Awesome R Package Development Tools

Table of contents

0.1 What is included? . . . 3

1 Swiss army knives **4**

2 Package templates **4**
2.1 Generic . . . 4
2.2 RMarkdown-based . . . 4
2.3 Shiny . . . 5
2.4 Meta-packages . . . 5

3 Naming things **5**

4 Working with package components **5**

5 Package configuration **6**

6 Package management tools **6**

7 Documentation **6**
7.1 Manual . . . 6
7.2 Math rendering in HTML/PDF manual . . . 7
7.3 Vignettes . . . 7
7.4 Tutorials . . . 7
7.5 Website . . . 7
7.6 Translation . . . 7
7.7 Lifecycle . . . 8
7.8 Badges and stickers . . . 8
7.9 Presentation . . . 8
7.10 Book . . . 8
7.11 Change log and versioning . . . 8

8 Documentation quality **8**

9 Observability and Monitoring **9**

10 Unit testing **9**
10.1 Generic R Packages . . . 9
10.2 Web/database applications . . . 10
10.3 Visual regression testing . . . 10
10.4 Mock testing . . . 10
10.5 Mutation testing . . . 10
10.6 Markdown documents . . . 11
10.7 Shiny applications . . . 11
10.8 Helpers for testing frameworks . . . 11

11 Code/Document Formatting **11**

12 Code analysis **12**
12.1 General . . . 12
12.2 Code review . . . 12
12.3 Code coverage . . . 12
12.4 Code quality . . . 13
12.5 Code complexity . . . 13
12.6 Code similarity . . . 13
12.7 Compiled code . . . 13
12.8 JavaScript code . . . 14
12.9 Lines of code . . . 14

13 Refactoring **14**

14 Code performance **14**
14.1 Benchmarking . . . 14
14.2 Profiling . . . 14

15 Reproducible Environments **15**
15.1 Package management . . . 15
15.2 Containerization . . . 15

16 Dependency Management **15**

17 CRAN/Bioconductor checks **16**

18 Usage **16**

19 CI/CD **17**

20 Security/Privacy 17

21 Build systems 17

22 Debugging 18

23 Input validation 18
23.1 Function argument validation . . . 18
23.2 Data validation . . . 18

24 Package metadata 19

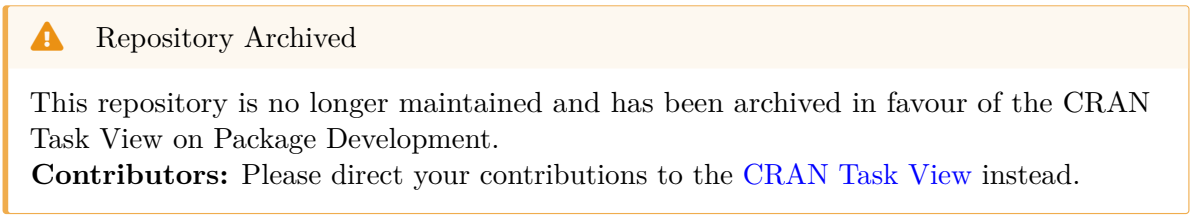
25 Reverse dependency checks 19

26 Gratitude 19

27 Integration with other languages 20
27.1 C++ . . . 20
27.2 Fortran . . . 20
27.3 Python . . . 20
27.4 Rust . . . 20
27.5 .NET Framework . . . 20
27.6 JavaScript/HTML/CSS . . . 20
27.7 Julia . . . 20

28 Upkeep 21

29 Sundry 21



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

0.1 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.

1 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}`

2 Package templates

2.1 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)

2.2 RMarkdown-based

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

2.3 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)

2.4 Meta-packages

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

3 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)

4 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)

5 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)

6 Package management tools

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

7 Documentation

7.1 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)

7.2 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)

7.3 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)

7.4 Tutorials

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

7.5 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)

7.6 Translation

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

7.7 Lifecycle

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

7.8 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)

7.9 Presentation

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

7.10 Book

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

7.11 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)

8 Documentation quality

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

9 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)

10 Unit testing

10.1 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)

10.2 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)

10.3 Visual regression testing

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

10.4 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)

10.5 Mutation testing

- {mutant} (mutation testing for R)

10.6 Markdown documents

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

10.7 Shiny applications

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

10.8 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)

11 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)

12 Code analysis

12.1 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.)

12.2 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)

12.3 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)

12.4 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)

12.5 Code complexity

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

12.6 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)

12.7 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)

12.8 JavaScript code

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

12.9 Lines of code

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

13 Refactoring

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

14 Code performance

14.1 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)

14.2 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)

15 Reproducible Environments

15.1 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)

15.2 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)

16 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)

17 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)

18 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)

19 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)

20 Security/Privacy

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

21 Build systems

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

22 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)

23 Input validation

23.1 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)

23.2 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)

24 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)

25 Reverse dependency checks

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

26 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)

27 Integration with other languages

27.1 C++

- {Rcpp}
- {cpp11}

27.2 Fortran

- {RFI}

27.3 Python

- {reticulate}

27.4 Rust

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

27.5 .NET Framework

- {rClr}

27.6 JavaScript/HTML/CSS

- {htmltools}
- {packer}

27.7 Julia

- {JuliaCall}

28 Upkeep

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

29 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)