
$ npx @wordpress/create-block@latest todo-list
$ cd todo-list
$ npm start
```

The `slug` provided (`todo-list` in the example) defines the folder name for the scaffolded plugin and the internal block name. The WordPress plugin generated must be installed manually.

(requires `node` version `>=22.13.0`)

> Watch a video introduction to create-block on Learn.wordpress.org

Usage

The `create-block` command generates a project with PHP, JS, and CSS code for registering a block with a WordPress plugin.

```
$ npx @wordpress/create-block@latest [options] [slug]
```

Demo

> The name for a block is a unique string that identifies a block. Block Names are structured as `namespace`/`slug`, where namespace is the name of your plugin or theme.
>
> In most cases, we recommended pairing blocks with WordPress plugins rather than themes, because only using plugin ensures that all blocks still work when your theme changes.

Interactive Mode

When no `slug` is provided, the script will run in interactive mode and will start prompting for the input required (`slug`, title, namespace...) to scaffold the project.

`namespace`

By default, blocks are created with the `create-block` namespace. You should specify your own unique namespace:

```
$ npx @wordpress/create-block@latest my-block --namespace=my-namespace
```

This creates `my-namespace/my-block` instead of `create-block/my-block`.

If you've already created a block, update the namespace in:
– `block.json` – the `name` property

`slug`

The use of `slug` is optional.

When provided it triggers the *quick mode*, where this `slug` is used:

- as the block slug (required for its identification)
- as the output location (folder name) for scaffolded files
- as the name of the WordPress plugin.

The rest of the configuration is set to all default values unless overridden with some options listed below.

`options`

```
-V, --version                output the version number
-t, --template <name>        project template type name; allowed values: "standard" (default),
"es5", the name of an external npm package, or the path to a local directory
--variant                    choose a block variant as defined by the template
--no-plugin                  scaffold block files only
--target-dir <directory>     the directory where the files will be scaffolded, defaults to the
slug
--namespace <value>          internal namespace for the block name
--title <value>              display title for the block and the WordPress plugin
--short-description <value>  short description for the block and the WordPress plugin
--category <name>            category name for the block
--wp-scripts                 enable integration with `@wordpress/scripts` package
--no-wp-scripts              disable integration with `@wordpress/scripts` package
--wp-env                     enable integration with `@wordpress/env` package
```

```
--textdomain <value>          text domain for internationalization
-h, --help                    output usage information
```

`--template`

This argument specifies an *external npm package* as a template.

```
$ npx @wordpress/create-block@latest --template my-template-package
```

This argument also allows to pick a *local directory* as a template.

```
$ npx @wordpress/create-block@latest --template ./path/to/template-directory
```

`--variant`

With this argument, `create-block` will generate a dynamic block based on the built-in template.

```
$ npx @wordpress/create-block@latest --variant dynamic
```

`--no-plugin`

With this argument, the `create-block` package runs in *No plugin mode* which only scaffolds block files into the current directory.

```
$ npx @wordpress/create-block@latest --no-plugin
```

`--wp-env`

With this argument, the `create-block` package will add to the generated plugin the configuration and the script to run `wp-env` package within the plugin. This will allow you to easily set up a local WordPress environment (via Docker) for building and testing the generated plugin.

```
$ npx @wordpress/create-block@latest --wp-env
```

`--textdomain`

With this argument, the `create-block` package will a generate a block with the provided text domain. If not specified, the block's slug is used as the default text domain.

```
$ npx @wordpress/create-block@latest --textdomain my-custom-domain
```

`--help`

With this argument, the `create-block` package outputs usage information.

```
$ npx @wordpress/create-block@latest --help
```

Available commands in the scaffolded project

The plugin folder created when executing this command, is a node package with a modern build setup that requires no configuration.

A set of scripts is available from inside that folder (provided by the `scripts` package) to make your work easier. Click here for a full description of these commands.

Note: You don't need to install or configure tools like webpack, Babel or ESLint yourself. They are preconfigured and hidden so that you can focus on coding.

For example, running the `start` script from inside the generated folder (`npm start`) would automatically start the build for development.

External Project Templates

Click here for information on External Project Templates

Contributing to this package

This is an individual package that's part of the Gutenberg project. The project is organized as a monorepo. It's made up of multiple self-contained software packages, each with a specific purpose.

The packages in this monorepo are published to npm and used by WordPress as well as other software projects.

To find out more about contributing to this package or Gutenberg as a whole, please read the project's main contributor guide.

□□□ #2
□ □□□ □□□ 2026-10-31 18:18:07 UTC
□ □□□ □□□ 2026-10-31 18:18:09 UTC