Package 'alluseful'

February 27, 2026

Title Contains my personal templates, functions, themes etc.

Version 0.1.4

Description This is a personal package created to contain all the useful elements I need for effective coding. It contains functions to set up my rstudio IDE how I like, personalised themes, templates for scripts, packages etc. It is not intended for distribution or use by anyone else.

License MIT + file LICENSE

Encoding UTF-8

Imports cli (>= 3.6.2), credentials (>= 2.0.1), emoji (>= 16.0.0), fs (>= 1.6.4), gert (>= 2.1.0), gitcreds (>= 0.1.2), glue (>= 1.7.0), here (>= 1.0.1), magrittr (>= 2.0.3), pacman (>= 0.5.1), purrr (>= 1.0.2), quarto (>= 1.4.4), rlang (>= 1.1.6), rstudioapi (>= 0.16.0), stats (>= 4.2.2), stringr (>= 1.5.1), usethis (>= 3.0.0)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Josh Moatt [aut, cre]

Maintainer Josh Moatt <joshua.moatt@defra.gov.uk>

R topics documented:

add_editor_fonts *add my custom fonts to the editor selection*

Description

Adds my custom fonts (all open source) to the editor UI

Usage

```
add_editor_fonts(dash = TRUE)
```

Arguments

dash	logical, default TRUE. If TRUE uses .config file on DASH RStudio server, else uses windows.

Value

fonts added to UI

Author(s)

Josh Moatt

add_proj_files *Add default project files to an existing project/repo*

Description

Simple function to add the standard project files to an existing R Project or local GitHub Repo. This should be used when setting up a new project after initial set up - particularly useful for if you have just cloned a new GitHub repo and want to add the template project files.

Usage

```
add_proj_files(
  path = here::here(),
  title = "Project title",
  readme = "github",
  gitignore = TRUE
)
```

Arguments

`path` string containing file path to add files. Default is current directory (set using `here()`)

`title` string with project title. If no title provided, defaults to "Project title"

`readme` string specifying readme format. One of "github", "markdown" or "html". Default is "github"

`github` logical controlling whether a gitignore is added. Default is TRUE

Details

The function is designed to be used after manually creating a local version of a GitHub repo.

The function will add the following:

- data, src and outputs folders
- README using `create_fs_readme`
- pipeline script using `create_fs_script`
- optionally creates a gitignore using `create_fs_gitignore`

The function allows users to include the author and project title when calling the function, which will be prefilled

Value

project files added to specified directory

Author(s)

Josh Moatt

`add_themes` *Add my rstudio themes*

Description

Function to add my rstudio themes from within this package.

Usage

```
add_themes()
```

Value

themes added to the rstudio api

Author(s)

Josh Moatt

`connect_github_pat` *Connect RStudio to your GitHub account via PAT*

Description

This is a simple function which connect RStudio to GitHub, allowing you to work on GitHub repositories. This is an essential part of Reproducible Analytical Pipelines and best practice for coding.

Usage

```
connect_github_pat(defra = TRUE)
```

Arguments

`defra`	default TRUE. If TRUE will set the defra proxy in the terminal (needed for linking to github)

Details

This is a simple function which will set you GitHub credentials and Personal Access Token (PAT) and connect RStudio to GitHub. This is essential if you want to work in RStudio in projects/repos stored on GitHub.

It uses `system()` to run the necessary code in the terminal to set your credentials, and uses `gitcreds_set()` and `set_github_pat()` from the `gitcreds` and `credentials` packages to connect your RStudio to GitHub.

`gitcreds_set()` is an interactive function and will prompt users for input. To replace existing credentials/PAT choose option 2. You will then be prompted for you PAT. PAT should be changed every 30 days to ensure security.

An additional feature I have added, not mentioned in the Defra instructions is to add the `set_github_pat()` function call to your .Rprofile. This will ensure your PAT is set for every R session, meaning you wont need to provide your PAT when running functions such as `install_github()` from the devtools package.

Note: For this function to work you must:

- have git installed on your local machine
- have a GitHub account
- have created a Personal Access Token (PAT) on GitHub.

Guidance on how to create a PAT can be found here: ADD LINK.

Value

GitHub credentials and PAT set

Author(s)

Josh Moatt

`connect_github_ssh` *Connect RStudio to your GitHub account using a SSH*

Description

Connect RStudio to your GitHub account using SSH - this is the preferred method of connecting RStudio and GitHub on the DASH platform. The function is set up to use my username and email address,using these to set my credentials and generate an SSH key. Once the SSH key has been added to GitHub, the function does the final bits of set-up. All steps and outputs are printed in the RStudio console rather than the terminal, which offers a more user friendly experience.

Usage

```
connect_github_ssh()
```

Details

The suggested method for connecting RStudio and GitHub on the DASH platform is via SSH. The alternative is via Personal Access Token, but this method can be a bit clunky and frustrating on the DASH platform. With SSH providing a much smoother user experience.

This function is designed to simplify the connection process and avoid the me having to interact with the terminal. Instead, all prompts and outputs are returned in the RStudio console.

The function will set my username and email then print an output to confirm they have been set. The function will then create a hidden SSH folder and sub-folder for "id_ed25519" (~/.ssh/id_ed25519). Once done, it will then generate an SSH key and save it to this folder. The SSH key will then be printed in the console. The function then adds GitHub as a known host.

At this point, the function pauses and asks the for confirmation that the SSH key has been added to GitHub. Now I can then copy the SSH key from the console, go to GitHub -> settings -> SSH and GPG keys and add the SSH key.

Once added, the I can respond to the prompt and the function will finish establishing the connection. If all works as expected, the message "Hi jpmoatt! You've successfully authenticated, but GitHub does not provide shell access" will be printed in the console.

Author(s)

Josh Moatt

`create_gitignore` *Create a gitignore file based on the my template*

Description

Use this function to create a gitignore for a project.

Usage

```
create_gitignore(type = "default", file_path = NULL, custom_txt = NULL)
```

Arguments

`type` description controlling which gitignore is added. "default" will add the standard template. "custom" will enable the user to provide a custom template via `custom_txt`.

`file_path` optional argument allowing users to specify file path where gitignore should be created. If not entered, will default to current project/working directory.

`custom_txt` optional argument allowing users to provide their own gitignore template. Must be provided as a string. Only used if type set to "custom".

Details

This function will create a gitignore for a project using a pre-defined template.

Used as default, it will automatically add the data and output folders (as created in the project template). This is to ensure nor restricted data or unpublished results are accidentally pushed to GitHub.

Alternatively, a custom gitignore can be provided by setting `type` to "custom" and providing a custom template to `custom_txt` as a string.

Note it will replace any existing .gitignore files present in the project already.

Value

A gitignore file is added to the project.

Author(s)

Josh Moatt

`create_readme` *Create a README using my template*

Description

Use this function to create a README for a project.

Usage

```
create_readme(
  format = c("markdown", "github", "html"),
  file_path = NULL,
  readme_title = NULL
)
```

Arguments

`format` controls the output format of the README. Default is "markdown", but can be "html" or "github".

`file_path` sting containing file path where README will be saved.

`readme_title` string containing README title. If no string provided will be set to "README (edit title)".

Details

This function will create a README for a project using a pre-defined template. The function will create a Quarto (.qmd) file and do a first render producing a desired output.

The output type can be controlled using the format option (default is markdown). Can also be html or github (gfm).

Value

Output is a .qmd file containing the desired README template and an initial render of the README in the desired output.

Author(s)

Josh Moatt

`create_script` *Create a new script with my header.*

Description

This function will create a new script with my header added.

Usage

```
create_script(
  file_name = NULL,
  file_path = NULL,
  project = NULL,
  date = format(Sys.Date(), "%d/%m/%Y")
)
```

Arguments

`file_name`	string containing desired name for script.
`file_path`	string containing folder name to save script. This is built on the here function in R, so will follow your root directory. If you want to save in a sub-folder, enter the full folder sequence, e.g. "folder/sub-folder".
`date`	string containing a date. By default, this will be set as today's date.

Value

An R script will be saved in the root directory or in the specified folder.

Author(s)

Josh Moatt

proj_template *Create RStudio project template*

Description

This is a function that is called in the "New project" viewer pane when the user chooses the useful project template. It should not be used away from the RStudio "New project" viewer.

I have not included additional information on how to use this function, as it is not intended to be used outside the template call.

To subsequently link this to a github repo, the best plan is to use `use_github()`.

Usage

```
proj_template(path, ...)
```

Author(s)

Josh Moatt

roxygen_addin *Addin to open new script with roxygen template*

Description

This addin will open a new blank R script which is populated with a standard roxygen header.

Usage

```
roxygen_addin()
```

Details

This addin will create a new blank script that is populated with standard roxygen header.

Addins must use the rstudioapi package. This addin uses the `documentNew()` function from the `rstudioapi` package to open the new R script.

Note: this opens an normal untitled script. It must be manually saved in the correct directory.

Value

A script will open in RStudio.

Author(s)

Josh Moatt

script_addin *Addin to open new script with my template*

Description

This addin will open a new blank R script which is populated with my header.

Usage

```
script_addin()
```

Details

This addin will create a new blank script that is populated with my template header.

Addins must use the rstudioapi package. This addin uses the `documentNew()` function from the `rstudioapi` package to open the new R script.

Note: this opens an normal untitled script. It must be manually saved in the correct directory.

Value

A script will open in RStudio.

Author(s)

Josh Moatt

setup_rstudio *Apply my preferred RStudio settings and layout.*

Description

This function will change the settings and layout of RStudio to my usual preferences. This has the source pane on the left, the console and terminal on the top right and all others on the bottom right. It also applies my usual global options (E.g. rainbow brackets, hexcode preview on etc).

Usage

```
setup_rstudio(font)
```

Arguments

`font`	string containing name of font to load. Need to refresh the browser window for font change to take effect.

Value

RStudio configured to my settings

Author(s)

Josh Moatt

`set_console_prompt` *Add my R console prompt.*

Description

Function to change the R console prompt from default to one of two custom options. Either a themed emoji and the git branch, or just the git branch. This can be set at the user level or the project level.

Usage

```
set_console_prompt(scope = c("user", "project"), prompt = c("emoji", "git"))
```

Arguments

`scope`	string. "user" sets the prompt globally, "project" sets it just for the active project.
`prompt`	string. "emoji" sets the prompt to be a themed emoji and git branch (if active) or "git" sets it to the active git branch.

Value

Altered R prompt

Author(s)

Josh Moatt

`set_proj_pat` *Add `set_github_pat()` to project specific .Rprofiles*

Description

A simple function to add the `credentials::set_github_pat()` function call to your project specific .Rprofile. This is needed for projects where `renv` is activated, so you can avoid hardcoding your PAT anywhere. See connect_github_pat for full details and reasoning behind this function.

Usage

```
set_proj_pat()
```

Value

updated .Rprofile

Author(s)

Josh Moatt

`set_script_template` *Create an default R script template.*

Description

This function can be used to create a default R script template. All scripts opened from this point will have this template applied as a header.

Usage

```
set_script_template(
  format = c("mine", "manual_edit", "blank"),
  template = NULL,
  dash = TRUE
)
```

Arguments

`format` what format the template will take. There are four options: "mine" (my default template), "custom" will apply a custom template (provided as a string), "manual_edit" will open the template so you can manually edit the template, and "blank" can be used to remove all pre-existing templates. Note: manual edit can also be used to edit existing templates.

`template` default is NULL, only used if format = "custom". Used to provide custom template design. Must be provided as a string. Must be provided if using format = "custom" or the function will return an error.

`dash` default is FALSE. If TRUE changes the file path to the one needed for the RStudio server on DASH

Details

This function is used to create a new template for R scripts. This will be the default that all subsequent R scripts opened will contain. By doing this is it should be easier to follow best practice and properly comment all scripts you create.

The default is stored in the appdata folder on your c drive: "~/AppData/Roaming/RStudio". It will create a "templates" folder where the default will be stored.

The function has various ways it can work which will give you the ability to create whatever header template you wish. It has four ways formatting options:

- mine
- custom
- manual_edit
- blank

"mine" will pre-load the default script with my template.

"custom" will allow you to provide your own custom template as a string, which will then be added to the default.

"manual" will allow you to manually edit the default template. It will open it in your R studio window and manual edits can be saved. Note, this can also be used to tweak a pre-existing template (e.g. to add your name and email to all scripts).

"blank" will delete the default R script and template. This returns R back to normal, and any script opened subsequently will be blank.

Value

New .R file created containing the script template

Author(s)

Josh Moatt

`sqllight_addin` *Apply my rstudio themes via addins*

Description

Addins to apply my rstudio themes

Usage

```
sqllight_addin()

kisstheme_addin()

draculatheme_addin()

cobalttheme_addin()
```

Value

themes added to the rstudio api

Author(s)

Josh Moatt

`use_github` *Create a GitHub repository from local project.*

Description

This simple function will turn your local R project into a git repo, then create a repo on GitHub and perform the initial set up and commit.

By default it will create a private repo on the Defra-Data-Science-Centre-of-Excellence organisation on GitHub. You can change where the repo is created using the defra_org argument and change the repo visibility using the visibility argument when calling the function.

Note: this function requires you to have a PAT and to have set this in your RStudio environment, even if connecting via SSH. This is because a PAT is required to create new repos, even when using SSH. See usethis::use_github for more details.

Usage

```
use_github(
  message = "Initial commit",
  defra_org = TRUE,
  visibility = "private",
  github_method = "ssh"
)
```

Arguments

`message`	initial commit message. Default is "Initial commit".
`defra_org`	logical, default is TRUE. Whether the repo should be created under the Defra-Data-Science-Centre-of-Excellence (TRUE) or a personal repo (FALSE).
`visibility`	string controlling visibility. One of "private" (default), "internal" or "public". If defra_org is FALSE repos can only be private or public. When "internal" used when defra_org is FALSE, a private repo will be created.
`github_method`	sting specifying GitHub connection method. One of "ssh" or "https". By default, the function uses "ssh" as this is the recommended connection method on the DASH platform.

Details

This function will take an existing R project on you local machine, turn it into a git repo and create a GitHub repository.

By default it will create a private repo on the Defra-Data-Science-Centre-of-Excellence organisation on GitHub. This can be changed using the defra_org argument.

The function will do the following:

- add a gitignore
- initalise the git repo
- stage any uncommitted files
- commit the files
- create a GitHub repo

- restart RStudio to activate the git pane in R

The gitignore is added using the `create_fs_gitignore()` function within this package.

The repo initialisation, staging and committing is all done using the `gert` package.

Creating the GitHub repo uses the `use_github()` function from the `usethis` package.

By default the function will create a private repo. There are three options controlled by the visibility argument:

- private
- internal
- public

For the differences between these options, see GitHub. Internal is only an option when creating a repo within the Defra organisation. If createing a personal repo with visibility set to "internal", this will create a private repo.

For this function to work you must:

- have git installed on your machine
- have a GitHub account
- have your GitHub credentials entered into RStudio
- have your Personal Access Token (PAT) entered in RStudio

Note: occasionally the function seems to fails to set the master branch and users are unable to push changes. If this happens try running `git push -u origin master` in the terminal, this should set you current branch as the master. We're not sure why this happens, but it is advisable to use `fs_connect_github()` to set your credentials properly before trying this function.

Value

R project is turned into a git repo and an associated github repo si created in the users github account.

Author(s)

Josh Moatt

Index