Package 'fssetup'

March 27, 2025

Title Package for Producing R Projects and Scripts Using the Defra Farming Stats Template

Version 0.1.4

Description This package contains the functions needed to create various features in R using the Defra Farming Stats Team templates. Including scripts, projects, readmes and gitignores. It also contains useful functions for linking RStudio and RStudio projects to GitHub.

License MIT + file LICENSE

Encoding UTF-8

Imports cli (>= 3.6.2), credentials (>= 2.0.1), fs (>= 1.6.4), gert (>= 2.1.0), gitcreds (>= 0.1.2), glue (>= 1.7.0), here (>= 1.0.1), httr (>= 1.4.7), httr2 (>= 1.0.5), magrittr (>= 2.0.3), openxlsx (>= 4.2.7.1), pkgbuild (>= 1.4.4), purrr (>= 1.0.4), quarto (>= 1.4.4), readr (>= 2.1.5), rstudioapi (>= 0.16.0), stringr (>= 1.5.1), usethis (>= 3.0.0)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

URL https://github.com/jpmoatt/fssetup

BugReports https://github.com/jpmoatt/fssetup/issues

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Josh Moatt [cre, aut]

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

R topics documented:

addin_fs_roxygen 2
addin_fs_script 2
create_fs_gitignore 3
create_fs_readme 4
create_fs_script 4
create_fs_script_template 6
customise_layout 7

fs_add_proj_files 8
fs_connect_github 9
fs_proj 10
fs_use_github 10
set_databricks_pat 12
temp_file 12
uc_volume_get 13
uc_volume_put 13

Index **15**

`addin_fs_roxygen` *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

```
addin_fs_roxygen()
```

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.

This package also contains another addin to open an R script with a farming stats header, which can be used for creating normal R scripts.

Value

A script will open in RStudio.

`addin_fs_script` *Addin to open new script with Farming Stats template*

Description

This addin will open a new blank R script which is populated with the Farming Stats header. Unlike the `create_fs_script()` function, this addin will also pre-populate the date.

Usage

```
addin_fs_script()
```

Details

This addin will create a new blank script that is populated with the farming stats template header. Unlike the `create_fs_script()` function in this package, this will auto-populate the script with today's date.

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.

This package also contains another addin to open an R script with a roxygen header, which can be used for creating scripts within packages.

Value

A script will open in RStudio.

`create_fs_gitignore` *Create a gitignore file based on the Farming Stats template*

Description

Use this function to create a gitignore for a Farming Stats project.

Usage

```
create_fs_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 Farming Stats template.

Used as default, it will automatically add the data and output folders (as created in the fssetup 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.

create_fs_readme *Create a Farming Stats README*

Description

Use this function to create a README for a Farming Stats project.

Usage

```
create_fs_readme(
  format = c("markdown", "github", "html"),
  file_path = NULL,
  author = 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.
author	string containing authors name. If no string provided author will be set to "add author".
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 Farming Stats 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.

create_fs_script *Create a new script with farming stats header.*

Description

This function will create a new script with the farming stats header added. There a multiple options for customization (see details).

Usage

```
create_fs_script(
  file_name = NULL,
  file_path = NULL,
  author = NULL,
  email = 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".
author	string containing author's name.
email	string containing author's email.
date	string containing a date. By default, this will be set as today's date.

Details

This is a simple function to create a new script with the farming stats header template added.

By just calling the function a script will be created with the template entered. All fields of the template will be blank except for the date, which is set to today's date.

There are options for customising and entering some details automatically. The details that can be pre-entered are:

- script name
- location to save
- author
- author's email
- date created

By default these are set to NULL (except for the date). But can be set to any string.

Value

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

`create_fs_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

```
create_fs_script_template(
  format = c("farming_stats", "custom", "manual_edit", "blank"),
  template = NULL,
  dash = FALSE
)
```

Arguments

`format`	what format the template will take. There are four options: "farming_stats" (default farming stats 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:

- farming_stats
- custom
- manual_edit
- blank

"farming_stats" will pre-load the default script with the farming stats 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

`customise_layout` *Customise RStudio IDE layout*

Description

Simple function I created to customise the RStudio IDE. Maks setting up RStudio on DASH much easier and quicker following a cluster restart.

Usage

```
customise_layout(...)
```

Arguments

`...`	Named arguments representing RStudio preferences to be updated. Each argument should be named after an RStudio preference and assigned the desired value.

Value

The function modifies RStudio preferences as a side effect.

Author(s)

Josh Moatt

Examples

```
## Not run:
# set pane layout
my_pane_layout <- list(
  quadrants = list("Source", "TabSet1", "Console", "TabSet2"),
  tabSet1 = list("History", "Presentation"),
 tabSet2 = list("Environment", "Files", "Plots", "Connections", "Packages", "Help", "Build", "VCS", "Tutorial"
  hiddenTabSet = list(),
  console_left_on_top = FALSE,
  console_right_on_top = TRUE,
  additional_source_columns = 0
)

# apply preferences
customise_layout(
  always_save_history = FALSE, # don't auto save history
```

```
  save_workspace = "never", # don't save workspace
  load_workspace = FALSE, # don't load previous workspace
  restore_last_project = FALSE, # don't restore last opened project
 continue_comments_on_newline = TRUE, # when commenting, hitting enter continues comment on new line
  highlight_selected_line = FALSE, # highlight line cursor is on
  highlight_r_function_calls = TRUE, # highlight R function calls
  show_margin = FALSE, # don't show margin (default = 80 characters)
  rainbow_parentheses = TRUE, # colour match brackets
  color_preview = TRUE, # hexcode previews on
  panes = my_pane_layout # Pane layout as set above
)

## End(Not run)
```

`fs_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

```
fs_add_proj_files(
  path = here::here(),
  author = "Author name",
  title = "Project title",
  readme = "github",
  gitignore = TRUE
)
```

Arguments

`path`	string containing file path to add files. Default is current directory (set using `here()`)
`author`	string with author name. If no name provided, defaults to "Author name"
`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

`fs_connect_github` *Connect RStudio to your GitHub account*

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

```
fs_connect_github(username, email)
```

Arguments

`username`	string containing GitHub username
`email`	string containing email address used for gitHub account

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 `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

`fs_proj` *Function for RStudio project template*

Description

This is a function that is called in the "New project" viewer pane when the user chooses a Farming Stats 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 `fs_use_github()`.

Usage

```
fs_proj(path, ...)
```

`fs_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 envrionment, 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

```
fs_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.

`set_databricks_pat` *Set databricks PAT*

Description

Simple function to set databricks PAT. Done via function to avoid committing to GitHub accidentally.

Usage

```
set_databricks_pat()
```

Value

databricks PAT written to environment

Author(s)

Josh Moatt

`temp_file` *function for creating a tempfile when exporting data to DASH*

Description

Very simple function to aid in exporting an object from the R environment on a DASH RStudio cluster to the databricks filestore. Desingned to be used alongside the function db_volume_write from the `brickster` package.

Usage

```
temp_file(object, file_type)
```

Arguments

`object`	object from R environment to export. So far works with all types (e.g. data.frame, openxlsx workbook etc).
`file_type`	string specifying file type of the tempfile. Can be "rds", "xlsx" or "csv". Other types can be added to the function if needed.

Value

a tempfile with specified extension.

Author(s)

Josh Moatt

`uc_volume_get` *Pull data from the DASH Unity Catalog into an RStudio cluster*

Description

Currently the DASH Unity Catalog (UC) and RStudio server cannot directly interact. So you must interact with the databricks API and copy data from DASH UC to the RStudio server. This is a simple function to read data from UC into RStudio cluster on databricks. It is built using the GET function from the httr package.

Note, you must have set up a databricks PAT in the databricks settings before doing this.

Usage

```
uc_volume_get(workspace, volume, token, out_file)
```

Arguments

`workspace`	databricks workspace (string)
`volume`	file path to data folder (string)
`token`	databricks PAT (string)
`out_file`	filepath on RStudio Cluster that data should be stored.

Value

data saved in Rstudio cluster.

Author(s)

Josh Moatt

`uc_volume_put` *Write data to the DASH Unity Catalog from RStudio cluster*

Description

Currently the DASH Unity Catalog (UC) and RStudio server cannot directly interact. So you must interact with the databricks API and copy data to the DASH UC from the RStudio server. This is a simple function to save data from the RStudio cluster onto databricks. It is built using the httr2 package and uses a PUT request to save the data.

Note, you must have set up a databricks PAT in the databricks settings before doing this.

Usage

```
uc_volume_put(workspace, volume, token, file, folder)
```

Arguments

workspace	databricks workspace (string)
volume	path to folder on databricks where data should be written (string)
token	databricks PAT (string)
file	name of file to be exported including file extension. Will be used to both specify the file to export and specify the file to be created on databricks (String).
folder	path to folder on RStudio Cluster that contains data to export.

Value

data saved in Rstudio cluster.

Author(s)

Josh Moatt

Index

addin_fs_roxygen, 2
addin_fs_script, 2

create_fs_gitignore, 3
create_fs_gitignore(), *11*
create_fs_readme, 4
create_fs_script, 4
create_fs_script(), *2, 3*
create_fs_script_template, 6
customise_layout, 7

documentNew(), *2, 3*

fs_add_proj_files, 8
fs_connect_github, 9
fs_connect_github(), *11*
fs_proj, 10
fs_use_github, 10
fs_use_github(), *10*

GET, *13*
gitcreds_set(), *9*

install_github(), *9*

set_databricks_pat, 12
set_github_pat(), *9*
system(), *9*

temp_file, 12

uc_volume_get, 13
uc_volume_put, 13
use_github(), *11*
usethis::use_github, *10*