Awesome R Package Development Tools

Indrajeet Patil

Table of contents

A curated list of awesome tools to assist development in R programming language.

⚠ Contributing

If you wish to contribute to this list, please read our Contributing Guidelines.
Important: Only edit `README.qmd` - do NOT edit `README.md`, `README.pdf`, or `index.html` as these are automatically generated by our GitHub Actions workflow (on push to main, not on PRs).

What is included?

- Only *tools* helpful for package development are included, and not other resources (e.g. books).

- All relevant tools are included, irrespective of their availability on CRAN/Bioconductor.
- Tools which are part of publicly archived/retired GitHub repositories are *not* included.

For detailed contribution instructions, please see our Contributing Guidelines.

1 Code of Conduct

Please note that the `awesome-r-pkgtools` project is released with a Contributor Code of Conduct. By contributing to this project, you agree to abide by its terms.

2 Swiss army knives

Tools useful across all stages of package development (some of these are meta-packages and their component packages are also included in respective sections for the sake of completeness), irrespective of whether the package is meant to be submitted to CRAN or Bioconductor.

- `{usethis}`
- `{devtools}`
- `{biocthis}`
- `{packager}`
- `{pacs}`
- `{pkgmaker}`

3 Package templates

3.1 Generic

- `{pkgkitten}` (useful for creating new packages for R)
- `{rcompendium}` (to make the creation of R package/research compendium easier)
- `{r.pkg.template}` (an opinionated R package template with CI/CD built-in)
- `{skeletor}` (An R Package Skeleton Generator)

3.2 RMarkdown-based

- `{fusen}` (to build a package from RMarkdown files)
- `{litr}` (to write a complete R package in a single R markdown document)

3.3 Shiny

- `{golem}` (framework for building shiny applications)
- `{leprechaun}` (leaner framework for building shiny applications)
- `{rhino}` (a framework to build high quality, enterprise-grade Shiny apps at speed)

3.4 Meta-packages

- `{pkgverse}` (for package meta-verse)
- `{metamakr}` (for package meta-verse)

4 Naming things

- `{available}` (to check if a package name is available to use)
- `{collidr}` (to check for namespace collisions)
- `{changer}` (to change the name of an existing R package)

5 Working with package components

- `{rprojroot}` (accessing files w.r.t. package root directory)
- `{desc}` (manipulating `DESCRIPTION` files)
- `{withr}` (to manage package side effects by safely and temporarily modifying global states)
- `{pkgload}` (to simulate the process of installing and loading a package)
- `{pkgbuild}` (to find tools needed to build packages)

6 Package configuration

- `{config}` (to manage environment specific configuration values)
- `{dotenv}` (to load environment variables from `.env` files)
- `{options}` (provides simple mechanisms for defining and interpreting package options)
- `{potions}` (to update and retrieve options, either in the workspace or during package development, without overwriting global options)

7 Package management tools

- `{pkgcache}` (to cache 'CRAN'-like metadata and packages)

8 Documentation

8.1 Manual

- `{roxygen2}` (to generate R package documentation from inline R comments)
- `{Rd2roxygen}` (in case you inherit a project where documentation was not written using `{roxygen2}`)
- `{rdoxygen}` (to create Doxygen documentation for R package C++ code)
- `{roxyglobals}` (to generate global variables with `{roxygen2}` documentation)
- `{sinew}` (generate `{roxygen2}` skeletons)
- `{autoimport}` (to automatically generate @importFrom roxygen tags from R files)
- `{roclang}` (helpers for diffusing content across function documentation)
- `{Rdpack}` (for inserting references, figures, and evaluated examples in Rd docs)
- `{roxygen2md}` (to allow Markdown syntax usage in `{roxygen2}` documentation)
- `{rd2markdown}` (to convert `.Rd` package documentation files into markdown files)
- `{rd2list}` (converts Rd docs to a human-readable list)
- `{pasteAsComment}` (RStudio addin for pasting copied code as roxygen comment)
- `roxygen2Comment` (Rstudio addin for adding and remove `{roxygen2}` comment)

8.2 Math rendering in HTML/PDF manual

- `{katex}` (to convert latex math expressions to HTML for use in package manual pages)
- `{mathjaxr}` (provides 'MathJax' and macros to enable its use within `Rd` files for rendering equations in the HTML help files)
- `{mathml}` (translates R expressions to 'MathML' or 'MathJax' so that they can be rendered in HTML manual and Shiny apps)

8.3 Vignettes

- `{knitr}` (a general-purpose tool for dynamic report generation to be used as a vignette builder for R package vignettes)
- `{rmarkdown}` (to convert R Markdown documents to a variety of formats)
- `{quarto}` (provides R interface to frequently used operations in the Quarto CLI)
- `{R.rsp}` (for incorporating static and dynamic vignettes)
- `{RmdConcord}` (to provide support for concordances in R Markdown files)
- `{prettydoc}` (creates lightweight yet pretty vignettes)
- `{readme2vignette}` (to convert README to vignette during package installation)

8.4 Tutorials

- `{learnr}` (to turn any R Markdown document into an interactive tutorial)

8.5 Website

- `{pkgdown}` (static website for package documentation)
- `{gitdown}` (software changes as a gitbook)
- `{altdoc}` (use `docute`, `docsify`, or `MkDocs` to create a static website for package documentation)

8.6 Translation

- `{potools}` (for translating messages and checking the "health" of the messaging corpus)

8.7 Lifecycle

- `{lifecycle}` (to manage the life cycle of exported functions)

8.8 Badges and stickers

- `{badger}` (to query information and generate badges for use in `README`)
- `{badgen}` (provides bindings to `badgen` to generate beautiful 'svg' badges in R without internet access)
- `{hexSticker}` (helper functions for creating reproducible hexagon sticker purely in R)
- `{hexFinder}` (to scavenge the web for possible hex logos for packages)
- `hexwall` (to create a wall of hexstickers)

8.9 Presentation

- `{xaringan}` (an RMarkdown output format for `remark.js` slides)

8.10 Book

- `{bookdown}` (authoring framework for books and technical documents with R Markdown)

8.11 Change log and versioning

- `{fledge}` (to streamline the process of updating change logs and versioning R packages developed in git repositories)
- `{newsmd}` (utilities to add updates to the `NEWS.md` file)
- `{autonewsmd}` (to auto-generate change log using conventional commits)

9 Documentation quality

- `{docreview}` (to check quality of docs)
- `{spelling}` (to check for spelling mistakes)
- `{gramr}` (for grammar suggestions)

10 Observability and Monitoring

- {otelsdk} (R SDK for OpenTelemetry)
- {logger} (provides a flexible and extensible logging framework for R)
- {loggit} (effortless newline-delimited JSON logger, with two primary log-writing interfaces)
- {log4r} (logging in R based on the widely-emulated 'log4j' system and etymology)
- {lgr} (a flexible, feature-rich yet light-weight logging framework based on 'R6' classes)
- {rsyslog} (write messages to the 'syslog' system logger API)
- {logging} (pure R implementation of the ubiquitous 'log4j' package)
- {lumberjack} (to log changes in data)

11 Unit testing

11.1 Generic R Packages

- {testthat} (a testing framework for R that is easy to learn and use; also provides snapshot testing)
- {patrick} (for parameterized unit testing with {testthat})
- {testdat} (a family of functions and reporting tools focused on checking of data)
- {tinytest} (zero-dependency unit testing framework that installs tests with the package)
- {tinysnapshot} (snapshots for unit tests using the {tinytest} framework)
- {tinytest2JUnit} (to convert {tinytest} output to JUnit XML needed by CI/CD)
- {checkmate.tinytest} (additional expectations for {tinytest} framework)
- {RUnit} (a standard unit testing framework, with additional code inspection and report generation tools)
- {testit} (a simple package for testing R packages)
- {realtest} (a framework unit testing that distinguishes between expected, acceptable, current, fallback, ideal, or regressive behaviours)
- {roxytest} (to inline {testthat} tests with {roxygen2})
- {doctest} (to write {testthat} tests by adding {roxygen2} tags)

- {exampletestr} (tests based on package examples)
- {roxut} (to write the unit tests in the same file as the function)
- {unitizer} (simplifies regression tests by comparing objects produced by test code with earlier versions of those same objects)
- {r-hedgehog} (property based testing)
- {muttest} (automatic mutation testing of R packages)
- {cucumber} (an implementation of the Cucumber testing framework in R)
- {quickcheck} (provides property-based testing in {testthat} framework)

11.2 Web/database applications

- {httptest}/{httptest2} (a test environment for HTTP requests in R)
- {webfakes} (to fake web apps for HTTP testing)
- {vcr} (to record HTTP requests and responses on disk and replay them for the unit tests)
- {dittodb} (makes testing against databases easy)

11.3 Visual regression testing

- {vdiffr} (for visual regression testing with {testthat})
- {gdiff} (for performing graphical difference different package or R versions)

11.4 Mock testing

- {mockthat} (provides a way to mock package function for unit testing, while coping with S3 dispatch)
- {mockr} (provides a way to mock package function for unit testing)
- {mockery} (provides a way to mock package function for unit testing and can be used with any testing framework)

11.5 Mutation testing

- {mutant} (mutation testing for R)

11.6 Markdown documents

- `{pandoc}` (to check Markdown documents across various version of Pandoc)

11.7 Shiny applications

- `{shinytest}` (testing Shiny apps)
- `{shinytest2}` (testing Shiny apps using a headless Chromium web browser)
- `{shinyloadtest}` (to load test deployed Shiny apps)

11.8 Helpers for testing frameworks

- `{testthis}` (RStudio addins for working with files that contain tests)
- `{xpectr}` (builds unit tests with the `{testthat}` package by providing tools for generating expectations)
- `{testdown}` (turn `{testthat}` results into a `{bookdown}` project)
- `{ttdo}` (provides 'diff'-style comparison of R objects for `{tinytest}` framework)

12 Code/Document Formatting

- `{styler}` (to format code according to a style guide)
- `{stylermd}` (to format text in Markdown documents)
- `{formatR}` (to format R source code)
- `{RFormatter}` (extension of `{formatR}` with slightly improved heuristics)
- `{grkstyle}` (extension package for `{styler}` that supports author's personal code style preferences)
- `{codegrip}` (addin for RStudio IDE to reshape R code and navigate across syntactic constructs)
- `{BiocStyle}` (provides standard formatting styles for Bioconductor PDF and HTML documents)
- `AlignAssign` (RStudio addin that aligns the assignment operators within a highlighted area)
- `{snakecase}` (helpful for having consistent case while naming objects in the package)

- `{dotInternals}` (to distinguish non-exported package functions by prepending their names with a dot)

13 Code analysis

13.1 General

- `{codetools}` (code analysis tools for R)
- `{goodpractice}` (Swiss army knife for good practices)
- `{inteRgrate}` (provides an opinionated set of rules for R package development)
- `{checklist}` (to provide an elaborate and strict set of checks for R packages and R code)
- `{pkgcheck}` (checks if package follows good practices recommended for packages in the `rOpenSci` ecosystem)
- `{pkgstats}` (provides metrics of R Packages)
- `{rchk}` (provides several bug-finding tools that look for memory protection errors in C source code using R API)
- `{sourcetools}` (tools for reading, tokenizing, and parsing R code)
- `{precommit}` (git hooks for common tasks like formatting files, spell checking, etc.)

13.2 Code review

- `{PaRe}` (reviews other packages during code review by looking at their dependencies, code style, code complexity, and how internally defined functions interact with one another)

13.3 Code coverage

- `{covr}` (to compute code coverage)
- `{covrpage}` (to include summary `README` of code coverage and more detailed information about tests)
- `{covtracer}` (provides tools for contextualizing tests)

13.4 Code quality

- `{lintr}` (static code analysis)
- `{flir}` (to fix lints found by `{lintr}`)
- `{roxylint}` (to lint `{roxygen2}`-generated documentation)
- `{checkglobals}` (to check R-packages for globals and imports)
- `{CodeDepends}` (analysis of R code for reproducible research and code view)
- `{adaptalint}` (infer code style from one package and use it to check another)
- `{box.linters}` (linters for `{box}` modules)
- `{roger}` (provides tools for grading the coding style and documentation of R scripts)
- `{cleanr}` (tests code for some of the most common code layout flaws)

13.5 Code complexity

- `{cyclocomp}` (to index the complexity of a function)
- `{pkgGraphR}` (to visualize the relationship between functions in an R package)

13.6 Code similarity

- `{dupree}` (identifies code blocks that have a high level of similarity within a set of R files)
- `{rscc}` (provides source code similarity evaluation by variable/function names)
- `{SimilaR}` (quantifies the similarity of the code-base of R functions by means of program dependence graphs)

13.7 Compiled code

- `{memtools}` (to solve memory leaks)
- `{sanitizers}` (to test for memory violations and other undefined behaviour)
- `{cppcheckR}` (to check `C` and `C++` code using `Cppcheck`)

13.8 JavaScript code

- `{jshintr}` (to run JSHint for static code analysis for JavaScript code included in the package)

13.9 Lines of code

- `{cloc}` (counts blank lines, comment lines, and physical lines of source code in source files)

14 Refactoring

- `{refactor}` (to check speed and performance of both the original and refactored version of code)

15 Code performance

15.1 Benchmarking

- `{bench}` (provides high precision benchmarks for R expressions)
- `{microbenchmark}` (infrastructure to accurately measure and compare the execution time of R expressions)
- `{tictoc}` (functions for timing R scripts)
- `{touchstone}` (to benchmark pull requests)
- `{benchmarkme}` (to crowd-source system benchmarking)
- `{comparer}` (to compare the results of different code chunks)

15.2 Profiling

- `{profvis}` (to profile and visualize profiling data)
- `{proffer}` (to create friendlier, faster visualizations for profiling data)
- `{jointprof}` (to profile packages with native code in C, C++, Fortran, etc.)
- `{xrprof}` (an external sampling profiler)

16 Reproducible Environments

16.1 Package management

- `{renv}` (to create project-local environments)
- `{rix}` (to create reproducible data science environments using the Nix package manager)
- `{bspm}` (to enable binary package installations via Linux distribution's package manager)
- `{rspm}` (to access Posit Public Package Manager for binary package installations on Linux)
- `{groundhogr}` (to load packages and their dependencies as available on chosen date on CRAN)

16.2 Containerization

- `{containerit}` (to package R script/session/workspace and all dependencies as a `Docker` container by generating a suitable `Dockerfile`)
- `{dockerfiler}` (to generate `Dockerfile` for R projects)
- `{pracpac}` (a `{usethis}`-like interface to create Docker images from R packages under development)
- `{usethat}` (to automate analytic project setup tasks)

17 Dependency Management

- `{pkgdepends}` (to find recursive dependencies of from various sources)
- `{deepdep}` (to visualize and explore package dependencies)
- `{itdepends}` (to assess usage, measure weights, visualize proportions, and assist removal of dependencies)
- `{DependenciesGraphs}` (to visualize package dependencies)
- `{DependencyReviewer}` (to investigate packages during code review by looking at their dependencies)
- `{pkgdepR}` (to visualize dependencies between functions for a group of R packages)
- `{deps}` (to manage source code dependencies by decorating R code with roxygen-style comments)

- `{pkgnet}` (to build a graph representation of a package and its dependencies)
- `{functiondepends}` (to find functions in an unstructured directory and explore their dependencies)
- `{pkgndep}` (checks the heaviness of the packages used)
- `{attachment}` (to deal with package dependencies during package development)

18 CRAN/Bioconductor checks

- `{rcmdcheck}` (to run `R CMD check` form R programmatically)
- `{BiocCheck}` (to run Bioconductor-specific package checks)
- `{rhub}` (to run `R CMD check` on CRAN architectures)
- `{checked}` (systematically run `R CMD check` against multiple packages)
- `{checkhelper}` (to help avoid problems with CRAN submissions)
- `{extrachecks}` (to run some additional CRAN checks)
- `{foghorn}` (to check for results and submission portal status)
- `{urlchecker}` (to checks for URL rot)

19 Usage

- `{cranlogs}` (for computing CRAN download counts)
- `{packageRank}` (for visualizing CRAN download counts)
- `{Visualize.CRAN.Downloads}` (to visualize CRAN downloads)
- `{dlstats}` (provides download statistics for packages)

20 CI/CD

CI/CD: continuous integration and either continuous delivery or continuous deployment

- `actions` (provides GitHub Actions relevant for R)
- `{gha}` (Useful functions for GitHub Actions)
- `actions-sync` (to manage GitHub Actions workflows across repositories)
- `{rworkflows}` (GitHub Actions to automates testing, documentation website building, and containerized deployment)
- `AzureR` (a family of packages for working with Azure from R)
- `r-appveyor` (for AppVeyor)
- `{tic}` (for Circle CI and GitHub Actions)
- `{circle}` (for Circle CI)
- `{jenkins}` (for Jenkins CI)
- `{cronR}` (to schedule R scripts/processes with the cron scheduler)

21 Security/Privacy

- `{gpg}` (GNU privacy guard for R)
- `{oysteR}` (to secure package against insecure dependencies)

22 Build systems

- `{fakemake}` (to mock Unix Make build system in case it is unavailable)

23 Debugging

- `{debugme}` (provides helpers to specify debug messages as special string constants, and control debugging of packages via environment variables)
- `{debugr}` (tools to print out the value of R objects/expressions while running an R script)
- `{winch}` (provides stack traces for call chains that cross between R and C/C++ function calls)
- `{flow}` (to visualize as flow diagrams the logic of functions, expressions, or scripts, which can ease debugging)
- `{boomer}` (provides debugging tools to inspect the intermediate steps of a call)

24 Input validation

24.1 Function argument validation

- `{chk}` (to check user-supplied function arguments)
- `{checkmate}` (fast and versatile argument checks)
- `{assertthat}` (to declare the pre and post conditions that you code should satisfy and to produce friendly error messages)
- `{assertive}` (provides readable check functions to ensure code integrity)
- `{valaddin}` (functional input validation)
- `{dreamerr}` (to check the arguments passed to a function and to offer informative error messages)
- `{erify}` (to check arguments and generate readable error messages)

24.2 Data validation

- `{assertr}` (to verify assumptions about data early)
- `{ensurer}` (to ensure values are as expected at runtime)
- `{validate}` (to check whether data lives up to expectations based on the domain-specific knowledge)

25 Package metadata

- `{codemetar}` (provides utilities to generate, parse, and modify `codemeta.jsonld` files automatically for R packages), or `{codemeta}` (a leaner version of `{codemetar}`)
- `{cffr}` (provides utilities to generate, parse, modify and validate `CITATION.cff` files automatically for R packages)
- `{citation}` (creates `CITATION.cff` from R package metadata)
- `{pkgapi}` (to create the map of function calls in a package)
- `{riskmetric}` (provides a collection of risk metrics to evaluate the quality of R packages)
- `{packagemetrics}` (for comparing among packages)
- `{devtoolbox}` (to create a summary report for R package and to extract dependency statistics in a tidy data frame)
- `{pkgattrs}` (useful for getting information on the contents of any R package)
- `{foreman}` (for unpacking, interrogating and subsetting R packages)
- `{sessioninfo}` (to include R session information)

26 Reverse dependency checks

- `{revdepcheck}` (for automated, isolated, reverse dependency checking)
- `{xfun}` (specifically, `xfun::rev_check()`)

27 Gratitude

To thank the contributors or maintainers of packages you rely on.

- `{thankr}` (to find out who maintains the packages you are using)
- `{allcontributors}` (to help acknowledge all contributions)

28 Integration with other languages

28.1 C++

- {Rcpp}
- {cpp11}

28.2 Fortran

- {RFI}

28.3 Python

- {reticulate}

28.4 Rust

- {rextendr}
- {savvy}
- {cargo}
- {hellorust}

28.5 .NET Framework

- {rClr}

28.6 JavaScript/HTML/CSS

- {htmltools}
- {packer}

28.7 Julia

- {JuliaCall}

29 Upkeep

- `{TODOr}` (RStudio addin to list things that you need to do or change)

30 Sundry

- `{lazyData}` (supplies a lazy data loading for packages with datasets that do not provide `LazyData: true`)
- `{pkglite}` (tools to represent and exchange R package source code as text files)
- `{gpttools}` (RStudio addin that allows using chatGPT to automate writing documentation, tests, etc.)
- `{rfold}` (to work with many R folders within an R package)
- `{many}` (to create R packages from many directories)
- `{prefixer}` (prefix function with their namespace)
- `{onetime}` (for package authors to run code only once for a given user on a given computer)
- `{rstudioapi}` (to conditionally access the RStudio API from CRAN packages)
- `{rcheology}` (to access data on base packages for previous versions of R)
- `{gitignore}` (to fetch gitignore templates)
- `{DIZutils}` (helpers for packages dealing with database connections)
- `{dang}` (Miscellaneous utilities for CRAN packages)

31 Session information

Session details

```
- Session info ---------------------------------------------------------------
 setting  value
 version  R version 4.5.2 (2025-10-31)
 os       Ubuntu 24.04.3 LTS
 system   x86_64, linux-gnu
 ui       X11
 language (EN)
 collate  C.UTF-8
 ctype    C.UTF-8
 tz       UTC
 date     2026-02-07
 pandoc   3.9 @ /opt/hostedtoolcache/pandoc/3.9/x64/ (via rmarkdown)
 quarto   1.9.19 @ /usr/local/bin/quarto

- Packages -------------------------------------------------------------------
 package     * version date (UTC) lib source
 base        * 4.5.2   2025-10-31 [3] local
 cli           3.6.5   2025-04-23 [1] RSPM
 clipr         0.8.0   2022-02-22 [1] RSPM
 compiler      4.5.2   2025-10-31 [3] local
 datasets    * 4.5.2   2025-10-31 [3] local
 desc          1.4.3   2023-12-10 [1] RSPM
 details     * 0.4.0   2025-02-09 [1] RSPM
 digest        0.6.39  2025-11-19 [1] RSPM
 evaluate      1.0.5   2025-08-27 [1] RSPM
 fastmap       1.2.0   2024-05-15 [1] RSPM
 graphics    * 4.5.2   2025-10-31 [3] local
 grDevices   * 4.5.2   2025-10-31 [3] local
 grid          4.5.2   2025-10-31 [3] local
 htmltools     0.5.9   2025-12-04 [1] RSPM
 httr          1.4.7   2023-08-15 [1] RSPM
 jsonlite      2.0.0   2025-03-27 [1] RSPM
 knitr         1.51    2025-12-20 [1] RSPM
 methods     * 4.5.2   2025-10-31 [3] local
 png           0.1-8   2022-11-29 [1] RSPM
 R6            2.6.1   2025-02-15 [1] RSPM
 rlang         1.1.7   2026-01-09 [1] RSPM
 rmarkdown     2.30    2025-09-28 [1] RSPM
 sessioninfo   1.2.3   2025-02-05 [1] any (@1.2.3)
 stats       * 4.5.2   2025-10-31 [3] local
 tools         4.5.2   2025-10-31 [3] local
 utils       * 4.5.2   2025-10-31 [3] local
 withr         3.0.2   2024-10-28 [1] RSPM
 xfun          0.56    2026-01-18 [1] RSPM
 yaml          2.3.12  2025-12-10 [1] RSPM

 [1] /home/runner/work/_temp/Library
 [2] /opt/R/4.5.2/lib/R/site-library
 [3] /opt/R/4.5.2/lib/R/library
 * -- Packages attached to the search path.
```