Package 'sketchy'

October 9, 2026

Type Package

Title Create Custom Research Compendiums

Version 1.0.6

Maintainer Marcelo Araya-Salas <marcelo.araya@ucr.ac.cr>

Description Provides functions to create and manage research compendiums for data analysis. Research compendiums are a standard and intuitive folder structure for organizing the digital materials of a research project, which can significantly improve reproducibility. The package offers several compendium structure options that fit different research project as well as the ability of duplicating the folder structure of existing projects or implementing custom structures. It also simplifies the use of version control.

License GPL (>= 2)

Imports knitr, crayon, utils, git2r, rmarkdown, cli, urlchecker, stringr

Depends R (>= 3.5.0)

LazyData TRUE

URL https://github.com/maRce10/sketchy

BugReports https://github.com/maRce10/sketchy/issues

NeedsCompilation no

Suggests testthat (>= 3.0.0), withr, formatR, pak, renv

Config/testthat/edition 3

Repository CRAN

Language en-US

Encoding UTF-8

Config/roxygen2/version 8.0.0

Author Marcelo Araya-Salas [aut, cre] (ORCID: <https://orcid.org/0000-0003-3594-619X>), Andrea Yure Arriaga Madrigal [aut]

Date/Publication 2026-10-08 22:00:02 UTC

Contents

add_to_gitignore 2
check_urls 3
compendiums 4
load_packages 6
make_compendium 7
open_wd 10
print_skeleton 11
spot_unused_files 12

Index **14**

`add_to_gitignore` *Add entries to gitignore*

Description

`add_to_gitignore` adds entries to gitignore based on file extension or file size

Usage

```
add_to_gitignore(add.to.gitignore = FALSE, cutoff = NULL, extension = NULL, path = ".")
```

Arguments

`add.to.gitignore`	Logical to control if files are added to 'gitignore' or just printed on the console.
`cutoff`	Numeric. Defines the file size (in MB) cutoff used to find files (i.e. only files above the threshold would returned). 99 (MB) is recommended when hosting projects at github as the current file size limit is 100 MB.
`extension`	Character string to define the file extension of the files to be searched for.
`path`	Path to the project directory. Default is current directory.

Details

The function can be used to avoid conflicts when working with large files or just avoid adding non-binary files to remote repositories. It mostly aims to simplify spotting/excluding large files. Note that file names can be manually added to the '.gitignore' file using a text editor.

Value

Prints the name of the files matching the searching parameters and invisibly returns them (paths relative to 'path'). If `add.to.gitignore = TRUE` the files matching the search parameters ('cutoff' and/or 'extension') are added to '.gitignore' (a file used by git to exclude files from version control, including adding them to github), using their path relative to 'path'. Files already listed in '.gitignore' are not added again.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

See Also

compendiums, make_compendium

Examples

```
{
data(compendiums)

make_compendium(name = "my_compendium", path = tempdir(),
 format = "basic", force = TRUE)

# save a file
write.csv(iris, file.path(tempdir(), "my_compendium", "iris.csv"))

# add the file to gitignore
add_to_gitignore(add.to.gitignore = TRUE,
path = file.path(tempdir(), "my_compendium"), extension = "csv")
}
```

---

check_urls *Check urls in dynamic report files*

Description

`check_urls` Check urls in dynamic report files (.md, .Rmd & .qmd)

Usage

```
check_urls(path = ".")
```

Arguments

`path`	Path to the directory containing the files to be checked. Default is current directory.

Details

The function can be used to check if url addresses in dynamic reports are broken. Taken from Nan Xiao's blogpost (https://nanx.me/blog/post/rmarkdown-quarto-link-checker/).

Value

A url_checker_db object with an added class with a custom print method.

Author(s)

Nan Xiao (<me@nanx.me>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

Xiao, N. (2023). A General-Purpose Link Checker for R Markdown and Quarto Projects. Blog post. https://nanx.me/blog/post/rmarkdown-quarto-link-checker/

See Also

`add_to_gitignore`, `make_compendium`

Examples

```
data(compendiums)

# make compendiums
make_compendium(name = "my_compendium", path = tempdir(),
format = "basic", force = TRUE)

# check urls in scripts
check_urls(path = file.path(tempdir(), "my_compendium", "scripts"))
```

---

compendiums *List with compendium skeletons*

Description

`compendiums` is a list containing the format of 15 different project folder skeletons. For each format 3 elements are provided: '$skeleton' (folder structure), '$comments' and '$info' (reference to the original source).

Usage

```
data(compendiums)
```

Format

A list with 15 compendium formats:

basic basic sketchy format

figures similar to basic, but including output/figures folders

project_template following Kenton White's ProjectTemplate

pakillo following Francisco Rodriguez-Sanchez' template

boettiger following Carl Boettiger's blog

wilson following Wilson et al. (2017) format

small_compendium following Marwick et al (2018) small compendium format

medium_compendium following Marwick et al (2018) medium compendium format

large_compendium following Marwick et al (2018) large compendium format

vertical following Vuorre *et al.* (2018) R package vertical

rrtools following Marwick (2018) (R package rrtools)

rdir following folder structure described on at a r-dir blog post (although seems like it was removed)

workflowr following Blischak *et al.* (2019) R package workflowr

sketchy same skeleton than 'basic' but including a custom Rmarkdown and quarto files for documenting data analyses

github_site quarto website published on GitHub Pages through a GitHub action (as in this repo), including 'manuscript' and 'archive' folders (see `make_compendium`)

References

Blischak, J. D., Carbonetto, P., & Stephens, M. 2019. *Creating and sharing reproducible research code the workflowr way*. F1000Research, 8.

Marwick, B. 2018. *rrtools: Creates a reproducible research compendium*.

Marwick, B., Boettiger, C., & Mullen, L. 2018. *Packaging data analytical work reproducibly using R (and friends)*. The American Statistician, 72(1), 80-88.

Vuorre, Matti, and Matthew J. C. Crump. 2020. *Sharing and Organizing Research Products as R Packages*. PsyArXiv. January 15.

Wilson G, Bryan J, Cranston K, Kitzes J, Nederbragt L. & Teal, T. K.. 2017. *Good enough practices in scientific computing*. PLOS Computational Biology 13(6): e1005510.

load_packages *Install and load packages*

Description

load_packages installs and loads packages from different repositories.

Usage

```
load_packages(packages, quiet = FALSE, upgrade.deps = FALSE, quite = NULL)
```

Arguments

packages	Character vector with the packages to be installed (if missing) and loaded. Packages can be given as pak package specifications: a package name for CRAN (e.g. "kableExtra"), "bioc::package" for Bioconductor, "user/repo" for GitHub, "gitlab::user/repo" for GitLab or "git::url" for any git repository. Specific versions can be requested (e.g. "user/repo@v1.0" or "kableExtra@1.4.0"). Alternatively, the vector names can indicate the repository: 'cran', 'github', 'gitlab', 'bitbucket' or 'bioconductor' (for 'github', 'gitlab' and 'bitbucket' the string must be in the form 'user/package').
quiet	Logical argument to control if installation output and package startup messages are suppressed. Default is FALSE (messages are printed).
upgrade.deps	Logical argument to control if dependencies of the packages being installed are upgraded to their latest version. Default is FALSE (dependencies are only upgraded if required). Packages that are already installed are never upgraded.
quite	Deprecated. Use 'quiet' instead.

Details

Superseded: this function will keep working, but it is no longer recommended. Using it in reports or scripts makes projects depend on 'sketchy'. Install packages with pak::pkg_install() and load them with library() instead (as in the templates added by make_compendium), and use make_compendium(renv = TRUE) to record the package versions used in a project. For example:

```
packages <- c("kableExtra", "bioc::ggtree", "maRce10/Rraven")
package_names <- sub(".*[/:]", "", sub("@.*$", "", packages))
missing <- !vapply(package_names, requireNamespace, logical(1), quietly = TRUE)
if (any(missing)) pak::pkg_install(packages[missing])
invisible(lapply(package_names, library, character.only = TRUE))
```

The function installs missing packages and loads (attaches) all packages in a single call. Packages that are already installed are just loaded (no internet connection is needed). Installation is done with pkg_install (the 'pak' package is installed from CRAN if needed). The package name to load is taken from the package specification (e.g. "Rraven" for "maRce10/Rraven"), so it won't work for repositories in which the package name differs from the repository name or the package is in a sub-folder.

Value

Invisibly returns a named logical vector indicating which packages were successfully loaded.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

See Also

pkg_install, make_compendium

Examples

```
## Not run:
# CRAN, Bioconductor and GitHub packages
load_packages(packages = c("kableExtra", "bioc::ggtree", "maRce10/Rraven"))

# same packages using vector names to indicate the repository
load_packages(packages = c("kableExtra", bioconductor = "ggtree",
github = "maRce10/Rraven"), quiet = TRUE)

## End(Not run)
```

---

make_compendium *Generate folder structures for research compendiums*

Description

make_compendium generates the folder structure of a research compendium.

Usage

```
make_compendium(name = "research_compendium", path = ".", force = FALSE,
format = "basic", packrat = FALSE,
git = FALSE, clone = NULL, readme = TRUE, Rproj = FALSE, renv = FALSE)
```

Arguments

name	character string: the research compendium directory name. No special characters should be used. Default is "research_compendium".
path	Path to put the project directory in. Default is current directory.
force	Logical controlling whether existing folders with the same name are used for setting the folder structure. The function will never overwrite existing files or folders.
format	A character vector of length 1 with the name of the built-in compendiums available in the example object 'compendiums' (see compendiums for available formats). Default is 'basic'. Alternatively, it can be a character vector with 2 or more elements with the names of the folders and subfolders to be included (e.g. `c("folder_1", "folder_1/subfolder_1", "folder_1/subfolder_2")`).
packrat	Deprecated (packrat has been superseded by renv). Use 'renv' instead.
git	Logical to control if a git repository is initialized (`git2r::init()`) when creating the compendium. Default is FALSE.
clone	Path to a directory containing a folder structure to be cloned. Default is NULL. If provided 'format' is ignored. The folders '.git', '.Rproj.user', '..Rcheck' and '.quarto' (and their sub-folders) are ignored.
readme	Logical. Controls if a readme file (in Rmd format) is added to the project. The file has predefined fields for documenting objectives and current status of the project. Default is TRUE.
Rproj	Logical. If TRUE a R project is created (i.e. a .Rproj file is saved in the main project directory).
renv	Logical to control if renv is initialized (`renv::init()`) when creating the compendium. renv records the exact version of the packages used in the project (in the 'renv.lock' file) so they can be restored later or by other users with `renv::restore()`. renv is set to record all packages installed in the project library (`renv::settings$snapshot.type("all")`), so packages installed by the templates' package lists are recorded when running `renv::snapshot()`. The project is not activated in the current R session (it is activated when the project is opened in a new R session). Default is FALSE.

Details

The function takes predefined folder structures to generate the directory skeleton of a research compendium.

The format "github_site" sets up a project that is published as a website on GitHub Pages (see https://github.com/maRce10/suwo_publication for an example). The 'scripts' folder is a quarto website ('_quarto.yml', 'index.qmd' home page and 'analysis.qmd' analysis template) and '.github/workflows/publish.yml' contains a GitHub action that renders and publishes the site every time changes are pushed to the 'main' (or 'master') branch. Pages rendered locally are not rerun on GitHub: their results are taken from the 'scripts/_freeze' folder (`execute: freeze: true`), which must be committed. Without renv, only the packages needed to replay these results are installed on GitHub, so the site must be rendered locally ('quarto render' within 'scripts/') before pushing. With `renv = TRUE` the packages recorded in 'renv.lock' are installed on GitHub, so pages

that have not been rendered locally are run there (data used must be committed too). In this case a 'scripts/.Rprofile' file is added so renv is also activated when quarto renders the site from the 'scripts' folder. To publish the site, push the project to GitHub and set 'Source' to 'GitHub Actions' in the repository Settings > Pages. Note that quarto must be installed.

Value

A folder skeleton for a research compendium. In addition the structure of the compendium is printed in the console. Template files are also added depending on the format (existing files are never overwritten):

- If the compendium includes a "manuscript" folder: a manuscript template in Rmarkdown format ("manuscript.Rmd"), a BibTex file ("references.bib", for showing how to add citations) and an APA citation style file ("apa.csl").
- `format = "sketchy"`: analysis templates in Rmarkdown ("analysis_template_rmarkdown.Rmd") and quarto ("analysis_template_quarto.qmd") format, with their style files ("rmd.css" and "qmd.css"), in the "scripts" folder.
- `format = "github_site"`: a quarto website in the "scripts" folder and a GitHub action to publish it (see details).
- `readme = TRUE`: "README.Rmd" (and its rendered "README.md").
- `Rproj = TRUE`: an R project file ("name.Rproj").
- `renv = TRUE`: renv files ("renv.lock", ".Rprofile" and "renv" folder).

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

Marwick, B., Boettiger, C., & Mullen, L. (2018). Packaging Data Analytical Work Reproducibly Using R (and Friends). American Statistician, 72(1), 80-88.

Alston, J., & Rick, J. (2020). A Beginners Guide to Conducting Reproducible Research.

See Also

compendiums, print_skeleton

Examples

```
{
data(compendiums)

# default format
make_compendium(name = "mycompendium", path = tempdir(), format = "basic",
force = TRUE)
```

```
# quarto website to be published on GitHub Pages
make_compendium(name = "my_site_compendium", path = tempdir(),
 format = "github_site", force = TRUE)

# custom format
make_compendium(name = "my_second_compendium", path = tempdir(),
 format = c("folder_1", "folder_1/subfolder_1", "folder_1/subfolder_2"),
 force = TRUE)
}
```

---

open_wd *Open working directory*

Description

open_wd opens the working directory in the default file browser.

Usage

```
open_wd(path = ".", verbose = TRUE)
```

Arguments

path	Directory path to be opened. By default it's the working directory.
verbose	Logical to control whether the 'path' is printed in the console. Default is TRUE.

Details

The function opens the working directory using the default file browser and prints the working directory in the R console. This function aims to simplify the manipulation of files and folders in a project.

Value

Opens the working directory using the default file browser.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run citation("sketchy") for the current version).

See Also

spot_unused_files

Examples

```
{
open_wd()
}
```

`print_skeleton` *Print folder structures*

Description

`print_skeleton` prints the folder structure of a research compendium.

Usage

```
print_skeleton(path = ".", comments = NULL, folders = NULL)
```

Arguments

`path`	path to the directory to be printed. Default is current directory.
`comments`	A character vector with the comments to be added to folders in the graphical representation of the folder skeleton printed on the console. If named, names must match folder paths (e.g. `c("data/raw" = "raw data")`) and only some folders can be commented. If unnamed, it must have one element per folder (in alphabetical order of folder paths).
`folders`	A character vector including the name of the sub-directories of the project. If supplied, 'path' is only used as the name of the root folder in the printed tree (e.g. `path = "my_project"`).

Details

The function prints the folder structure of an existing project.

Value

The folder skeleton is printed in the console. A `cli_tree` object (see `tree`) is returned invisibly.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

See Also

compendiums, make_compendium

Examples

```
{
data(compendiums)

make_compendium(name = "my_other_compendium", path = tempdir(), format = "basic",
force = TRUE)

print_skeleton(path = file.path(tempdir(), "my_other_compendium"))
}
```

---

spot_unused_files *Spot/remove unused image and data files*

Description

spot_unused_files allow user to identify and optionally archive unused image or data files in a project directory

Usage

```
spot_unused_files(
  path = ".",
 file.extensions = c("png", "jpg", "jpeg", "gif", "bmp", "tiff", "tif", "csv", "xls",
    "xlsx", "txt"),
  script.extensions = c("R", "Rmd", "qmd"),
  archive = FALSE,
  ignore.folder = NULL,
  remove.empty = FALSE
)
```

Arguments

path	A character string with the path to the directory to be analyzed. Default is current directory.
file.extensions	A character vector with the file extensions to be considered. By default the function looks for the following image and file extensions: "png", "jpg", "jpeg", "gif", "bmp", "tiff", "tif", "csv", "xls", "xlsx" and "txt".
script.extensions	A character vector with the script extensions to be considered. Default is c("R", "Rmd", "qmd").

archive	A logical value indicating whether to archive the unused files. If TRUE the spotted files will be moved into the folder "archive" within 'path', keeping their original sub-folder structure. Default is FALSE.
ignore.folder	A character string with the path or paths to the directory(ies) to be ignored. Default is NULL.
remove.empty	A logical value indicating whether to remove empty folders within 'path' after moving the unused files. Hidden folders (e.g. '.git') and their contents are never removed. Default is FALSE.

Details

This function is used to spot/archive unused files in a project directory. The function searches recursively for all script files ('script.extensions') and all files with the target extensions ('file.extensions'). A file is considered unused when its name is not found in any of the scripts. Files already in the "archive" folder are ignored. It is useful to keep the project directory clean and organized. It is recommended to first run the function with `archive = FALSE` to check which files are spotted and then run `archive = TRUE` to move them into the "archive" folder.

Value

A data frame with 2 columns: file.name (self explanatory) and folder (where the file was found) with the unused files. If no unused files are found NULL is returned invisibly.

Author(s)

Marcelo Araya-Salas (<marcelo.araya@ucr.ac.cr>)

References

Araya-Salas, M., & Arriaga Madrigal, A. Y. sketchy: Create Custom Research Compendiums. R package (run `citation("sketchy")` for the current version).

See Also

add_to_gitignore, make_compendium

Examples

```
## Not run:
spot_unused_files(path = "path/to/your/project")

## End(Not run)
```

Index

∗ **data manipulation**
open_wd, 10
∗ **datasets**
compendiums, 4

add_to_gitignore, 2, *4*, *13*

check_urls, 3
compendiums, *3*, 4, *8, 9*, *12*

load_packages, 6

make_compendium, *3–7*, 7, *12, 13*

open_wd, 10

pkg_install, *6, 7*
print_skeleton, *9*, 11

spot_unused_files, *10*, 12

tree, *11*