Package ‘ottrpal’

January 9, 2024

Type Package

Title Companion Tools for Open-Source Tools for Training Resources (OTTR)

Version 1.2.1

Description Tools for converting Open-Source Tools for Training Resources (OTTR) courses into Leanpub or Coursera courses. ‘ottrpal’ is for use with the OTTR Template repository to create courses.

License GPL-3

URL https://github.com/jhudsl/ottrpal

BugReports https://github.com/jhudsl/ottrpal/issues

Depends R (>= 3.5.0)

Imports bookdown, curl, dplyr, fs, glue, httr, jsonlite, knitr (>= 1.33), magrittr, openssl, purrr, R.utils, readr, rmarkdown (>= 2.10), rprojroot, rvest, stringr, xml2, yaml

Suggests remotes, testthat, tibble, utils

VignetteBuilder knitr

ByteCompile true

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Candace Savonen [aut, cre] (<https://orcid.org/0000-0001-6331-7070>), John Muschelli [aut] (<https://orcid.org/0000-0001-6469-1750>), Carrie Wright [ctb], Howard Baek [ctb]

Maintainer Candace Savonen <cansav09@gmail.com>

Repository CRAN

Date/Publication 2024-01-09 20:00:05 UTC
R topics documented:

- authorize .......................................................... 3
- auth_from_secret .................................................. 3
- bad_quiz_path ...................................................... 4
- bookdown_destination ............................................ 5
- bookdown_file ....................................................... 5
- bookdown_path ....................................................... 6
- bookdown_rmd_files .............................................. 6
- bookdown_to_book_txt ............................................ 7
- bookdown_to_embed_leanpub ..................................... 7
- bookdown_to_leanpub ............................................. 9
- check_all_questions ................................................ 10
- check_question .................................................... 11
- check_quiz .......................................................... 12
- check_quizzes ....................................................... 13
- check_quiz_attributes ........................................... 14
- check_quiz_question_attributes ................................ 14
- convert_coursera_quizzes ....................................... 15
- convert_footnotes ................................................ 16
- convert_quiz ........................................................ 16
- convert_utube_link ............................................... 17
- encrypt_creds_path ............................................... 17
- encrypt_creds_user_path ....................................... 18
- example_repo_cleanup .......................................... 18
- example_repo_setup .............................................. 19
- extract_meta ....................................................... 19
- extract_object_id ................................................ 20
- get_bookdown_spec ............................................... 21
- get_chapters ....................................................... 22
- get_gs_pptx ......................................................... 22
- get_object_id_notes ............................................. 23
- get_slide_id ........................................................ 23
- good_quiz_path .................................................... 24
- gs_id_from_slide .................................................. 24
- gs_png_url .......................................................... 25
- key_encrypt_creds_path ........................................ 26
- leanpub_check ...................................................... 26
- make_embed_markdown ........................................... 27
- parse_quiz .......................................................... 28
- parse_quiz_df ...................................................... 29
- parse_q_tag .......................................................... 29
- pptx_notes ........................................................... 30
- remove_yaml_header .............................................. 31
- render_without_toc ............................................... 31
- replace_html ......................................................... 32
- set_knitr_image_path ........................................... 33
- set_up_leanpub ..................................................... 34
authorize

Authorize R package to access the Google Slides API

Description
This is a function to authorize the R package to access the Google Slides API interactively.

Usage
authorize(token = NULL, cache = FALSE, ...)

Arguments
- **token**: An output from `oauth2.0_token` to set as the authentication token.
- **cache**: Should the token be cached as an .httr-oauth file?
- **...**: Additional arguments to send to `oauth2.0_token`

Value
OAuth token saved to the environment so the package can use the users’ Google data

Examples
```r
## Not run:
authorize()
## End(Not run)
```

auth_from_secret

Use secrets to authorize R package to access Google Slides API

Description
This is a function to authorize the R package to access the Google Slides API. If no client.id and client.secret is provided, the package would provide predefined values.

Usage
auth_from_secret(access_token = NULL, refresh_token = NULL)
Arguments

access_token  Access token can be obtained from running authorize() interactively: token $<-
authorize(); token$credentials$access_token

refresh_token  Refresh token can be obtained from running authorize() interactively: token $<-
authorize(); token$credentials$refresh_token

Value

OAuth token saved to the environment so the package can use the users’ Google data

Examples

### Not run:

```r
token <- authorize()

auth_from_secret(
  token$credentials$access_token,
  token$credentials$refresh_token
)

## End(Not run)
```

bad_quiz_path  

Path to bad example quiz

Description

Path to bad example quiz

Usage

bad_quiz_path()

Value

The file path to an example bad quiz included in the package that will fail the quiz checks.

Examples

```r
quiz_path <- bad_quiz_path()
```
bookdown_destination  
 Declare file path to docs/ folder

Description

Declare file path to docs/ folder

Usage

bookdown_destination(path = ".")

Arguments

path  Where to look for the _bookdown.yml file. Passes to get_bookdown_spec() function. By default looks in current directory

Value

The file paths to Rmds listed in the _bookdown.yml file.

bookdown_file  
 Find file path to _bookdown.yml

Description

Find file path to _bookdown.yml

Usage

bookdown_file(path = ".")

Arguments

path  Where to look for the _bookdown.yml file. Passes to the bookdown_file() function. By default looks in current directory

Value

The file path to _bookdown.yml
bookdown_path  
Find main Bookdown directory

Description
Find main Bookdown directory

Usage
bookdown_path(path = ".")

Arguments
path Where to look for the file. By default looks in current directory.

Value
Returns the directory where the _bookdown.yml is contained.

bookdown_rmd_files  
Get file paths all Rmds in the bookdown directory

Description
Get file paths all Rmds in the bookdown directory

Usage
bookdown_rmd_files(path = ".")

Arguments
path Where to look for the _bookdown.yml file. Passes to get_bookdown_spec() function. By default looks in current directory

Value
The file paths to Rmds listed in the _bookdown.yml file.
Create Book.txt file from files existing in quiz directory

Usage

bookdown_to_book_txt(
  path = ".", 
  md_files = NULL, 
  output_dir = "manuscript", 
  quiz_dir = "quizzes", 
  verbose = TRUE
)

Arguments

path    path to the bookdown book, must have a `_bookdown.yml` file
md_files vector of file path of the md’s to be included
output_dir output directory to put files. It should likely be relative to path
quiz_dir Where are the quizzes stored? Default looks for folder called "quizzes".
verbose print diagnostic messages

Value

A list of quiz and chapter files in order in a file called Book.txt – How Leanpub wants it.

Convert Bookdown to Embed version of Leanpub

Convert Bookdown to Embed version of Leanpub
Usage

bookdown_to_embed_leanpub(
  path = ".",
  chapt_img_key = NULL,
  bookdown_index = file.path(base_url, "index.html"),
  base_url = NULL,
  clean_up = FALSE,
  default_img = NULL,
  render = TRUE,
  output_dir = "manuscript",
  make_book_txt = FALSE,
  quiz_dir = "quizzes",
  run_quiz_checks = FALSE,
  remove_resources_start = FALSE,
  verbose = TRUE,
  footer_text = ""
)

Arguments

path: path to the bookdown book, must have a `.bookdown.yml` file

chapt_img_key: File path to a TSV whose contents are the chapter urls (`url`), the chapter titles (`chapt_title`), the file path to the image to be used for the chapter (`img_path`). Column names `url`, `chapt_title`, and `img_path` must be used. If no chapter title column supplied, the basename of the url will be used. If no image column supplied, default image used.

bookdown_index: The file path of the rendered bookdown index.html file

default_img: A google slide link to the default image to be used for all chapters

base_url: The base url of where the chapters are published – the url to provide to the iframe in Leanpub e.g. https://jhudatascience.org/OTTR_Template/coursera

clean_up: TRUE/FALSE the old output directory should be deleted and everything created fresh.

render: if `TRUE`, then [bookdown::render_book()] will be run on each Rmd.

output_dir: output directory to put files. It should likely be relative to path

make_book_txt: Should [ottrpal::bookdown_to_book_txt()] be run to create a `Book.txt` in the output directory?

quiz_dir: directory that contains the quiz .md files that should be checked and incorporated into the Book.txt file. If you don’t have quizzes, set this to NULL

run_quiz_checks: TRUE/FALSE run quiz checks

remove_resources_start: remove the word 'resources/' at the front of any image path.

verbose: print diagnostic messages

footer_text: Optionally can add a bit of text that will be added to the end of each file before the references section.
Value

A directory of output files in a folder `manuscript` for publishing on Leanpub.

Examples

```r
## Not run:

ottrpal::bookdown_to_embed_leanpub(
  base_url = "https://jhudatascience.org/OTTR_Template/",
  make_book_txt = TRUE,
  quiz_dir = NULL
)

## End(Not run)
```

bookdown_to_leanpub  

Convert Bookdown to Leanpub

Description

Convert Bookdown to Leanpub

Usage

```r
bookdown_to_leanpub(
  path = ".",
  render = TRUE,
  output_dir = "manuscript",
  make_book_txt = FALSE,
  quiz_dir = "quizzes",
  run_quiz_checks = FALSE,
  remove_resources_start = FALSE,
  verbose = TRUE,
  footer_text = NULL,
  clean_up = FALSE
)
```

Arguments

- **path**: path to the bookdown book, must have a `_bookdown.yml` file
- **render**: if `TRUE`, then [bookdown::render_book()](https://bookdown.org/ottrpal/reference/bookdown_to_bookdown.html) will be run on each Rmd.
- **output_dir**: output directory to put files. It should likely be relative to path
- **make_book_txt**: Should [ottrpal::bookdown_to_book_txt()](https://bookdown.org/ottrpal/reference/bookdown_to_bookdown.html) be run to create a `Book.txt` in the output directory?
- **quiz_dir**: directory that contains the quiz .md files that should be checked and incorporated into the Book.txt file. If you don’t have quizzes, set this to NULL.
check_all_questions

run_quiz_checks
TRUE/FALSE run quiz checks

remove_resources_start
remove the word 'resources/' at the front of any image path.

verbose
print diagnostic messages

footer_text
Optionally can add a bit of text that will be added to the end of each file before the references section.

clean_up
TRUE/FALSE the old output directory should be deleted and everything created fresh.

Value
A list of output files and diagnostics

check_all_questions  Check all quiz questions

Description
Takes output from [ottrpal::parse_quiz] and runs checks on each question in a quiz by calling [ottrpal::check_question] for each question. First splits questions into their own data frame. Returns a list of messages/warnings about each question’s set up.

Usage
check_all_questions(
  quiz_specs,
  quiz_name = NA,
  verbose = TRUE,
  ignore_coursera = TRUE
)

Arguments
quiz_specs  quiz_specs which is output from [ottrpal::parse_quiz].
quiz_name   The name of the quiz being checked.
verbose     Whether progress messages should be given.
ignore_coursera  Coursera doesn’t like ‘;’ or ‘:’ in the quizzes. Do not convert quizzes to coursera and ignore ! and : in question prompts that would not be allowed in Leanpub quizzes when converted to a Coursera quiz. Default is to ignore Coursera compatibility.

Value
A list of the output from [ottrpal::check_question] with messages/warnings regarding each question and each check.
check_question

**Examples**

```r
## Not run:

# Using good quiz md example

quiz_path <- good_quiz_path()
good_quiz <- readLines(quiz_path)
good_quiz_specs <- parse_quiz(good_quiz)
good_quiz_checks <- check_all_questions(good_quiz_specs)

# Using bad quiz md example

bad_quiz <- readLines(bad_quiz_path())
bad_quiz_specs <- parse_quiz(bad_quiz)
bad_quiz_checks <- check_all_questions(bad_quiz_specs)

## End(Not run)
```

**check_question**

*Check Quiz Question Set Up*

**Description**

Check quiz question set up to see if it is compliant with Leanpub and Coursera needs. Based off of [Markua guide](https://leanpub.com/markua/read#leanpub-auto-quizzes-and-exercises). Is called by [ottrpal::check_all_questions] and run for each question.

**Usage**

```r
check_question(
  question_df,
  quiz_name = NA,
  verbose = TRUE,
  ignore_coursera = TRUE
)
```

**Arguments**

- `question_df` Which is an individual question’s data frame after being parse from
- `quiz_name` The name of the quiz the question is from
- `verbose` Whether progress messages should be given
- `ignore_coursera` Coursera doesn’t like ‘;’ or ‘:’ in the quizzes. Do not convert quizzes to coursera and ignore ! and : in question prompts that would not be allowed in Leanpub quizzes when converted to a Coursera quiz. Default is to ignore Coursera compatibility
check_quiz

Value

A list of messages/warnings regarding each check for the given question.

Examples

```r
## Not run:

# Use readLines to read in a quiz
quiz_path <- good_quiz_path()
quiz_lines <- readLines(quiz_path)

# Use group_split to get the questions
questions_df <- parse_quiz(quiz_lines)$data %>%
    dplyr::group_split(question)

good_quiz_checks <- check_question(questions_df[[2]])

## End(Not run)
```

---

check_quiz  

Check Quiz

Description

For a file path to a quiz, check whether it is properly formatted for Leanpub.

Usage

```r
check_quiz(quiz_path, verbose = TRUE, ignore_coursera = TRUE)
```

Arguments

- **quiz_path**  
  A file path to a quiz markdown file
- **verbose**  
  print diagnostic messages? TRUE/FALSE
- **ignore_coursera**  
  Coursera doesn’t like ’; or ’:’ in the quizzes. Do not convert quizzes to coursera and ignore ! and : in question prompts that would not be allowed in Leanpub quizzes when converted to a Coursera quiz. Default is to ignore Coursera compatibility

Value

A list of checks. "good" means the check passed. Failed checks will report where it failed.
check_quizzes

Examples

## Not run:

# Take a look at a good quiz's checks:
quiz_path <- good_quiz_path()
good_checks <- check_quiz(quiz_path)

# Take a look at a failed quiz's checks:
quiz_path <- bad_quiz_path()
failed_checks <- check_quiz(quiz_path)

## End(Not run)

---

check_quizzes  

Check all quizzes in a directory

Description

Check the formatting of all quizzes in a given directory.

Usage

check_quizzes(
  quiz_dir = "quizzes",
  write_report = TRUE,
  verbose = TRUE,
  ignore_coursera = TRUE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quiz_dir</td>
<td>A path to a directory full of quizzes that should all be checked with [ottr-pal::check_all_quizzes].</td>
</tr>
<tr>
<td>write_report</td>
<td>TRUE/FALSE save warning report to a CSV file?</td>
</tr>
<tr>
<td>verbose</td>
<td>print diagnostic messages</td>
</tr>
<tr>
<td>ignore_coursera</td>
<td>Coursera doesn’t like ‘;’ or ‘:’ in the quizzes. Do not convert quizzes to coursera and ignore ! and : in question prompts that would not be allowed in Leanpub quizzes when converted to a Coursera quiz. Default is to ignore Coursera compatibility</td>
</tr>
</tbody>
</table>

Value

A list checks performed on each quiz
check_quiz_question_attributes

Examples

## Not run:

## Make a temporary quiz directory
quiz_dir <- dirname(good_quiz_path())

## Now check the quizzes in that directory
all_quiz_results <- check_quizzes(quiz_dir = quiz_dir)

## End(Not run)

check_quiz_attributes  Check Quiz Attributes

Description

Check Quiz Attributes

Usage

check_quiz_attributes(quiz_specs, quiz_name = NULL, verbose = TRUE)

Arguments

- **quiz_specs**: The output from `ottrpal::parse_quiz`.
- **quiz_name**: A character string indicating the name of the quiz being checked.
- **verbose**: Would you like progress messages? TRUE/FALSE

Value

A logical

check_quiz_question_attributes  Check a question's attributes

Description

This is ran automatically by `ottrpal::check_all_questions` for all questions. It checks that the attributes specified are accepted ones by Leanpub.

Usage

check_quiz_question_attributes(question_df, quiz_name = NULL, verbose = TRUE)
convert_coursera_quizzes

Arguments

question_df  a data.frame obtained from [ottrpal::parse_quiz_df] and dplyr::group_split(question).
quiz_name    inherited from parse
verbose      print diagnostic messages

Value

Will return a warning for any quiz question attributes used that are not supported.

Description

Convert Leanpub md quiz to Coursera yaml quiz

Usage

convert_coursera_quizzes(
  input_quiz_dir = "quizzes",
  output_quiz_dir = "coursera_quizzes",
  verbose = TRUE
)

Arguments

input_quiz_dir  A path to a directory of leanpub formatted quiz md files. By default assumes
                "quizzes" and looks in current directory.
output_quiz_dir A folder (existing or not) that the new coursera converted quizzes should be
                saved to. By default saves to "coursera_quizzes".
verbose         Would you like the progress messages: TRUE/FALSE?

Value

A folder of coursera ready quiz files saved to the output directory specified as a yamls.

Examples

# Set up a directory with a quiz in it for this example
tdir <- tempfile()
dir.create(tdir, showWarnings = FALSE, recursive = TRUE)

file.copy(
  from = good_quiz_path(),
convert_quiz

    to = file.path(tdir, basename(good_quiz_path()))
    )

# Provide path to directory of quizzes
convert_coursera_quizzes(tdir)

system("rm -r coursera_quizzes")

convert_footnotes

---

**convert_footnotes**

*Convert Bookdown footnotes to Leanpub-formatted footnotes*

**Description**

Convert Bookdown footnotes to Leanpub-formatted footnotes

**Usage**

```r
convert_footnotes(content)
```

**Arguments**

- `content`: a character vector containing the lines of content from a file read in with `readLines()`

**Value**

a character vector containing the content given but with Leanpub formatted footnotes

---

**convert_quiz**

*Convert Leanpub md quiz to Coursera yaml quiz*

**Description**

Convert a Leanpub-formatted md quiz file to a Coursera-formatted yaml quiz file in preparation for uploading to Coursera.

**Usage**

```r
convert_quiz(quiz_path, output_quiz_dir = dirname(quiz_path), verbose = TRUE)
```

**Arguments**

- `quiz_path`: A path to a quiz .md file to be converted.
- `output_quiz_dir`: An existing folder where you would like the new version of the quiz to be saved. Default is the directory of the quiz_path provided
- `verbose`: Would you like the progress messages?
Value

A Coursera-ready quiz file saved to the output directory specified as a yaml.

Examples

```r
## Not run:
quiz_path <- good_quiz_path()

# Provide path to quiz to convert
convert_quiz(quiz_path)

## End(Not run)
```

---

**convert_utube_link**  
Convert youtube link

**Description**

Convert youtube link

**Usage**

```r
convert_utube_link(utube_link)
```

**Arguments**

- `utube_link` a link to a youtube video that may or may not be "www.youtube.com/embed" or "www.youtube.com/watch?v=" format.

**Value**

Returns a youtube link in the "watch" format so it will render properly in Leanpub or Coursera-friendly files

---

**encrypt_creds_path**  
Get file path to an encrypted credentials RDS

**Description**

Get file path to an encrypted credentials RDS

**Usage**

```r
encrypt_creds_path()
```
encrypt_creds_user_path

Get file path to an default credentials RDS

Description
Get file path to an default credentials RDS

Usage
encrypt_creds_user_path()

example_repo_cleanup  Clean up example repo files

Description
Clean up example repo files

Usage
example_repo_cleanup(files_to_remove, verbose = FALSE)

Arguments
files_to_remove  List of example files to delete.
verbose        TRUE/FALSE would you like progress messages?

Value
Will delete example files copied from [ottrpal::example_repo_setup()] function

Examples
## Not run:
# Run this to get the files we need
downloaded_files <- ottrpal::example_repo_setup()

# Run this to delete them
downloaded_files <- ottrpal::example_repo_setup(files_to_remove = basename(example_files))

## End(Not run)
example_repo_setup  

Set up example repo files

Description

Set up example repo files

Usage

example_repo_setup(dest_dir = tempdir())

Arguments

dest_dir  
The destination directory you would like the example repo files to be placed. By default is current directory.

Value

Sets up example files that can be used to test `ottrpal` functions.

Examples

## Not run:

# Run this to get the files we need
example_files <- ottrpal::example_repo_setup()

## End(Not run)

extract_meta  

Extract meta fields from a tag

Description

Extract meta fields from a tag

Usage

extract_meta(tags)

Arguments

tags  
A single tag or vector of tags to extract the fields from.

Value

A named vector indicating the field and entry associated with it.
Examples

```r
### Simple example
tag <- "{quiz, id: quiz_name_here, attempts: 10}"

# Extract metadata tags
meta <- extract_meta(tag)

### Example using a file
quiz_path <- good_quiz_path()
quiz_lines <- readLines(quiz_path)

# Put this in a data.frame so we can identify the content
quiz_df <- parse_quiz_df(quiz_lines)

# Extract the tags
tags <- quiz_df %>%
   dplyr::filter(type == "tag") %>%
   dplyr::pull("original")

# Extract metadata tags
meta <- extract_meta(tags)
```

---

**extract_object_id**

*Extract Object IDs using Google Slides API*

**Description**

Performs a HTTP GET method to request the IDs of every slide in a Google Slides presentation. The ID of the first slide is always 'p'.

**Usage**

```r
extract_object_id(
  slide_url,
  token = NULL,
  access_token = NULL,
  refresh_token = NULL
)
```

**Arguments**

- **slide_url** URL whose 'General access' is set to 'Anyone with the link'
- **token** OAuth 2.0 Access Token. If you don’t have a token, use [authorize()] to obtain an access token from Google’s OAuth 2.0 server.
- **access_token** Access token can be obtained from running authorize() interactively (token <- authorize(); token$credentials$access_token). This allows it to be passed in using two secrets.
refresh_token  Refresh token can be obtained from running authorize() interactively (token <-
authorize(); token$credentials$refresh_token). This allows it to be passed in
using two secrets.

Value

Character vector of object ID(s)

Examples

```r
## Not run:
# First, obtain access token and store token for extract_object_id() to use
authorize(client_id = "MY_CLIENT_ID", client_secret = "MY_CLIENT_SECRET")
# Use stored token to talk to Google Slides API
extract_object_id(slide_url = "https://docs.google.com/presentation/d/1H5aF_ROKvxE-H
FHhoOy9vU2Y-yZM_PiV0q-JBL17Gss/edit?usp=sharing")
```

## End(Not run)

---

**get_bookdown_spec**  
*Load in Bookdown specifications from _bookdown.yml*

**Description**

Load in Bookdown specifications from _bookdown.yml

**Usage**

```r
get_bookdown_spec(path = ".")
```

**Arguments**

`path`  
Where to look for the _bookdown.yml file. Passes to the bookdown_file() function. By default looks in current directory

**Value**

The yaml contents using yaml::yaml.load_file()
get_chapters

Make Leanpub file that has embed webpage of a chapter

Description

Make Leanpub file that has embed webpage of a chapter

Usage

get_chapters(bookdown_index = file.path("docs", "index.html"), base_url = NULL)

Arguments

bookdown_index The file path of the rendered bookdown index.html file
base_url The base url of where the chapters are published – the url to provide to the iframe in Leanpub e.g. https://jhudatascience.org/OTTR_Template/coursera

Value

A data.frame of the chapter urls and their titles that are to be ported to Leanpub. This can be passed to

get_gs_pptx

Download Google Slides pptx file

Description

Download Google Slides pptx file

Usage

get_gs_pptx(id)

Arguments

id Identifier of Google slides presentation, passed to get_slide_id

Value

Downloaded file (in temporary directory)

Note

This downloads presentations if they are public and also try to make sure it does not fail on large files
get_object_id_notes

Retrieve Speaker Notes and their corresponding Object (Slide) IDs from a Google Slides presentation

Description
Google Slides API calls a presentation slide ID as an ’object ID’.

Usage
get_object_id_notes(slide_url)

Arguments
slide_url URL whose ’General access’ is set to ’Anyone with the link’

Value
Data frame of Object IDs and Speaker notes.

Examples
## Not run:
get_object_id_notes("https://docs.google.com/presentation/d/1H5aF_ROKVxExHFHo0YvJY2Y-y2M_PiV0q-JBL17Gss/edit?usp=sharing")
## End(Not run)

get_slide_id
Get Slide ID from URL

Description
Get Slide ID from URL

Usage
get_slide_id(x)

Arguments
x URL of slide

Value
A character vector
Examples

```r
x <- paste0(
    "https://docs.google.com/presentation/d/",
    "1Tg-GTGnUPdu0tZKYMoelqUNZnUp3vvg_7TtpUPL7e8",
    "/edit#slide=id.g154aa4fae2_0_58"
)
gs_id_from_slide(x)
```

---

**good_quiz_path**

*Path to good example quiz*

---

**Description**

Path to good example quiz

**Usage**

`good_quiz_path()`

**Value**

The file path to an example good quiz included in the package that will pass the quiz checks.

**Examples**

```r
quiz_path <- good_quiz_path()
```

---

**gs_id_from_slide**

*Google Slides Helper Functions*

---

**Description**

Google Slides Helper Functions

**Usage**

`gs_id_from_slide(file)`

`get_image_link_from_slide(file)`

`get_image_from_slide(file)`

**Arguments**

- `file` (markdown file for manuscript)
gs_png_url

Value

A scalar character vector

Description

Get Google Slide PNG URL

Usage

gs_png_url(url)
get_slide_page(url)

gs_png_download(url, output_dir = ".", overwrite = TRUE)

include_slide(
    url,
    output_dir = knitr::opts_chunk$get("fig.path"),
    overwrite = TRUE,
    ...
)

Arguments

url URL to Google Slide
output_dir path to output png
overwrite should the slide PNG be overwritten?
... for include_slide, options passed to [knitr::include_graphics()]

Value

A character vector of URLs

Examples

url <- paste0(
    "https://docs.google.com/presentation/d/",
    "12DPZgPteQ8wgal6kSPP58zhPhjZ7QSPZLe3NKA8M3eo/edit",
    
    
    
    
    "#slide=gc8648f14c3_0_397&t=4"
)

id <- get_slide_id(url)
gs_png_url(url)
key_encrypt_creds_path

*Get file path to an key encryption RDS*

---

**Description**

Get file path to an key encryption RDS

**Usage**

```r
key_encrypt_creds_path()
```

---

**leanpub_check**

*Check Leanpub Course or Book*

---

**Description**

Check Leanpub Course or Book

**Usage**

```r
leanpub_check(path = ".", verbose = TRUE)
```

**Arguments**

- `path` — path to the Leanpub book/course
- `verbose` — print diagnostic messages

**Value**

A list of output files and diagnostics
make_embed_markdown

Description

Make Leanpub file that has embed webpage of a chapter

Usage

make_embed_markdown(
  url,
  chapt_title,
  width_spec = 800,
  height_spec = 600,
  img_path,
  output_dir = "manuscript",
  verbose = TRUE,
  footer_text = ""
)

Arguments

url The url to the chapter that is to be embed
chapt_title Title of chapter to be used as file name and printed on iframe
width_spec How wide should the iframe be in pixels?
height_spec How high should the iframe be in pixels?
img_path File path to image to use for poster
output_dir output directory to put files. It should likely be relative to path
verbose print diagnostic messages
footer_text Optionally can add a bit of text that will be added to the end of each file before the references section.

Value

A markdown file with an iframe of the provided chapter
parse_quiz  

Parse Quiz and Other Checking Functions

Description

Parse Quiz and Other Checking Functions

Extract lines of the quiz

Usage

parse_quiz(quiz_lines, quiz_name = NULL, verbose = FALSE)

extract_quiz(quiz_lines)

Arguments

quiz_lines  
A quiz’s contents read in with readLines()

quiz_name  
A character vector indicating the name of the quiz.

verbose  
Would you like progress messages? TRUE/FALSE

Value

A list of elements, including a ‘data.frame’ and metadata for questions

the lines of the quiz that actually contain the content of the quiz.

Examples

```r
quiz_lines <- c(
  "(quiz, id: quiz_00_filename)",
  "### Lesson Name quiz",
  "(choose-answers: 4)",
  "? What do you think?",
  "",
  "C) The answer to this one",
  "o) Not the answer",
  "o) Not the answer either",
  "C) Another correct answer",
  "m) Mandatory different answer",
  "",
  "(/quiz)"
)
quiz_specs <- parse_quiz(quiz_lines)
check_quiz_attributes(quiz_specs)
```
parse_quiz_df

Parse quiz into a data.frame

Description

Parse quiz into a data.frame

Usage

parse_quiz_df(quiz_lines, remove_tags = FALSE)

Arguments

quiz_lines A character vector of the contents of the markdown file obtained from readLines()
remove_tags TRUE/FALSE remove tags and empty lines?

Value

A data frame containing a type column which indicates what type of line each is.

Examples

## Not run:

# Use readLines() to read in a quiz
quiz_path <- good_quiz_path()
quiz_lines <- readLines(quiz_path)

# Can use this to parse the quiz into a data.frame
quiz_df <- parse_quiz_df(quiz_lines)

## End(Not run)

parse_q_tag

Parse apart a tag

Description

Parse apart a tag

Usage

parse_q_tag(tag)
**Arguments**

- **tag**: A single tag to extract from

**Value**

A named vector indicating the field and entry associated with it.

**Examples**

```r
tag <- "{quiz, id: quiz_name_here, attempts: 10}"
parse_q_tag(tag)
```

---

**Description**

Get Notes from a PowerPoint (usually from Google Slides)

**Usage**

```r
pptx_notes(file, ...)
pptx_slide_text_df(file, ...)
pptx_slide_note_df(file, ...)
unzip_pptx(file)
```

**Arguments**

- **file**: Character. Path for ‘PPTX’ file
- **...**: additional arguments to pass to `xml_notes`, particularly `xpath`

**Value**

Either a character vector or ‘NULL’

**Examples**

```r
## Not run:
pptx_notes(ex_file)
pptx_slide_note_df(ex_file)
pptx_slide_text_df(ex_file)
## End(Not run)
```
remove_yaml_header

Remove YAML header

Description
Remove YAML header

Usage
remove_yaml_header(file)

Arguments
file file name of the markdown file

Value
A character vector of the text without the YAML header

Examples
file <- system.file("extdata/00_template.Rmd", package = "ottrpal")
out <- remove_yaml_header(file)
head(out)

render_without_toc Create TOC-less Bookdown for use in Coursera

Description
Create a version of Leanpub that does not have a TOC and has quizzes in the Coursera yaml format. Requires Bookdown output files including "assets", "resources", and "libs".

Usage
render_without_toc(
  output_dir = file.path("docs", "no_toc"),
  output_yaml = "/output.yml",
  convert_quizzes = FALSE,
  input_quiz_dir = "quizzes",
  output_quiz_dir = "coursera_quizzes",
  verbose = TRUE
)

render_without_toc
replace_html

Arguments

- **output_dir**: A folder (existing or not) that the TOC-less Bookdown for Coursera files should be saved. By default is `file.path("docs", "coursera")`
- **output_yaml**: A output.yml file to be provided to bookdown. By default is ".output.yml"
- **convert_quizzes**: TRUE/FALSE whether or not to convert quizzes. Default is TRUE
- **input_quiz_dir**: A path to a directory of Leanpub-formatted quiz md files. By default assumes "quizzes" and looks in current directory.
- **output_quiz_dir**: A folder (existing or not) where the coursera quizzes should be saved. By default is "coursera_quizzes".
- **verbose**: Would you like the progress messages? TRUE/FALSE

Value

A folder of coursera ready quiz files and html chapter files saved to output directories specified.

---

replace_html

*Replace HTML and other Tags in Leanpub Markdown*

Description

Replace HTML and other Tags in Leanpub Markdown

Usage

```r
replace_html(
  path = "manuscript",
  remove_resources_start = TRUE,
  footer_text = NULL,
  fullbleed = FALSE,
  verbose = TRUE
)
```

```r
replace_single_html(
  file,
  footer_text = NULL,
  remove_resources_start = TRUE,
  fullbleed = FALSE,
  verbose = TRUE
)
```
set_knitr_image_path

Arguments

- path: path to the markdown files that need replacement.
- remove_resources_start: remove the word 'resources/' at the front of any image path.
- footer_text: a bit of text that will be added to the end of each file before the references section.
- fullbleed: should the image have the attribute 'fullbleed: true'?
- verbose: print diagnostic messages
- file: individual markdown file

Value

A list of output files and diagnostics

---

**set_knitr_image_path**  
*Set image path for 'knitr'*

Description

Set image path for 'knitr'

Usage

```r
set_knitr_image_path(verbos = FALSE)
```

Arguments

- verbose: print out what the figure path is

Value

When used inside a knitted R Markdown document, will set the image path to a place compatible with 'ottrpal' output folders.
Description

Set up Manuscript folder for Leanpub publishing

Usage

```r
set_up_leanpub(
  path = ".",  
clean_up = FALSE,  
render = TRUE,  
output_dir = "manuscript",  
make_book_txt = FALSE,  
quiz_dir = "quizzes",  
run_quiz_checks = FALSE,  
remove_resources_start = FALSE,  
verbose = TRUE,  
footer_text = NULL,  
embed = NULL
)
```

Arguments

- **path**: path to the bookdown book, must have a ‘_bookdown.yml’ file
- **clean_up**: TRUE/FALSE the old output directory should be deleted and everything created fresh.
- **render**: if ‘TRUE’, then [bookdown::render_book()] will be run on each Rmd.
- **output_dir**: output directory to put files. It should likely be relative to path
- **make_book_txt**: Should [ottrpal::bookdown_to_book_txt()] be run to create a ‘Book.txt’ in the output directory?
- **quiz_dir**: directory that contains the quiz .md files that should be checked and incorporated into the Book.txt file. If you don’t have quizzes, set this to NULL
- **run_quiz_checks**: TRUE/FALSE run quiz checks
- **remove_resources_start**: remove the word ‘resources/’ at the front of any image path.
- **verbose**: print diagnostic messages
- **footer_text**: Optionally can add a bit of text that will be added to the end of each file before the references section.
- **embed**: is this being run by bookdown_to_embed_leanpub? TRUE/FALSE

Value

A list of output files and diagnostics
simple_references \hspace{1cm} Run Simple References

Description
Run Simple References

Usage
\begin{verbatim}
simple_references(x, bib_files, add_reference_header = FALSE)
\end{verbatim}

Arguments
\begin{itemize}
\item \textbf{x} \hspace{1cm} A filename of a markdown or Rmarkdown file, or the full output from \texttt{readLines} on that file
\item \textbf{bib_files} \hspace{1cm} bibliography files to use for pandoc
\item \textbf{add_reference_header} \hspace{1cm} Should the \texttt{## References} header be added at the end of the output?
\end{itemize}

Value
A character vector of the file, with references subbed in

xml_notes \hspace{1cm} Get Notes from XML

Description
Get Notes from XML

Usage
\begin{verbatim}
xml_notes(file, collapse_text = TRUE, xpath = "/a:r//a:t")
\end{verbatim}

Arguments
\begin{itemize}
\item \textbf{file} \hspace{1cm} XML file from a PPTX
\item \textbf{collapse_text} \hspace{1cm} should text be collapsed by spaces?
\item \textbf{xpath} \hspace{1cm} xpath to pass to \texttt{xml2::xml_find_all()}
\end{itemize}

Value
A character vector
Index

auth_from_secret, 3
authorize, 3

bad_quiz_path, 4
bookdown_destination, 5
bookdown_file, 5
bookdown_path, 6
bookdown_rmd_files, 6
bookdown_to_book_txt, 7
bookdown_to_embed_leanpub, 7
bookdown_to_leanpub, 9

check_all_questions, 10
check_question, 11
check_quiz, 12
check_quiz_attributes, 14
check_quizzes, 13
convert_coursera_quizzes, 15
convert_footnotes, 16
convert_quiz, 16
convert_utube_link, 17

crypt_creds_path, 17
crypt_creds_user_path, 18
eexample_repo_cleanup, 18
eexample_repo_setup, 19
extract_meta, 19
extract_object_id, 20
extract_quiz(parse_quiz), 28

get_bookdown_spec, 21
get_chapters, 22
get_gs_pptx, 22
get_image_from_slide
(gs_id_from_slide), 24
get_image_link_from_slide
(gs_id_from_slide), 24
get_object_id_notes, 23
get_slide_id, 22, 23

get_slide_page(gs_png_url), 25
good_quiz_path, 24
gs_id_from_slide, 24
gs_png_download(gs_png_url), 25
gs_png_url, 25

include_slide(gs_png_url), 25

key_encrypt_creds_path, 26

leanpub_check, 26

make_embed_markdown, 27

oauth2.0_token, 3

parse_q_tag, 29
parse_quiz, 28
parse_quiz_df, 29
pptx_notes, 30
pptx_slide_note_df(pptx_notes), 30
pptx_slide_text_df(pptx_notes), 30

remove_yaml_header, 31
render_without_toc, 31
replace_html, 32
replace_single_html(replace_html), 32

set_knitr_image_path, 33

set_up_leanpub, 34

simple_references, 35

unzip_pptx(pptx_notes), 30

xml_notes, 30, 35