Package ‘ottrpal’

May 18, 2023

Version 1.1.1
Title Companion Tools for Open-Source Tools for Training Resources (OTTR)
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.
Depends R (>= 3.5.0)
License GPL-3
Imports dplyr, httr, fs, rmarkdown (>= 2.10), xml2, rvest, readr, R.utils, bookdown, glue, curl, rprojroot, magrittr, yaml, stringr, knitr (>= 1.33), purrr
Suggests remotes, testthat, tibble, utils
Encoding UTF-8
ByteCompile true
Type Package
VignetteBuilder knitr

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

BugReports https://github.com/jhudsl/ottrpal/issues
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]
Maintainer Candace Savonen <cansav09@gmail.com>
Repository CRAN
Date/Publication 2023-05-18 16:00:02 UTC
### R topics documented:

- bad_quiz_path ................................................................. 3
- bookdown_destination .................................................... 3
- bookdown_file ............................................................... 4
- bookdown_path .............................................................. 4
- bookdown_rmd_files ....................................................... 5
- bookdown_to_book_txt .................................................... 5
- bookdown_to_embed_leanpub ............................................. 6
- bookdown_to_leanpub ..................................................... 7
- check_all_questions ...................................................... 8
- check_question ............................................................ 9
- check_quiz ................................................................. 10
- check_quizzes .............................................................. 11
- check_quiz_attributes ................................................... 12
- check_quiz_question_attributes ........................................ 13
- convert_coursera_quizzes ............................................... 13
- convert_footnotes ........................................................ 14
- convert_quiz ............................................................... 15
- convert_utube_link ....................................................... 15
- download_gs_file ......................................................... 16
- example_repo_cleanup ................................................... 17
- example_repo_setup ...................................................... 17
- extract_meta ............................................................... 18
- get_bookdown_spec ....................................................... 19
- get_chapters ............................................................... 19
- get_slide_id ............................................................... 20
- good_quiz_path ............................................................ 20
- gs_id_from_slide .......................................................... 21
- gs_png_url ................................................................. 21
- leanpub_check ............................................................. 22
- make_embed_markdown .................................................... 23
- parse_quiz ................................................................. 24
- parse_quiz_df .............................................................. 25
- parse_q_tag ............................................................... 25
- remove_yaml_header ..................................................... 26
- render_without_toc ....................................................... 27
- replace_html ............................................................... 28
- set_knitr_image_path .................................................... 29
- set_up_leanpub ............................................................ 29
- simple_references ....................................................... 30

**Index**

31
bad_quiz_path

| 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

| 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

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

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.

---

**bookdown_to_book_txt**  
*Create Book.txt file from files existing in quiz directory*

**Description**
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.

---

**bookdown_to_embed_leanpub**

*Convert Bookdown to Embed version of Leanpub*

---

**Description**

Convert Bookdown to Embed version of Leanpub

**Usage**

```r
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
- **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.
- **default_img**
  - A google slide link to the default image to be used for all chapters
- **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
## Not run:
```r
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)
```
check_all_questions

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>path to the bookdown book, must have a ‘_bookdown.yml’ file</td>
</tr>
<tr>
<td>render</td>
<td>if ‘TRUE’, then [bookdown::render_book()] will be run on each Rmd.</td>
</tr>
<tr>
<td>output_dir</td>
<td>output directory to put files. It should likely be relative to path</td>
</tr>
<tr>
<td>make_book_txt</td>
<td>Should [ottrpal::bookdown_to_book_txt()] be run to create a ‘Book.txt’ in the output directory?</td>
</tr>
<tr>
<td>quiz_dir</td>
<td>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</td>
</tr>
<tr>
<td>run_quiz_checks</td>
<td>TRUE/FALSE run quiz checks</td>
</tr>
<tr>
<td>remove_resources_start</td>
<td>remove the word ‘resources/’ at the front of any image path.</td>
</tr>
<tr>
<td>verbose</td>
<td>print diagnostic messages</td>
</tr>
<tr>
<td>footer_text</td>
<td>Optionally can add a bit of text that will be added to the end of each file before the references section.</td>
</tr>
<tr>
<td>clean_up</td>
<td>TRUE/FALSE the old output directory should be deleted and everything created fresh.</td>
</tr>
</tbody>
</table>

Value

A list of output files and diagnostics

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

```r
check_all_questions(
  quiz_specs, 
  quiz_name = NA, 
  verbose = TRUE, 
  ignore_coursera = TRUE
)
```
**check_question**

**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.

**Examples**

```r
# 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)
```

**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

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 compat-
ibility

Value

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

Examples

## Not run:

```r
# 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.
check_quizzes

Usage

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.

Examples

```r
## 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)
```

Description

Check the formatting of all quizzes in a given directory.

Usage

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

quiz_dir  A path to a directory full of quizzes that should all be checked with [otrrpal::check_all_quizzes].
write_report  TRUE/FALSE save warning report to a CSV file?
verbose  print diagnostic messages
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 checks performed on each quiz

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**

```r
check_quiz_question_attributes(question_df, quiz_name = NULL, verbose = TRUE)
```

**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.

---

**convert_coursera_quizzes**

*Convert Leanpub md quiz to Coursera yaml quiz*

**Description**

Convert Leanpub md quiz to Coursera yaml quiz

**Usage**

```r
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

```r
# 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(),
  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 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**
```
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**
```
## 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**
```
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.

download_gs_file      Download Google Slides File

Description

Download Google Slides File

Usage

download_gs_file(id, out_type = "pptx")

Arguments

id: Identifier of Google slides presentation, passed to get_slide_id
out_type: output type of file to download. Usually ‘pdf’ or ‘pptx’

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.
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
example_files <- ottrpal::example_repo_setup()

# Run this to delete them
example_repo_cleanup(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())
```
extract_meta

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
debug_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

### 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)
# 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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Where to look for the _bookdown.yml file. Passes to the bookdown_file() function. By default looks in current directory</td>
</tr>
</tbody>
</table>

## 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

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

## Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bookdown_index</td>
<td>The file path of the rendered bookdown index.html file</td>
</tr>
<tr>
<td>base_url</td>
<td>The base url of where the chapters are published – the url to provide to the iframe in Leanpub e.g. <a href="https://jhudatascience.org/OTTR_Template/coursera">https://jhudatascience.org/OTTR_Template/coursera</a></td>
</tr>
</tbody>
</table>
Value

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

good_quiz_path

get_slide_id

Description

Get Slide ID from URL

Usage

get_slide_id(x)

Arguments

x  URL of slide

Value

A character vector

Examples

x <- paste0(  "https://docs.google.com/presentation/d/",  "1Tg-GTGnUPduO1tZYuMoelqUNZnUp3vvg_7TtpUPL7e8",  "/edit#slide=id.g154aa4fae2_0_58"
)  
get_slide_id(x)

good_quiz_path

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

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

---

### `gs_id_from_slide`

#### 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

#### Value

A scalar character vector

---

### `gs_png_url`

#### 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"),
)
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 \texttt{knitr::include_graphics()}

Value

A character vector of URLs

Examples

url <- paste0(
  "https://docs.google.com/presentation/d/",
  "12PZgPteQBwgal6KSPPP58zhPhjZ7QSPZLe3NkA8M3eo/edit",
  "#slide=id.gc8648f14c3_0_397&t=4"
)
id <- get_slide_id(url)
gs_png_url(url)
**Description**

Make Leanpub file that has embed webpage of a chapter

**Usage**

```r
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
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 of 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)
remove_yaml_header

Arguments

tag A single tag to extract from

Value

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

Examples

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

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**

```r
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
)
```

**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
)

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

### 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
set_knitr_image_path(verbos = FALSE)

Arguments
verbos 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.

set_up_leanpub  Set up Manuscript folder for Leanpub publishing

Description
Set up Manuscript folder for Leanpub publishing

Usage
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
)
simple_references

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  Run Simple References

Description

Run Simple References

Usage

`simple_references(x, bib_files, add_reference_header = FALSE)`

Arguments

- **x**
  - A filename of a markdown or Rmarkdown file, or the full output from ‘readLines’ on that file
- **bib_files**
  - Bibliography files to use for pandoc
- **add_reference_header**
  - Should the ‘## References’ header be added at the end of the output?

Value

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

bad_quiz_path, 3
bookdown_destination, 3
bookdown_file, 4
bookdown_path, 4
bookdown_rmd_files, 5
bookdown_to_book_txt, 5
bookdown_to_embed_leanpub, 6
bookdown_to_leanpub, 7
check_all_questions, 8
check_question, 9
check_quiz, 10
check_quiz_attributes, 12
check_quiz_question_attributes, 13
check_quizzes, 11
convert_coursera_quizzes, 13
convert_footnotes, 14
convert_quiz, 15
convert_utube_link, 15
download_gs_file, 16
download_gs_file, 16
example_repo_cleanup, 17
example_repo_setup, 17
extract_meta, 18
extract_quiz (parse_quiz), 24
get_bookdown_spec, 19
get_chapters, 19
get_image_from_slide
   (gs_id_from_slide), 21
get_image_link_from_slide
   (gs_id_from_slide), 21
get_slide_id, 16, 20
good_quiz_path, 20
gs_id_from_slide, 21
gs_png_download (gs_png_url), 21
gs_png_url, 21
include_slide (gs_png_url), 21

leanpub_check, 22
make_embed_markdown, 23
parse_q_tag, 25
parse_quiz, 24
parse_quiz_df, 25
remove_yaml_header, 26
render_without_toc, 27
replace_html, 28
replace_single_html (replace_html), 28
set_knitr_image_path, 29
set_up_leanpub, 29
simple_references, 30