



*Full credit is given to the above companies including the OS that this PDF file was generated!*

## **Red Hat Enterprise Linux Release 9.2 Manual Pages on 'npm-publish.1' command**

**\$ man npm-publish.1**

NPM-PUBLISH(1)

NPM-PUBLISH(1)

NAME

npm-publish - Publish a package

Synopsis

npm publish <package-spec>

Description

Publishes a package to the registry so that it can be installed by name.

By default npm will publish to the public registry. This can be overridden by specifying a different default registry or using a npm help scope in the name, combined with a scope-configured registry (see package.json /configuring-npm/package-json).

A package is interpreted the same way as other commands (like npm install) and can be:

- ? a) a folder containing a program described by a package.json /configuring-npm/package-json file
- ? b) a gzipped tarball containing (a)
- ? c) a url that resolves to (b)
- ? d) a <name>@<version> that is published on the registry (see npm help registry) with (c)
- ? e) a <name>@<tag> (see npm help dist-tag) that points to (d)
- ? f) a <name> that has a "latest" tag satisfying (e)
- ? g) a <git remote url> that resolves to (a)

The publish will fail if the package name and version combination already exists in the specified registry.

Once a package is published with a given name and version, that specific name and version combination can never be used again, even if it is removed with `npm help unpublish`.

As of `npm@5`, both a `sha1sum` and an `integrity` field with a `sha512sum` of the tarball will be submitted to the registry during publication. Subsequent installs will use the strongest supported algorithm to verify downloads.

Similar to `--dry-run` see `npm help pack`, which figures out the files to be included and packs them into a tarball to be uploaded to the registry.

#### Files included in package

To see what will be included in your package, run `npx npm-packlist`.

All files are included by default, with the following exceptions:

• Certain files that are relevant to package installation and distribution are always included. For example, `package.json`, `README.md`, `LICENSE`, and so on.

• If there is a "files" list in `package.json` / `configuring-npm/packaging-npm`, then only the files specified will be included. (If directories are specified, then they will be walked recursively and their contents included, subject to the same ignore rules.)

• If there is a `.gitignore` or `.npmignore` file, then ignored files in that and all child directories will be excluded from the package. If both files exist, then the `.gitignore` is ignored, and only the `.npmignore` is used. `.npmignore` files follow the same pattern rules [https://git-scm.com/book/en/v2/Git-Basics-Recording-changes-to-the-repository#\\_ignoring\\_files](https://git-scm.com/book/en/v2/Git-Basics-Recording-changes-to-the-repository#_ignoring_files) as `.gitignore` files.

• If the file matches certain patterns, then it will never be included, unless explicitly added to the "files" list in `package.json`, or un-ignored with a `!` rule in a `.npmignore` or `.gitignore` file.

• Symbolic links are never included in npm packages.

See `npm help developers` for full details on what's included in the publication.

lished package, as well as details on how the package is built.

## Configuration

### tag

? Default: "latest"

? Type: String

If you ask npm to install a package and don't tell it a specific version,

then it will install the specified tag.

Also the tag that is added to the package@version specified by the npm tag command, if no explicit tag is given.

When used by the npm diff command, this is the tag used to fetch the tarball that will be compared with the local files by default.

### access

? Default: 'restricted' for scoped packages, 'public' for unscoped packages

? Type: null, "restricted", or "public"

When publishing scoped packages, the access level defaults to restricted. If you want your scoped package to be publicly viewable (and installable) set --access=public. The only valid values for access are public and restricted. Unscoped packages always have an access level of public.

Note: Using the --access flag on the npm publish command will only set the package access level on the initial publish of the package. Any subsequent npm publish commands using the --access flag will not have an effect to the access level. To make changes to the access level after the initial publish use npm access.

### dry-run

? Default: false

? Type: Boolean

Indicates that you don't want npm to make any changes and that it should only report what it would have done. This can be passed into any of the commands that modify your local installation, eg, install, update, dedupe, uninstall, as well as pack and publish.

Note: This is NOT honored by other network related commands, eg

dist-tags, owner, etc.

## otp

? Default: null

? Type: null or String

This is a one-time password from a two-factor authenticator. It's needed when publishing or changing package permissions with npm access.

If not set, and a registry response fails with a challenge for a one-time password, npm will prompt on the command line for one.

## 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.

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

#### See Also

? npm help "package spec"

? npm-packlist package <http://npm.im/npm-packlist>

? npm help registry

? npm help scope

? npm help adduser

? npm help owner

? npm help deprecate

? npm help dist-tag

? npm help pack

? npm help profile

February 2023

NPM-PUBLISH(1)