Package 'swirlify'

August 29, 2016

Title A Toolbox for Writing 'swirl' Courses

Description A set of tools for writing and sharing interactive courses to be used with swirl.

URL http://swirlstats.com

Version 0.5.0

License MIT + file LICENSE

Depends R (>= 3.2.0)

Imports swirl (>= 2.4.2), stringr, yaml, rmarkdown, whisker, shiny, shinyAce, base64enc

Encoding UTF-8

Suggests testthat, digest

RoxygenNote 5.0.1

NeedsCompilation no

Author Sean Kross [aut, cre], Nick Carchedi [aut], Chih-Cheng Liang [ctb], Wush Wu [ctb]

Maintainer Sean Kross <sean@seankross.com>

Repository CRAN

Date/Publication 2016-07-14 00:28:45

R topics documented:

`add_license` *Add a LICENSE.txt file to your course*

Description

Licensing your course is important if you want to share your course. For more information see https://github.com/swirldev/swirlify/wiki/Licensing-Your-Course. For more information about Creative Commons licenses see http://creativecommons.org/licenses/. For more information about software licenses see http://www.gnu.org/licenses/license-list.en.html.

Usage

```
add_license(author, year = format(Sys.Date(), "%Y"),
  open_source_content = TRUE, content_license = "CC BY 4.0",
  open_source_data = TRUE, data_license = "CC0", open_source_code = TRUE,
  code_license = "MIT")
```

Arguments

`author`	The author of the course. This can be an organization.
`year`	The year the course was written.
`open_source_content`	If `TRUE` a Creative Commons content license will be included pertaining to the content of your course.
`content_license`	Specify which Creative Commons license you would like to use for the content of your course. This must be equal to one of the following: `"CC BY 4.0"`, `"CC BY-SA 4.0"`, `"CC BY-ND 4.0"`, `"CC BY-NC 4.0"`, `"CC BY-NC-SA 4.0"`, `"CC BY-NC-ND 4.0"`, or `"CC0"`.

open_source_data
: If TRUE a Creative Commons content license will be included pertaining to the data distributed with your course.

data_license
: Currently this value must be equal to "CC0", but in the future it may be able to be other values.

open_source_code
: If TRUE a free software license will be included pertaining to the software included in your course.

code_license
: Specify which open source software license you would like to use for the content of your course. This must be equal to one of the following: "MIT", "GPL3", "CC0".

Examples

```
## Not run:

# Add a license with simple open source options
add_license("Team swirl")

# Add a license so that derivative works are shared alike
add_license("Team swirl", content_license  = "CC BY-SA 4.0", code_license ="GPL3")

# Add a license that reserves all of the author's rights
add_license("Team Bizzaro swirl", open_source_content = FALSE,
                                  open_source_data = FALSE,
                                  open_source_code = FALSE)

## End(Not run)
```

---

add_to_manifest *Add current lesson to course manifest*

Description

The MANIFEST file located in the course directory allows you to specify the order in which you'd like the lessons to be listed in swirl. If the MANIFEST file does not exist, lessons are listed alphabetically. Invisibly returns the path to the MANIFEST file.

Usage

```
add_to_manifest()
```

Value

MANIFEST file path, invisibly

Examples

```
## Not run:
# Check what lesson you're working on, then add it to the MANIFEST
get_current_lesson()
add_to_manifest()

## End(Not run)
```

`count_questions` *Count number of questions in current lesson*

Description

Returns and prints the number of questions in the current lesson.

Usage

```
count_questions()
```

Value

Number of questions as an integer, invisibly

Examples

```
## Not run:
count_questions()

## End(Not run)
```

`demo_lesson` *Demo the current lesson in swirl*

Description

Jump right in to the current swirl lesson without needing to navigate swirl's menus. It's also possible to jump into the middle of a lesson.

Usage

```
demo_lesson(from = NULL, to = NULL)
```

Arguments

`from`	Question number to begin with. Defaults to beginning of lesson.
`to`	Question number to end with. Defaults to end of lesson.

Examples

```
## Not run:
# Demo current lesson from beginning through the end
demo_lesson()
# Demo current lesson from question 5 through the end
demo_lesson(5)
# Demo current lesson from question 8 through question 14
demo_lesson(8, 14)

## End(Not run)
```

`find_questions` *Get question numbers for any questions matching a regular expression*

Description

Get question numbers for any questions matching a regular expression

Usage

```
find_questions(regex)
```

Arguments

`regex`	The regular expression to look for in the lesson. This gets passed along to `stringr::str_detect`, so the same rules apply. See `str_detect`.

Value

A vector of integers representing question numbers.

Examples

```
## Not run:
set_lesson()
find_questions("plot")
find_questions("which")

## End(Not run)
```

get_current_lesson *See what lesson you are currently working on*

Description

Prints the current lesson and course that you are working on to the console

Usage

```
get_current_lesson()
```

Examples

```
## Not run:
get_current_lesson()

## End(Not run)
```

google_form_decode *Decode Student's Submissions from Google Forms*

Description

Decode Student's Submissions from Google Forms

Usage

```
google_form_decode(path = file.choose())
```

Arguments

path	The path to a csv file downloaded from Google Forms or Google Sheets which contains student's encoded responses.

Value

A data frame containing each student's results.

Examples

```
## Not run:

# Choose the csv file yourself
google_form_decode()

# Explicity specify the path
google_form_decode("~/Desktop/My_Course.csv")


## End(Not run)
```

`lesson_to_html` *Turn a swirl lesson into a pretty webpage*

Description

Create an easily shareable HTML version of your swirl lesson. This function detects the lesson you are working on automatically via `getOption('swirlify_lesson_file_path')`, converts it to R Markdown (Rmd), then generates a stylized HTML document and opens it in your default browser. To prevent clutter, the Rmd files are not kept by default, but they can be kept by setting `keep_rmd = TRUE`.

Usage

```
lesson_to_html(dest_dir = NULL, open_html = FALSE, keep_rmd = FALSE,
  quiet = FALSE, install_course = TRUE)
```

Arguments

`dest_dir`	Destination directory (i.e. where to put the output files). If not set, default is the directory which contains the course directory.
`open_html`	Should the HTML file produced be opened in your browser? Default is `FALSE`.
`keep_rmd`	Should the Rmd file be kept after the HTML is is produced? Default is `FALSE`.
`quiet`	Should the Rmd rendering output be silenced? Default is `FALSE`.
`install_course`	Should the course be installed? Default is `TRUE`.

Details

The output is formatted to be a readable, standalone tutorial. This means that information contained in the swirl lesson such as answer tests and hints are excluded from the Rmd/HTML output.

new_lesson *Create new lesson in the YAML format.*

Description

Creates a new lesson and possibly a new course in your working directory. If the name you provide for `course_name` is not a directory in your working directory, then a new course directory will be created. However if you've already started a course with the name you provide for `course_name` and that course is in your working directory, then a new lesson will be created inside of that course with the name you provide for `lesson_name`.

Usage

```
new_lesson(lesson_name, course_name, open_lesson = TRUE)
```

Arguments

`lesson_name`	The name of the lesson.
`course_name`	The name of the course.
`open_lesson`	If `TRUE` the new `lesson.yaml` file will open for editing via `file.edit`. The default value is `TRUE`.

Examples

```
## Not run:
# Make sure you have your working directory set to where you want to
# create the course.
setwd(file.path("~", "Developer", "swirl_courses"))

# Make a new course with a new lesson
new_lesson("How to use pnorm", "Normal Distribution Functions in R")

# Make a new lesson in an existing course
new_lesson("How to use qnorm", "Normal Distribution Functions in R")

## End(Not run)
```

pack_course *Create an* `.swc` *file of the course you are working on*

Description

"Pack" the course you are working on into a single compressed file that is easy to share. Invisibly returns the path to the `.swc` file.

Usage

```
pack_course(export_path = NULL)
```

Arguments

export_path	Optional, full path to the directory you want the swirl course file to be exported to. If not specified, then the file will appear in the same directory as the course directory.

Value

A string, the path to the new .swc file, invisibly.

Examples

```
## Not run:
# Set any lesson in the course you want to pack
set_lesson()

# Pack the course
pack_course()

# Export the .swc file to a directory that you specify
pack_course(file.path("~", "Desktop"))

## End(Not run)
```

---

set_lesson *Select an existing lesson to work on*

Description

Choose a lesson to work on with swirlify by specifying the path to the lesson.yaml file or interactively choose a lesson file.

Usage

```
set_lesson(path_to_yaml = NULL, open_lesson = TRUE, silent = FALSE)
```

Arguments

path_to_yaml	Optional, full path to YAML lesson file. If not specified, then you will be prompted to select file interactively.
open_lesson	Should the lesson be opened automatically? Default is TRUE.
silent	Should the lesson be set silently? Default is FALSE.

Examples

```
## Not run:
# Set the lesson interactively
set_lesson()

# You can also specify the path to the \code{yaml} file you wish to work on.
set_lesson(file.path("~", "R_Programming", "Functions", "lesson.yaml"))

## End(Not run)
```

`swirlify` *Lauch a Shiny application for writing swirl lessons*

Description

This function launches a user interface for writing swirl lessons.

Usage

```
swirlify(lesson_name = NULL, course_name = NULL)
```

Arguments

`lesson_name`	The name of the new lesson you want to create. The default value is `NULL`. If you've already selected a lesson to work on using `set_lesson` then you do not need to provide a value for this argument.
`course_name`	The name of the new course you want to create. The default value is `NULL`. If you've already selected a course to work on using `set_lesson` then you do not need to provide a value for this argument.

Examples

```
## Not run:

# Set lesson beforehand
set_lesson()
swirlify()

# Start a new lesson in your current directory
swirlify("Lesson 1", "My Course")


## End(Not run)
```

testit *(Deprecated)*

Description

This function is deprecated. Please use `demo_lesson` instead.

Usage

```
testit(from = NULL, to = NULL)
```

Arguments

`from`	Question number to begin with. Defaults to beginning of lesson.
`to`	Question number to end with. Defaults to end of lesson.

test_course *Run tests on a course*

Description

Run basic tests on all questions in the lessons listed in the `MANIFEST`. See `add_to_manifest` for information about the `MANIFEST` file.

Usage

```
test_course()
```

Examples

```
## Not run:
# To test a course, set any lesson in that course as the current lesson
set_lesson()

# Run tests on every lesson in that course listed in the MANIFEST
test_course()

## End(Not run)
```

test_lesson *Run tests on a lesson*

Description

Run basic tests on all questions in the current lesson.

Usage

```
test_lesson()
```

Examples

```
## Not run:
# Set a lesson interactively
set_lesson()

# Run tests on that lesson
test_lesson()

## End(Not run)
```

unpack_course *Unpack an* .swc *file into a swirl course*

Description

Invisibly returns the path to the unpacked course directory.

Usage

```
unpack_course(file_path = file.choose(), export_path = dirname(file_path))
```

Arguments

file_path	Optional, full path to the .swc file you wish to unpack. If not specified, you will be prompted to choose a file interactively.
export_path	Optional, full path to the directory where the swirl course should be exported. If not specified, the course will appear in the same directory as the .swc file.

Value

A string, the path to the unpacked course directory, invisibly.

Examples

```
## Not run:
# Unpack a course and interactively choose a .swc file
unpack_course()

# Unpack a course where the .swc file is explicitly specified
unpack_course(file.path("~", "Desktop", "R_Programming.swc"))

# Unpack a course and specify where the .swc file is located and where the
# course should be exported.
unpack_course(file.path("~", "Desktop", "R_Programming.swc"),
 file.path("~", "Developer", "swirl"))

## End(Not run)
```

---

wq_command *Template for R command question*

Description

Template for R command question

Usage

```
wq_command(output = "explain what the user must do here",
  correct_answer = "EXPR or VAL",
  answer_tests = "omnitest(correctExpr='EXPR', correctVal=VAL)",
  hint = "hint")
```

Arguments

output	Text that is displayed to the user.
correct_answer	A string that designates the correct answer, in this case an R expression or a value.
answer_tests	An internal function from swirl for testing the user's choice. See AnswerTests.
hint	A string that is printed to the console if the user answers this question incorrectly.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_command()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_command("Assign the value 5 to the variable x.",
 "x <- 5", "omnitest(correctExpr='x <- 5')", "Just type: x <- 5")

## End(Not run)
```

wq_figure *Template for figure questions*

Description

Template for figure questions

Usage

```
wq_figure(output = "explain the figure here", figure = "sourcefile.R",
  figure_type = "new")
```

Arguments

output	Text that is displayed to the user.
figure	An R script that produces a figure that is displayed in the R plotting window.
figure_type	Either "new" or "add". "new" idicates that a new plot should be displayed, while "add" indicates that features are being added to a plot that is already displayed.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_figure()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_figure("Here we can see the curve of the normal distribution.",
 "normalplot.R", "new")

## End(Not run)
```

wq_message *Template for output without a question*

Description

Template for output without a question

Usage

```
wq_message(output = "put your text output here")
```

Arguments

output	Text that is displayed to the user.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_message()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_message("Welcome to a course on the central limit theorem.")

## End(Not run)
```

`wq_multiple` *Template for multiple choice question*

Description

Template for multiple choice question

Usage

```
wq_multiple(output = "ask the multiple choice question here",
  answer_choices = c("ANS", "2", "3"), correct_answer = "ANS",
  answer_tests = "omnitest(correctVal= 'ANS')", hint = "hint")
```

Arguments

`output`	Text that is displayed to the user.
`answer_choices`	A vector of strings containing a user's choices.
`correct_answer`	A string that designates the correct answer.
`answer_tests`	An internal function from `swirl` for testing the user's choice. See `AnswerTests`.
`hint`	A string that is printed to the console if the user answers this question incorrectly.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_multiple()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_multiple("Which of the following is not a planet in our solar system?",
 c("Venus", "Saturn", "Pluto"), "Pluto", "omnitest(correctVal= 'Pluto')",
   "It's the smallest celestial body you can choose.")

## End(Not run)
```

wq_numerical *Template for exact numerical question*

Description

Template for exact numerical question

Usage

```
wq_numerical(output = "explain the question here", correct_answer = "42",
  answer_tests = "omnitest(correctVal=42)", hint = "hint")
```

Arguments

output	Text that is displayed to the user.
correct_answer	The numerical answer to the question.
answer_tests	An internal function from swirl for testing the user's choice. See AnswerTests.
hint	A string that is printed to the console if the user answers this question incorrectly.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_numerical()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_numerical("The golden ratio is closest to what integer?",
 "2", "omnitest(correctVal=2)", "It's greater than 1 and less than 3.")

## End(Not run)
```

wq_script *Template for R script question*

Description

Template for R script question

Usage

```
wq_script(output = "explain what the user must do here",
  answer_tests = "custom_test_name()", hint = "hint",
  script = "script-name.R")
```

Arguments

output	Text that is displayed to the user.
answer_tests	An internal function from swirl for testing the user's choice. See AnswerTests.
hint	A string that is printed to the console if the user answers this question incorrectly.
script	The name of the script template to be opened. This template should be in a directory called scripts located inside the lesson directory.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_script()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_script("Write a function that adds three numbers.",
 "add_three_test()", "Something like: add3 <- function(x, y, z){x+y+z}",
 "add-three.R")

## End(Not run)
```

---

wq_text *Template for text question*

Description

Template for text question

Usage

```
wq_text(output = "explain the question here", correct_answer = "answer",
  answer_tests = "omnitest(correctVal='answer')", hint = "hint")
```

Arguments

output	Text that is displayed to the user.
correct_answer	The answer to the question in the form of a string.
answer_tests	An internal function from swirl for testing the user's choice. See AnswerTests.
hint	A string that is printed to the console if the user answers this question incorrectly.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_text()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_text("Where is the Johns Hopkins Bloomberg School of Public Health located?",
 "Baltimore", "omnitest(correctVal='Baltimore')", "North of Washington, south of Philadelphia.")

## End(Not run)
```

`wq_video` *Template for video question*

Description

The `url` provided for `video_link` can be a link to any website.

Usage

```
wq_video(output = "Would you like to watch a short video about ___?",
  video_link = "http://address.of.video")
```

Arguments

`output`	Text that is displayed to the user.
`video_link`	A link to a url. Please make sure to use `http://` or `https://`.

Examples

```
## Not run:
# While writing a new lesson by hand just use:
wq_video()

# If converting from another format to a swirl course you may want to sue the
# API:
wq_video("Now Roger will show you the basics on YouTube.",
 "https://youtu.be/dQw4w9WgXcQ")

## End(Not run)
```

Index