

# Package ‘tinypkgr’

April 22, 2026

**Title** Minimal R Package Development Utilities

**Version** 0.2.1

**Description** Lightweight wrappers around 'R CMD INSTALL', 'R CMD check', 'R CMD build', 'win-builder' uploads, and 'CRAN' submission. Provides functions for installing, loading, checking, building, and submitting R packages with minimal dependencies (only 'curl' for uploads). Background on R package development is in Wickham and Bryan (2023, ISBN:9781098134945), ``Writing R Extensions" <<https://cran.r-project.org/doc/manuals/R-exts.html>>, and the 'CRAN' Repository Policy <<https://cran.r-project.org/web/packages/policies.html>>.

**License** GPL-3

**URL** <https://github.com/cornball-ai/tinypkgr>

**BugReports** <https://github.com/cornball-ai/tinypkgr/issues>

**Encoding** UTF-8

**Imports** curl

**Suggests** tinyrox, tinytest

**NeedsCompilation** no

**Author** Troy Hernandez [aut, cre] (ORCID: <<https://orcid.org/0009-0005-4248-604X>>), cornball.ai [cph]

**Maintainer** Troy Hernandez <troy@cornball.ai>

**Repository** CRAN

**Date/Publication** 2026-04-22 08:50:02 UTC

## Contents

|                           |   |
|---------------------------|---|
| build . . . . .           | 2 |
| check . . . . .           | 3 |
| check_win_devel . . . . . | 4 |
| install . . . . .         | 4 |
| load_all . . . . .        | 5 |

|                             |   |
|-----------------------------|---|
| maintainer . . . . .        | 6 |
| reload . . . . .            | 7 |
| submit_cran . . . . .       | 8 |
| use_github_action . . . . . | 8 |
| use_version . . . . .       | 9 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>11</b> |
|--------------|-----------|

---

|       |                              |
|-------|------------------------------|
| build | <i>Build Package Tarball</i> |
|-------|------------------------------|

---

## Description

Builds a source package tarball suitable for CRAN submission.

## Usage

```
build(path = ".", dest_dir = tools::R_user_dir("tinypkgr", "cache"))
```

## Arguments

|          |  |
|----------|--|
| path     | Path to package root directory.  |
| dest_dir | Directory to place the tarball. Defaults to the per-package user cache dir ('tools::R_user_dir("tinypkgr", "cache)'), which is CRAN's recommended location for package-owned output and persists across sessions. Pass an explicit path to place the tarball somewhere else. |

## Value

Path to the built tarball (invisibly).

## Examples

```
# Scaffold a throwaway package in tempdir() and build a source tarball.
pkg <- file.path(tempdir(), "buildpkg")
dir.create(file.path(pkg, "R"), recursive = TRUE, showWarnings = FALSE)
writeLines(c(
  "Package: buildpkg",
  "Title: Example",
  "Version: 0.0.1",
  "Authors@R: person('A', 'B', email = 'a@b.com', role = c('aut','cre'))",
  "Description: Example.",
  "License: GPL-3"
), file.path(pkg, "DESCRIPTION"))
writeLines("add <- function(x, y) x + y", file.path(pkg, "R", "add.R"))

out <- file.path(tempdir(), "tarballs")
dir.create(out, showWarnings = FALSE)
tarball <- build(pkg, dest_dir = out)
file.exists(tarball)
```

```
unlink(pkg, recursive = TRUE)
unlink(out, recursive = TRUE)
```

---

|       |                      |
|-------|----------------------|
| check | <i>Check Package</i> |
|-------|----------------------|

---

### Description

Runs R CMD build and R CMD check on the package.

### Usage

```
check(path = ".", args = c("--as-cran", "--no-manual"),
      error_on = c("warning", "error", "note"))
```

### Arguments

|          |   |
|----------|---|
| path     | Path to package root directory.   |
| args     | Character vector of additional arguments to pass to R CMD check. Default includes "--as-cran" and "--no-manual".        |
| error_on | Severity level that causes an error: "error", "warning", or "note". Default is "warning" (fails on errors or warnings). |

### Value

TRUE if check passes, FALSE otherwise (invisibly). Also throws an error if check fails at or above error\_on level.

### Examples

```
# Runs 'R CMD build' + 'R CMD check' (typically tens of seconds).
# Intermediate files go under tempdir(), so nothing is written to the
# caller's working directory. Wrapped in if(interactive()) so the
# example is shown to users but not executed during R CMD check.

if (interactive()) {
  check()
  check(error_on = "error")
  check(args = c("--as-cran", "--no-manual"))
}
```

check\_win\_devel      *Check Package on Windows via win-builder*

---

**Description**

Uploads package to win-builder.r-project.org for testing on Windows. Results are emailed to the package maintainer.

**Usage**

```
check_win_devel(path = ".", r_version = c("devel", "release", "oldrelease"))
```

**Arguments**

path                  Path to package root directory.  
r\_version             Which R version to test: "devel", "release", or "oldrelease". Default is "devel".

**Value**

TRUE if upload succeeded (invisibly).

**Examples**

```
# Uploads a tarball to the public 'win-builder' FTP server. Wrapped  
# in if(interactive()) so CRAN's automated checks never touch the  
# network.  
  
if (interactive()) {  
  check_win_devel()  
  check_win_devel(r_version = "release")  
}
```

---

install                  *Install Package*

---

**Description**

Wrapper around R CMD INSTALL with quiet mode option.

**Usage**

```
install(path = ".", quiet = TRUE)
```

**Arguments**

path            Path to package root directory.  
quiet           Logical. Suppress output except errors? Default TRUE.

**Value**

TRUE if successful, FALSE otherwise (invisibly).

**Examples**

```
# Calls 'R CMD INSTALL', which writes to the user's R library.  
# Wrapped in if(interactive()) so CRAN's automated checks never  
# mutate the library.  
  
if (interactive()) {  
  install()  
  install(quiet = FALSE)  
}
```

---

load\_all

*Load All Package Code*

---

**Description**

Sources all R files in a package for interactive development, without requiring a full install.

**Usage**

```
load_all(path = ".", env = new.env(parent = globalenv()), quiet = TRUE)
```

**Arguments**

path            Path to package root directory.  
env             Environment to source files into. Defaults to a fresh environment whose parent is the global environment.  
quiet           Logical. Suppress file sourcing messages? Default TRUE.

**Value**

The environment into which files were sourced (invisibly). Does not modify the search path. If you want search-path behavior, call `attach()` yourself:

```
attach(tinypkgr::load_all(), name = "tinypkgr:mypkg")
```

**See Also**

[kitten](#) for scaffolding a new package.



```
  "Description: Example.",
  "License: GPL-3"
), file.path(pkg, "DESCRIPTION"))

maintainer(pkg)

unlink(pkg, recursive = TRUE)
```

---

reload *Reload an Installed Package*

---

### Description

Unloads a package if loaded, reinstalls it, and loads it again. Convenience function for the install-reload cycle during development.

### Usage

```
reload(path = ".", document = FALSE, quiet = TRUE)
```

### Arguments

|          |   |
|----------|---|
| path     | Path to package root directory.   |
| document | If TRUE and tinyrox is available, run tinyrox::document() before installing. Default FALSE. |
| quiet    | Logical. Suppress install output? Default TRUE.   |

### Value

TRUE if successful (invisibly).

### Examples

```
# Calls install() under the hood, which writes to the user's R
# library. Wrapped in if(interactive()) so checks never mutate it.

if (interactive()) {
  reload()
  reload(document = TRUE)
}
```

---

|             |                               |
|-------------|-------------------------------|
| submit_cran | <i>Submit Package to CRAN</i> |
|-------------|-------------------------------|

---

**Description**

Uploads package to CRAN for review. You will receive a confirmation email that must be clicked to complete the submission.

**Usage**

```
submit_cran(path = ".", comments = "cran-comments.md")
```

**Arguments**

|          |   |
|----------|---|
| path     | Path to package root directory.                 |
| comments | Path to cran-comments.md file, or NULL to skip. |

**Value**

TRUE if submission succeeded (invisibly).

**Examples**

```
# Uploads to CRAN and prompts interactively for confirmation.  
# Wrapped in if(interactive()) so CRAN's automated checks never  
# attempt the upload.  
  
if (interactive()) {  
  submit_cran()  
}
```

---

|                   |   |
|-------------------|---|
| use_github_action | <i>Add a GitHub Actions CI Workflow</i> |
|-------------------|---|

---

**Description**

Writes `github/workflows/ci.yaml` using the r-ci template (Ubuntu and macOS, via `eddelbuettel/github-actions/r-ci@master`). Adds `^\.github$` to `.Rbuildignore` if not already present.

**Usage**

```
use_github_action(path)
```

**Arguments**

`path` Path to package root directory. Required; no default is provided because this function writes files under ‘path’, and CRAN Repository Policy forbids defaulting write paths to the user’s home filespace (which includes ‘getwd()’).

**Value**

Path to the created YAML file (invisibly).

**Examples**

```
# Scaffold a throwaway package in tempdir() and add the workflow.
pkg <- file.path(tempdir(), "ghapkg_example")
dir.create(pkg, showWarnings = FALSE)
writeLines(c(
  "Package: ghapkg",
  "Title: Example",
  "Version: 0.0.1",
  "Authors@R: person('A', 'B', email = 'a@b.com', role = c('aut','cre'))",
  "Description: Example.",
  "License: GPL-3"
), file.path(pkg, "DESCRIPTION"))

yaml <- use_github_action(path = pkg)
file.exists(yaml)

unlink(pkg, recursive = TRUE)
```

---

use\_version

*Bump Package Version*

---

**Description**

Increments the Version field in DESCRIPTION and prepends a new section header to NEWS.md (if present) so the two never drift apart.

**Usage**

```
use_version(which = c("patch", "minor", "major", "dev"), path)
```

**Arguments**

`which` Which component to bump: "patch" (0.2.0 -> 0.2.1), "minor" (0.2.0 -> 0.3.0), "major" (0.2.0 -> 1.0.0), or "dev" (0.2.0 -> 0.2.0.1, or 0.2.0.1 -> 0.2.0.2).

`path` Path to package root directory. Required; no default is provided because this function edits files in ‘path’, and CRAN Repository Policy forbids defaulting write paths to the user’s home filespace (which includes ‘getwd()’).

**Value**

The new version string (invisibly).

**Examples**

```
# Scaffold a throwaway package in tempdir() and bump its version.
pkg <- file.path(tempdir(), "vxpkg")
dir.create(pkg, showWarnings = FALSE)
writeLines(c(
  "Package: vxpkg",
  "Title: Example",
  "Version: 0.1.0",
  "Authors@R: person('A', 'B', email = 'a@b.com', role = c('aut','cre'))",
  "Description: Example.",
  "License: GPL-3"
), file.path(pkg, "DESCRIPTION"))

use_version("patch", path = pkg)
read.dcf(file.path(pkg, "DESCRIPTION"))[1, "Version"]

unlink(pkg, recursive = TRUE)
```

# Index

build, [2](#)

check, [3](#)

check\_win\_devel, [4](#)

install, [4](#)

kitten, [5](#)

load\_all, [5](#)

maintainer, [6](#)

reload, [7](#)

submit\_cran, [8](#)

use\_github\_action, [8](#)

use\_version, [9](#)