Synopsis

```
npm run-script <command> [-- <args>]

aliases: run, rum, urn
```

Description

This runs an arbitrary command from a package's `"scripts"` object. If no `"command"` is provided, it will list the available scripts.

`run[-script]` is used by the test, start, restart, and stop commands, but can be called directly, as well. When the scripts in the package are printed out, they're separated into lifecycle (test, start, restart) and directly-run scripts.

Any positional arguments are passed to the specified script. Use `--` to pass `-`-prefixed flags and options which would otherwise be parsed by npm.

For example:

```
npm run test -- --grep="pattern"
```

The arguments will only be passed to the script specified after `npm run` and not to any `pre` or `post` script.

The `env` script is a special built-in command that can be used to list environment variables that will be available to the script at runtime. If an "env" command is defined in your package, it will take precedence over the built-in.

In addition to the shell's pre-existing `PATH`, `npm run` adds `node_modules/.bin` to the `PATH` provided to scripts. Any binaries provided by locally-installed dependencies can be used without the `node_modules/.bin` prefix. For example, if there is a `devDependency` on `tap` in your package, you should write:

```
"scripts": {"test": "tap test/*.js"}
```

instead of

```
"scripts": {"test": "node_modules/.bin/tap test/*.js"}
```

The actual shell your script is run within is platform dependent. By default, on Unix-like systems it is the `/bin/sh` command, on Windows it is `cmd.exe`. The actual shell referred to by `/bin/sh` also depends on the system. You can customize the shell with the `script-shell` configuration.

Scripts are run from the root of the package folder, regardless of what the current working directory is when `npm run` is called. If you want your script to use different behavior based on what subdirectory you're in, you can use the `INIT_CWD` environment variable, which holds the full path you were in when you ran `npm run`.

`npm run` sets the `NODE` environment variable to the `node` executable with which `npm` is executed.

If you try to run a script without having a `node_modules` directory and it fails, you will be given a warning to run `npm install`, just in case you've forgotten.

Workspaces support

You may use the `workspace` or `workspaces` configs in order to run an arbitrary command from a package's `"scripts"` object in the context of the specified workspaces. If no `"command"` is provided, it will list the available scripts for each of these configured workspaces.

Given a project with configured workspaces, e.g:

```
.
+-- package.json
`-- packages
    +-- a
    |   `-- package.json
    +-- b
    |   `-- package.json
    `-- c
        `-- package.json
```

Assuming the workspace configuration is properly set up at the root level `package.json` file. e.g:

```
{
    "workspaces": [ "./packages/*" ]
}
```

And that each of the configured workspaces has a configured `test` script, we can run tests in all of them using the `workspaces` config:

```
npm test --workspaces
```

Filtering workspaces

It's also possible to run a script in a single workspace using the `workspace` config along with a name or directory path:

```
npm test --workspace=a
```

The `workspace` config can also be specified multiple times in order to run a specific script in the context of multiple workspaces. When defining values for the `workspace` config in the command line, it also possible to use `-w` as a shorthand, e.g:

```
npm test -w a -w b
```

This last command will run `test` in both `./packages/a` and `./packages/b` packages.

Configuration

`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.

`if-present`

- Default: false
- Type: Boolean

If true, npm will not exit with an error code when `run-script` is invoked for a script that isn't defined in the `scripts` section of `package.json` . This option can be used when it's desirable to optionally run a script when it's present and fail if the script fails. This is useful, for example, when running scripts that may only apply for some builds in an otherwise generic CI setup.

`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.

`script-shell`

- Default: '/bin/sh' on POSIX systems, 'cmd.exe' on Windows

- Type: null or String

The shell to use for scripts run with the `npm exec`, `npm run` and `npm init <pkg>` commands.

See Also

- [npm scripts](#)
- [npm test](#)
- [npm start](#)
- [npm restart](#)
- [npm stop](#)
- [npm config](#)
- [npm workspaces](#)