Synopsis

```
npm rebuild [[<@scope>/]<name>[@<version>] ...]

alias: rb
```

Description

This command runs the `npm build` command on the matched folders. This is useful when you install a new version of node, and must recompile all your C++ addons with the new binary. It is also useful when installing with `--ignore-scripts` and `--no-bin-links`, to explicitly choose which packages to build and/or link bins.

If one or more package names (and optionally version ranges) are provided, then only packages with a name and version matching one of the specifiers will be rebuilt.

Configuration

`global`

- Default: false
- Type: Boolean

Operates in "global" mode, so that packages are installed into the `prefix` folder instead of the current working directory. See [folders](#) for more on the differences in behavior.

- packages are installed into the `{prefix}/lib/node_modules` folder, instead of the current working directory.
- bin files are linked to `{prefix}/bin`
- man pages are linked to `{prefix}/share/man`

`bin-links`

- Default: true
- Type: Boolean

Tells npm to create symlinks (or `.cmd` shims on Windows) for package executables.

Set to false to have it not do this. This can be used to work around the fact that some file systems don't support symlinks, even on ostensibly Unix systems.

`foreground-scripts`

- Default: false
- Type: Boolean

Run all build scripts (ie, `preinstall`, `install`, and `postinstall`) scripts for installed packages in the foreground process, sharing standard input, output, and error with the main npm process.

Note that this will generally make installs run slower, and be much noisier, but can be useful for debugging.

`ignore-scripts`

- Default: false
- Type: Boolean

If true, npm does not run scripts specified in package.json files.

Note that commands explicitly intended to run a particular script, such as `npm start`, `npm stop`, `npm restart`, `npm test`, and `npm run-script` will still run their intended script if `ignore-scripts` is set, but they will *not* run any pre- or post-scripts.

`workspace`

- Default:
- Type: String (can be set multiple times)

Enable running a command in the context of the configured workspaces of the current project while filtering by running only the workspaces defined by this configuration option.

Valid values for the `workspace` config are either:

- Workspace names
- Path to a workspace directory
- Path to a parent workspace directory (will result in selecting all workspaces within that folder)

When set for the `npm init` command, this may be set to the folder of a workspace which does not yet exist, to create the folder and set it up as a brand new workspace within the project.

This value is not exported to the environment for child processes.

`workspaces`

- Default: null
- Type: null or Boolean

Set to true to run the command in the context of **all** configured workspaces.

Explicitly setting this to false will cause commands like `install` to ignore workspaces altogether. When not set explicitly:

- Commands that operate on the `node_modules` tree (install, update, etc.) will link workspaces into the `node_modules` folder. - Commands that do other things (test, exec, publish, etc.) will operate on the root project, *unless* one or more workspaces are specified in the `workspace` config.

This value is not exported to the environment for child processes.

`include-workspace-root`

- Default: false
- Type: Boolean

Include the workspace root when workspaces are enabled for a command.

When false, specifying individual workspaces via the `workspace` config, or all workspaces via the `workspaces` flag, will cause npm to operate only on the specified workspaces, and not on the root project.

See Also

- [npm install](#)