Package 'represtools'

October 14, 2022

Version 0.1.3

Title Reproducible Research Tools

Description Reproducible research tools automates the creation of an analysis directory structure and work flow. There are R markdown
skeletons which encapsulate typical analytic work flow steps. Functions will create appropriate modules which may
pass data from one step to another.

Maintainer Brian Fannin <captain@pirategrunt.com>

URL https://pirategrunt.com/represtools/

BugReports https://github.com/PirateGrunt/represtools/issues

Depends R (>= 3.2.0)

License GPL (>= 3)

SystemRequirements GNU make

LazyData true

Imports whisker

Suggests testthat, knitr, rmarkdown

RoxygenNote 6.1.1

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

Author Brian Fannin [aut, cre]

Repository CRAN

Date/Publication 2019-10-03 17:30:03 UTC

R topics documented:

Analyze   *Create an analysis file*

Description

Create an analysis file

Usage

```
Analyze(fileStem, inputFiles, title)
```

Arguments

fileStem	The file stem
inputFiles	A vector of files to load
title	Title of the document

Cook   *Create a cook file*

Description

Create a cook file

Usage

```
Cook(fileStem, inputFiles, title)
```

Arguments

fileStem	The file stem
inputFiles	A vector of files to load
title	Title of the Cook document

DescribeObjects	*I'm throwing in the towel on this function. I'm just not clear on the behavior that I'd like:*

Description

This will apply functions to a vector of objects.

Usage

```
DescribeObjects(objects, FUNS, env)
```

Arguments

objects	A list of objects to be described
FUNS	the functions which will describe the objects
env	An enviroment. If missing, the global environment will be used.

Details

1. Given a vector of objects, apply a vector of functions. This is most common when I want to apply both 'str' and 'summary' to a data frame.

2. How is that implemented in practice? I either want the output printed or I want to capture the output to print later, but I'm not sure. The easiest option would be to print everything.

3. And that's what I'll do.

Describe the objects

This may be used to quickly summarize a set of objects. Note that functions are called for their side effects of printing output

x may be given as a character vector or a list.

Examples

```
## Not run:
myFit1 <- lm(myData, y ~ x1 + x2)
myFit2 <- lm(myData, y ~ x3)

DesribeObjects(list(myFit1, myFit2), broom::tidy)

## End(Not run)
```

Gather   *Create a gather file*

Description

Create a gather file

Usage

```
Gather(fileStem, title)
```

Arguments

fileStem	The file stem
title	Title of the Gather document

GetSQL   *Read a SQL file*

Description

Read a SQL file

Usage

```
GetSQL(filename, warn = FALSE)
```

Arguments

filename	A character variable with a filename
warn	Warn if there is a missing final end of line. Default is TRUE

Value

String

ListObjects *List objects*

Description

Form a character vector of the names of objects, based on patterns.

Usage

```
ListObjects(patterns = c("^df", "^plt", "^fit"), env)
```

Arguments

patterns	A vector of patterns
env	An enviroment. If missing, the global environment will be used.

Details

This will collect objects based on a vector of character patterns.

Value

character vector

LoadObjects *LoadObjects*

Description

This function will load a set of objects stored in .rda files. By default, objects are loaded into the Global environment.

Usage

```
LoadObjects(params, env = .GlobalEnv, loadFiles)
```

Arguments

params	A list of parameters. Should include the named items inputDir and inputFiles.
env	An environment in which to load objects. Defaults to the global environment.
loadFiles	A character vector of files containing saved data.

Details

The params object is defined within an RMarkdown file and contains named items "inputDir" and "inputFiles". As an expedient, one may pass a vector of filenames. This vector will only be used if the params argument has been omitted.

Value

Character vector containing the names of loaded objects.

Examples

```
## Not run:
loadedObjects <- LoadObjects(params)

## End(Not run)
```

Make   *Run the make command*

Description

This will run the make program with whatever target the user selects. The default target is "all". The function works by calling the [system2](#) function. It is presumed that an appropriate make utilty exists on the user's computer. Further note that the default makefile for an analysis project uses GNU extensions like wildcards.

Usage

```
Make(target = "all", command = "", ...)
```

Arguments

target	The make target to run
command	User-supplied location of the make command
...	Additional arguments to the [system2](#) function

Details

The function will to try to confirm

See Also

[system2](#) [system](#) [Sys.which](#)

Examples

```
## Not run:
Make()
Make("clean", command = "/usr/bin/gmake")
Make("gather")
Make("MyFile.rda")

## End(Not run)
```

NamesToObjects   *NamesToObjects*

Description

Returns a list of the objects named in x

Usage

```
NamesToObjects(x, envir)
```

Arguments

x	Vector or list of named objects
envir	Environment in which to search

Details

Return a list of named objects

x is given as either a character vector containing names of objects to return, or a list. If x is a list, it will first be converted to character. If objects do not exist, no error or warning will be generated.

Value

A list of objects

Examples

```
## Not run:
mojo <- 4
gonzo <- "Moe"
lstObjects <- NamesToObjects(c("mojo", "gonzo"))

## End(Not run)
```

NewResearch   *Create a new Research*

Description

Create a new Research

Usage

```
NewResearch(path = ".", RStudio = TRUE, useExistingDirectory = FALSE)
```

Arguments

path	A character variable indicating the name of the project to create
RStudio	Boolean indicating whether or not to create an RStudio project. Default is TRUE.
useExistingDirectory	
	Overwrite the directory if it already exists. Default is FALSE.

Value

NewResearch will invisibly return the name of the project.

OutputFile   *OutputFile*

Description

Construct an output file from a params list.

Usage

```
OutputFile(params)
```

Arguments

params	List object with output directory and file stem elements

Present	*Create a presentation file*

Description

Create a presentation file

Usage

```
Present(fileStem, inputFiles, title, outputFormat)
```

Arguments

fileStem	The file stem
inputFiles	A vector of files to load
title	Title of the document
outputFormat	A string indicating the outputFormat

represtools	*represtools.*

Description

represtools.

Index