Package 'vvcanvas'

May 8, 2026

Title 'Canvas' LMS API Integration

Version 0.0.8

Description
Allow R users to interact with the 'Canvas' Learning Management System (LMS) API (see <https://canvas.instructure.com/doc/api/all_resources.html> for details).
It provides a set of functions to access and manipulate course data, assignments, grades, users, and other resources available through the 'Canvas' API.

URL https://github.com/vusaverse/vvcanvas,
https://vusaverse.github.io/vvcanvas/

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, htm2txt, httr, jsonlite, magrittr, mime, purrr, rlang, stringr, tidyr, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Hajo Bons [aut, cre],
Tomer Iwan [aut],
Niels Smits [ctb] (ORCID: <https://orcid.org/0000-0003-3669-9266>),
VU Analytics [cph]

Maintainer Hajo Bons <h.b.bons@vu.nl>

Repository CRAN

Date/Publication 2026-01-19 13:20:02 UTC

Contents

`append_access_token` *Appends access_token to URL if not present*

Description

Appends access_token to URL if not present

Usage

```
append_access_token(url, access_token)
```

Arguments

`url`	The URL to which the access_token should be appended.
`access_token`	The Canvas API access token (string).

Value

The URL with the access_token appended if it was not already present.

---

canvas_api_key *Get the Canvas API key from the environment variable*

Description

Get the Canvas API key from the environment variable

Usage

```
canvas_api_key()
```

Value

The Canvas API key stored in the CANVAS_API_KEY environment variable.

---

canvas_authenticate *Authenticate with Canvas LMS API*

Description

This function handles authentication with the Canvas LMS API. It uses the provided API key and base URL, or falls back to the `CANVAS_API_KEY` and `CANVAS_BASE_URL` environment variables if none are provided.

Usage

```
canvas_authenticate(api_key = canvas_api_key(), base_url = canvas_base_url())
```

Arguments

`api_key`	The API key for authenticating with the Canvas LMS API. Defaults to the `CANVAS_API_KEY` environment variable.
`base_url`	The base URL of the Canvas instance. Defaults to the `CANVAS_BASE_URL` environment variable.

Value

A list containing the authenticated 'api_key' and 'base_url'.

Note

The function verifies authentication by making a test request to the `/api/v1/users/self` endpoint of the Canvas instance. If the response status code is not 200, it throws an error message indicating that authentication failed.

Examples

```
## Not run:
# Authenticate with the Canvas LMS API
api_key <- "your_api_key"
base_url <- "https://canvas.example.com"
canvas <- canvas_authenticate(api_key, base_url)

## End(Not run)
```

`canvas_base_url` *Get the Canvas base URL from the environment variable*

Description

Get the Canvas base URL from the environment variable

Usage

```
canvas_base_url()
```

Value

The Canvas base URL stored in the CANVAS_BASE_URL environment variable.

`create_assignment_group` *Create an Assignment Group in Canvas LMS*

Description

Creates a new assignment group in a specific course using the Canvas LMS API.

Usage

```
create_assignment_group(canvas, course_id, group_name, group_position = NULL)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course in which to create the assignment group.
`group_name`	The name of the assignment group.
`group_position`	The position of the assignment group in the course (optional).

Value

A confirmation message that the assignment group has been created.

`create_conversation` *Creates a new conversation with one or more recipients.*

Description

Canvas allows for an e-mail like communication with students, called 'conversations', which can be handled from the API.

Usage

```
create_conversation(
  canvas,
  recipients = NULL,
  subject = " ",
  body = " ",
  force_new = TRUE,
  group_conversation = FALSE,
  attachments = NULL,
  attachment_ids = NULL,
  media_comment_id = NULL,
  media_comment_type = NULL,
  mode = "sync",
  scope = NULL,
  filter = NULL,
  filter_mode = NULL,
  context_code = NULL
)
```

Arguments

`canvas`	An object containing the Canvas API key and base URL,
`recipients`	An array of recipient ids.
`subject`	The subject of the conversation. Maximum length is 255 characters.
`body`	The message to be sent. Unfortunately, Canvas only allows for plain text.
`force_new`	logical. Forces a new message to be created, even if there is an existing private conversation.
`group_conversation`	logical. Defaults to `FALSE`. When false, individual private conversations will be created with each recipient. If true, this will be a group conversation (i.e. all recipients may see all messages and replies). Must be set true if the number of recipients is over the set maximum (default is 100).
`attachments`	An array of paths to local files that should be uploaded.
`attachment_ids`	An array of attachments ids. These must be files that have been previously uploaded to the sender's 'conversation attachments' folder.

media_comment_id
: Media comment id of an audio or video file to be associated with this message.

media_comment_type
: Type of the associated media file, values allowed are audio or video.

mode
: Determines whether the messages will be created/sent synchronously or asynchronously. Defaults to sync.

scope
: Used when generating 'visible' in the API response, values allowed are unread, starred, archived.

filter
: Used when generating 'visible' in the API response.

filter_mode
: Used when generating 'visible' in the API response, values allowed are and, or (default).

context_code
: The course or group that is the context for this conversation. Same format as courses or groups in the recipients argument.

Details

Note that the user should have rights to access the folder.

Value

A confirmation message indicating that the message has been successfully send.

See Also

get_single_conversation() and get_conversations().

---

create_course_datalake

Create a data lake for a course.

Description

This function retrieves data from various endpoints for a specific course in the Canvas LMS API and stores the data as JSON files in a specified storage location.

Usage

```
create_course_datalake(canvas, course_id, storage_location)
```

Arguments

canvas
: An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.

course_id
: The ID of the course for which to create the data lake.

storage_location
: The path to the storage location where the data files will be saved.

Value

NULL.

Note

This function retrieves data from various endpoints. Access to certain endpoints may require specific roles.

`create_course_section` *Create a Course Section in Canvas LMS*

Description

Creates a new course section in a specific course using the Canvas LMS API.

Usage

```
create_course_section(
  canvas,
  course_id,
  section_name,
  section_start_date = NULL,
  section_end_date = NULL
)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course in which to create the section.
`section_name`	The name of the section.
`section_start_date`	(Optional) The start date of the section.
`section_end_date`	(Optional) The end date of the section.

Value

A confirmation message that the section has been created.

create_folder *Create a Folder in Canvas LMS*

Description

Creates a new folder in a specific course using the Canvas LMS API.

Usage

```
create_folder(canvas, course_id, folder_name, parent_folder_id = NULL)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the folder.
folder_name	The name of the folder.
parent_folder_id	(Optional) The ID of the parent folder in which to create the folder.

Value

A confirmation message that the folder has been created.

create_group_category *Create a Group Category in Canvas LMS*

Description

Creates a new group category in a specific course using the Canvas LMS API.

Usage

```
create_group_category(
  canvas,
  course_id,
  category_name,
  allow_self_signup = FALSE
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the group category.
category_name	The name of the group category.
allow_self_signup	(Optional) Whether to allow self-signup for groups in the category. Defaults to FALSE.

Value

A confirmation message that the group category has been created.

`create_module` *Create a Module in Canvas LMS*

Description

Creates a new module in a specific course using the Canvas LMS API.

Usage

```
create_module(canvas, course_id, module_name, position = NULL)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course in which to create the module.
`module_name`	The title of the module.
`position`	(Optional) The position of this module in the course

Value

A confirmation message that the module has been created.

`create_module_item` *Create a Module Item in Canvas LMS*

Description

Creates a new item in a specific module in a specifc course using the Canvas LMS API.

Usage

```
create_module_item(
  canvas,
  course_id,
  module_id,
  item_title,
  item_type = "Page",
  position = NULL,
  page_url = NULL,
  page_id = NULL
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the module item.
module_id	The ID of the module in which to create the item.
item_title	The title of the new item within the module.
item_type	The type of item. Defaults to "Page".
position	(Optional) The position of the item within the module. Defaults to 1.
page_url	(Optional) The url of the page.
page_id	(Optional) The id of the page.

Value

A confirmation message that the page has been created.

---

create_page *Create a Page in Canvas LMS*

Description

Creates a new page in a specific course using the Canvas LMS API.

Usage

```
create_page(canvas, course_id, page_title, page_body, published = FALSE)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the page.
page_title	The title of the page.
page_body	The body/content of the page.
published	(Optional) Whether the page should be published. Defaults to FALSE.

Value

A confirmation message that the page has been created.

delete_course_section *Delete a Course Section in Canvas LMS*

Description

Deletes an existing course section using the Canvas LMS API.

Usage

```
delete_course_section(canvas, section_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
section_id	The ID of the section to delete.

Value

A confirmation message that the section has been deleted.

delete_page *Delete a Page in Canvas LMS*

Description

Deletes a page in a specific course using the Canvas LMS API.

Usage

```
delete_page(canvas, course_id, page_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to delete the page.
page_id	The ID of the page.

Value

A confirmation message that the page has been deleted.

download_course_file *Downloads a file from a given URL.*

Description

This function downloads a file from a specified URL and saves it locally.

Usage

```
download_course_file(canvas, file_url, download_path)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
file_url	The URL of the file to download.
download_path	The path where the file should be downloaded.

Value

The path of the downloaded file.

Examples

```
## Not run:
# Download a file from a given URL
canvas <- canvas_authenticate(api_key, base_url)
file_url <- "https://example.com/file.pdf"
download_path <- "path/to/save/file.pdf"
file_path <- download_course_file(canvas, file_url, download_path)

## End(Not run)
```

edit_section *Edit a Course Section in Canvas LMS*

Description

Modifies an existing course section using the Canvas LMS API.

Usage

```
edit_section(
  canvas,
  section_id,
  section_name = NULL,
  sis_section_id = NULL,
  integration_id = NULL,
  section_start_date = NULL,
  section_end_date = NULL,
  restrict_enrollments_to_section_dates = NULL,
  override_sis_stickiness = NULL
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
section_id	The ID of the section to edit.
section_name	The new name of the section (optional).
sis_section_id	The new SIS ID of the section (optional).
integration_id	The new integration ID of the section (optional).
section_start_date	The new start date of the section (optional).
section_end_date	The new end date of the section (optional).
restrict_enrollments_to_section_dates	Whether to restrict user enrollments to the start and end dates of the section (optional).
override_sis_stickiness	Whether to override SIS stickiness (optional).

Value

A confirmation message that the section has been edited.

---

extract_next_url *Extracts the 'next' URL from a Link header*

Description

Extracts the 'next' URL from a Link header

Usage

```
extract_next_url(link_header)
```

Arguments

link_header	The Link header string from an httr response.

Value

The URL (character) for the next page, or NULL if not present.

---

get_accounts *Get a list of accounts from the Canvas LMS API*

Description

Retrieves a paginated list of accounts that the current user can view or manage.

Usage

```
get_accounts(canvas, include = NULL, per_page = 100)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
include	A vector of additional information to include. Default is NULL.
per_page	Number of accounts to retrieve per page. Default is 100.

Value

A list of accounts retrieved from the Canvas LMS API.

---

get_all_courses *Retrieves a paginated list of all courses visible in the public index.*

Description

This function retrieves a paginated list of all courses visible in the public index using the Canvas LMS API. *NOTE* This function might take a while to finish.

Usage

```
get_all_courses(canvas, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
per_page	(Optional) The number of courses to retrieve per page of results (default is 100).

Value

A data frame of courses visible in the public index.

`get_assignments` *Get Assignments from Canvas LMS API*

Description

Fetches a list of assignments within a specific course from the Canvas LMS API.

Usage

```
get_assignments(canvas, course_id)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course for which to retrieve the assignments.

Value

A list of assignments retrieved from the Canvas LMS API.

`get_assignment_data` *Get course-level assignment data from the Canvas LMS API*

Description

Retrieves the course-level assignment data for a specific course from the Canvas LMS API.

Usage

```
get_assignment_data(canvas, course_id, per_page = 100)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course for which to retrieve the assignment data.
`per_page`	Number of assignment data to retrieve per page. Default is 100.

Value

A list of assignment data retrieved from the Canvas LMS API.

get_assignment_details *Get Assignment Details from Canvas LMS API*

Description

Retrieves detailed information about a specific assignment from the Canvas LMS API.

Usage

```
get_assignment_details(canvas, course_id, assignment_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which the assignment resides.
assignment_id	The ID of the assignment for which to retrieve the details.

Value

A dataframe containing the detailed information about the assignment.

get_assignment_groups *Retrieves the assignment groups within a course.*

Description

This function retrieves the assignment groups within a specific course in the Canvas LMS API.

Usage

```
get_assignment_groups(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the `canvas_authenticate` function.
course_id	The ID of the course for which to fetch the assignment groups.
per_page	The number of entries to show per page.

Value

A list of assignment groups within the specified course.

get_assignment_submissions *Retrieves assignment submissions.*

Description

This function retrieves the submissions to a specific assignment in a specific course in the Canvas LMS API.

Usage

```
get_assignment_submissions(canvas, course_id, assignment_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch submissions.
assignment_id	The ID of the assignment for which to fetch the submissions.
per_page	(Optional) The number of submissions to retrieve per page of results (default is 100).

Value

A data frame containing the submission data.

get_calendar_events *Get Calendar Events from Canvas LMS API*

Description

Retrieve the paginated list of calendar events or assignments for the current user.

Usage

```
get_calendar_events(
  canvas,
  type = "event",
  start_date = NULL,
  end_date = NULL,
  undated = FALSE,
  all_events = FALSE,
  context_codes = NULL,
  excludes = NULL,
  includes = NULL,
```

```
  important_dates = FALSE,
  blackout_date = FALSE
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
type	The type of events to retrieve. Default is "event". Allowed values: event, assignment.
start_date	Only return events since the start_date (inclusive). Defaults to today.
end_date	Only return events before the end_date (inclusive). Defaults to start_date.
undated	Defaults to false (dated events only). If true, only return undated events and ignore start_date and end_date.
all_events	Defaults to false (uses start_date, end_date, and undated criteria). If true, all events are returned, ignoring start_date, end_date, and undated criteria.
context_codes	List of context codes of courses, groups, users, or accounts whose events you want to see. If not specified, defaults to the current user (i.e personal calendar, no course/group events).
excludes	Array of attributes to exclude. Possible values are "description", "child_events" and "assignment".
includes	Array of optional attributes to include. Possible values are "web_conference" and if calendar_series flag is on, "series_natural_language".
important_dates	Defaults to false. If true, only events with important dates set to true will be returned.
blackout_date	Defaults to false. If true, only events with blackout date set to true will be returned.

Value

A data frame of calendar events retrieved from the Canvas LMS API.

---

get_conversations *Retrieves conversations.*

Description

This function retrieves all details concerning conversations of the user.

Usage

```
get_conversations(canvas)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.

Value

Returns a data frame containing the details of conversations for the current user, most recent ones first.

See Also

get_single_conversation() and create_conversation().

---

get_courses *Get Courses from Canvas LMS API*

Description

Retrieves a list of courses from the Canvas LMS API.

Usage

```
get_courses(canvas, per_page = 100)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
per_page	Number of courses to retrieve per page. Default is 100.

Value

A list of courses retrieved from the Canvas LMS API.

---

get_course_announcements *Retrieves course announcements.*

Description

This function retrieves a list of announcements for a specific course in the Canvas LMS API.

Usage

```
get_course_announcements(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to retrieve announcements.
per_page	(Optional) The number of announcements to retrieve per page of results (default is 100).

Value

A data frame of course announcements.

---

get_course_details *Get Course Details from Canvas LMS API*

Description

Retrieves detailed information about a specific course from the Canvas LMS API.

Usage

```
get_course_details(canvas, course_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the details.

Value

A dataframe containing the detailed information about the course.

---

get_course_enrollments *Retrieves the course enrollments for a course.*

Description

This function retrieves the enrollments of students and other roles in a specific course in the Canvas LMS API.

Usage

```
get_course_enrollments(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the enrollments.
per_page	(Optional) The number of enrollments to retrieve per page of results (default is 100).

Value

A data frame of course enrollments for the specified course.

---

get_course_files *Retrieves a list of files within a course.*

Description

This function retrieves a list of files within a specific course in the Canvas LMS API.

Usage

```
get_course_files(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the files.
per_page	(Optional) The number of files to retrieve per page of results (default is 100).

Value

A data frame of files within the specified course.

get_course_folders *Retrieves course folders.*

Description

This function retrieves a list of folders for a specific course in the Canvas LMS API.

Usage

```
get_course_folders(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to retrieve folders.
per_page	Number of courses to retrieve per page. Default is 100.

Details

Returns a list of all folders and sub folders for the given course. Note that for some reported sub folders its main folder may be outside of the course.

Value

A data frame of course folders.

See Also

get_course_root_folder() and get_user_folders()

get_course_gradebook *Constructs the gradebook of a course.*

Description

This function generates a gradebook for the assignments in a specific course in the Canvas LMS API.

Usage

```
get_course_gradebook(canvas, course_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to generate the gradebook.

Details

The gradebook shown in Canvas is not accessible through the API. Here it is reconstructed using the (visible) assignments and students in the course. It may be useful for performing more advanced grade calculations (like allowing for conditional extra credit) before adjusting assignment grades (see, update_course_grades and update_section_grades).

Value

A data frame containing the gradebook with student in rows (identifiable through canvas_user_id) and assignments in columns (identifiable through assigment names).

See Also

get_assignments(), and get_assignment_submissions().

---

get_course_groups *Retrieves the list of groups in a course.*

Description

This function retrieves the list of groups in a specific course in the Canvas LMS API.

Usage

```
get_course_groups(canvas, course_id, per_page = 100)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to fetch the groups.
per_page	(Optional) The number of groups to retrieve per page of results (default is 100).

Value

A data frame of groups in the specified course.

get_course_media_objects

Retrieves the media objects in a course.

Description

This function retrieves the media objects associated with a specific course in the Canvas LMS API.

Usage

```
get_course_media_objects(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the media objects.
per_page	The number of entries to show per page.

Value

A data frame containing the media objects in the specified course.

get_course_pages

Retrieves the pages within a course.

Description

This function retrieves the pages within a specific course in the Canvas LMS API.

Usage

```
get_course_pages(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the pages.
per_page	(Optional) The number of pages to retrieve per page of results (default is 50).

Value

A list of pages within the specified course.

get_course_participation

Get course-level participation data from Canvas LMS API

Description

Fetches the participation data for a specific course from the Canvas LMS API.

Usage

```
get_course_participation(canvas, course_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the participation data.

Value

The participation data for the specified course retrieved from the Canvas LMS API.

get_course_quizzes

Retrieves course quizzes.

Description

This function retrieves a list of quizzes for a specific course in the Canvas LMS API.

Usage

```
get_course_quizzes(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the `canvas_authenticate` function.
course_id	The ID of the course for which to retrieve quizzes.
per_page	(Optional) The number of quizzes to retrieve per page of results (default is 100).

Value

A data frame of course quizzes.

get_course_root_folder

Retrieves root folder of a course.

Description

This function retrieves the root folder associated with a course.

Usage

```
get_course_root_folder(canvas, course_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch associated folders.

Value

Returns a data frame containing the details on the root folder associated with the specified course.

See Also

get_course_folders() and get_user_folders()

get_course_sections

Retrieves course sections.

Description

This function retrieves a list of sections for a specific course in the Canvas LMS API.

Usage

```
get_course_sections(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to retrieve sections.
per_page	(Optional) The number of sections to retrieve per page of results (default is 100).

Value

A data frame of course sections.

get_course_students *Retrieves the list of students in a course.*

Description

This function retrieves the list of students enrolled in a specific course in the Canvas LMS API.

Usage

```
get_course_students(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the students.
per_page	(Optional) The number of students to retrieve per page of results (default is 100).

Value

A data frame of students enrolled in the specified course.

get_course_users *Retrieves the users in a course.*

Description

This function retrieves the users enrolled in a specific course in the Canvas LMS API.

Usage

```
get_course_users(
  canvas,
  course_id,
  per_page = 100,
  include = c("enrollments", "locked", "avatar_url", "test_student", "bio",
    "custom_links", "current_grading_period_scores", "uuid")
)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the users.
per_page	The number of entries to show per page.
include	Optional parameters to include in the response. Possible values: "enrollments", "locked", "avatar_url", "test_student", "bio", "custom_links", "current_grading_period_scores", "uuid".

Value

A data frame containing the users in the specified course.

---

get_department_grade_data

Get department-level grade data from the Canvas LMS API

Description

Retrieves the department-level grade data for a specific account and term from the Canvas LMS API.

Usage

```
get_department_grade_data(
  canvas,
  account_id,
  type = "current",
  term_id = NULL,
  per_page = 100
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
account_id	The ID of the account for which to retrieve the grade data.
type	The type of courses to include in the data. Can be 'current', 'completed', or 'term'.
term_id	The ID of the term for which to retrieve the grade data. Only used when type is 'terms/<term_id>'.
per_page	Number of grade data to retrieve per page. Default is 100.

Value

A data frame of grade data retrieved from the Canvas LMS API.

get_department_participation_data

Get department-level participation data from the Canvas LMS API

Description

Retrieves the department-level participation data for a specific account and term from the Canvas LMS API.

Usage

```
get_department_participation_data(
  canvas,
  account_id,
  type = "current",
  term_id = NULL,
  per_page = 100
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
account_id	The ID of the account for which to retrieve the participation data.
type	The type of courses to include in the data. Can be 'current', 'completed', or 'term'.
term_id	The ID of the term for which to retrieve the participation data. Only used when type is 'terms/<term_id>'.
per_page	Number of participation data to retrieve per page. Default is 100.

Value

A data frame of participation data retrieved from the Canvas LMS API.

get_department_statistics

Get department-level statistics from the Canvas LMS API

Description

Retrieves department-level statistics for a specific account and term from the Canvas LMS API.

Usage

```
get_department_statistics(canvas, account_id, type = "current", term_id = NULL)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
account_id	The ID of the account for which to retrieve the statistics.
type	The type of courses to include in the data. Can be 'current', 'completed', or 'term'.
term_id	The ID of the term for which to retrieve the statistics. Only used when type is 'terms/<term_id>'.

Value

A list of department-level statistics retrieved from the Canvas LMS API.

---

get_department_statistics_by_subaccount

Get department-level statistics by subaccount from the Canvas LMS API

Description

Retrieves department-level statistics for a specific account and term from the Canvas LMS API.

Usage

```
get_department_statistics_by_subaccount(
  canvas,
  account_id,
  type = "current",
  term_id = NULL
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
account_id	The ID of the account for which to retrieve the statistics.
type	The type of courses to include in the data. Can be 'current', 'completed', or 'term'.
term_id	The ID of the term for which to retrieve the statistics. Only used when type is 'terms/<term_id>'.

Value

A list of department-level statistics retrieved from the Canvas LMS API.

get_discussions *Retrieves the discussion topics within a course.*

Description

This function retrieves the discussion topics within a specific course in the Canvas LMS API.

Usage

```
get_discussions(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the discussion topics.
per_page	The number of entries to show

Value

A list of discussion topics within the specified course.

get_favorite_courses *Get Favorite Courses in Canvas LMS*

Description

Retrieves the data of favorite courses for the authenticated user using the Canvas LMS API.

Usage

```
get_favorite_courses(canvas, exclude_blueprint_courses = NULL)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
exclude_blueprint_courses	When set, only return courses that are not configured as blueprint courses (optional).

Value

The dataframe of favorite courses.

get_folder_files *Retrieves files in a specific folder.*

Description

This function lists all files available in a specific folder.

Usage

```
get_folder_files(canvas, folder_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
folder_id	The ID of the folder for which to fetch associated files.

Value

Returns a data frame containing the details on the files stored in the specified folder.

get_group_categories *Get group categories for a context*

Description

This function retrieves the group categories for a specific context (e.g., course) in the Canvas LMS API.

Usage

```
get_group_categories(canvas, course_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the group categories.

Value

A data frame of group categories in the specified context.

get_group_info *Get information about a single group*

Description

This function retrieves information about a specific group in the Canvas LMS API.

Usage

```
get_group_info(canvas, group_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
group_id	The ID of the group for which to retrieve the information.

Value

A list containing the information about the specified group.

get_group_memberships *Get group memberships*

Description

This function retrieves the memberships for a specific group in the Canvas LMS API.

Usage

```
get_group_memberships(canvas, group_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
group_id	The ID of the group for which to retrieve the memberships.

Value

A data frame of memberships in the specified group.

get_group_users *Get users in a group*

Description

This function retrieves the users in a specific group in the Canvas LMS API.

Usage

```
get_group_users(canvas, group_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
group_id	The ID of the group for which to retrieve the users.

Value

A data frame of users in the specified group.

get_modules *Retrieves the modules within a course.*

Description

This function retrieves the modules within a specific course in the Canvas LMS API.

Usage

```
get_modules(canvas, course_id, per_page = 100)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch the modules.
per_page	The number of entries to show

Value

A list of modules within the specified course.

get_module_items *Retrieves the items within a specific module.*

Description

This function retrieves the items within a specific module of a course in the Canvas LMS API.

Usage

```
get_module_items(canvas, course_id, module_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course containing the module.
module_id	The ID of the module for which to fetch the items.

Value

A list of items within the specified module.

get_page_content *Retrieves the content body of a specified page.*

Description

This function retrieves the content body of a specified page within a course in the Canvas LMS API.

Usage

```
get_page_content(canvas, course_id, page_id, return_as_plain_text = TRUE)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course to which the page belongs.
page_id	The ID of the page for which to fetch the content body.
return_as_plain_text	A logical value indicating whether to return the content as plain text (default is TRUE).

Value

The content body of the specified page, either as plain text or raw HTML.

get_quiz_submissions *Retrieves quiz submissions.*

Description

This function retrieves the submissions to a specific quiz in a specific course in the Canvas LMS API.

Usage

```
get_quiz_submissions(canvas, course_id, quiz_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to fetch submissions.
quiz_id	The ID of the quiz for which to fetch the submissions.

Value

A data frame containing the submission data.

See Also

get_assignment_submissions()

get_roles *Retrieve Roles for a Canvas Account*

Description

This function retrieves a paginated list of the roles available to a specific account in the Canvas LMS system.

Usage

```
get_roles(canvas, account_id, per_page = 100)
```

Arguments

canvas	A list containing the base URL and API key for the Canvas LMS instance.
account_id	The ID of the account for which you want to retrieve the roles.
per_page	The number of roles to retrieve per page (default is 100).

Value

A data frame containing the roles available to the specified account.

`get_section_information` *Get Section Information in Canvas LMS*

Description

Retrieves information about a specific course section using the Canvas LMS API.

Usage

```
get_section_information(canvas, course_id, section_id)
```

Arguments

`canvas`	A list containing the 'api_key' and 'base_url' for authentication.
`course_id`	The ID of the course.
`section_id`	The ID of the section.

Value

The information about the section.

`get_section_students` *Retrieve Students in a Section*

Description

This function retrieves a list of students enrolled in a specific section of a course in Canvas LMS.

Usage

```
get_section_students(canvas, section_id, per_page = 100)
```

Arguments

`canvas`	A list containing the Canvas API key and base URL, typically obtained through a `canvas_authenticate` function.
`section_id`	The ID of the section for which to retrieve students.
`per_page`	(Optional) The number of student records to retrieve per page of results. Defaults to 100.

Value

A data frame containing details of students enrolled in the specified section.

Examples

```
## Not run:
canvas <- list(base_url = "https://your_canvas_instance.instructure.com", api_key = "your_api_key")
section_id <- "67890"
students <- get_section_students(canvas, section_id)
print(students)

## End(Not run)
```

---

get_single_conversation

Retrieves a single conversation.

Description

This function retrieves all details concerning a specific conversation.

Usage

```
get_single_conversation(canvas, conversation_id)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
conversation_id	The id of the specific conversation.

Value

Returns information for a specific conversation for the current user. Response includes all fields that are present in the list/index action as well as messages and extended participant information.

See Also

get_conversations() and create_conversation().

get_student_summaries *Get student summaries for a course from Canvas LMS API*

Description

Retrieves the student summaries for a specific course from the Canvas LMS API.

Usage

```
get_student_summaries(canvas, course_id, per_page = 100)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the student summaries.
per_page	Number of student summaries to retrieve per page. Default is 100.

Value

A list of student summaries retrieved from the Canvas LMS API.

get_users *Get all users from an account*

Description

Retrieves users from a specific account in the Canvas LMS API with optional filtering and pagination.

Usage

```
get_users(
  canvas,
  account_id = "self",
  per_page = 100,
  search_term = NULL,
  enrollment_type = NULL,
  sort = NULL,
  order = NULL,
  include_deleted_users = NULL,
  uuids = NULL,
  include = NULL
)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
account_id	The ID of the account. Defaults to "self" for the current user's account.
per_page	The number of entries to show per page. Default is 100.
search_term	Optional. The partial name or full ID of the users to match and return in the results list. Must be at least 3 characters.
enrollment_type	Optional. When set, only return users enrolled with the specified course-level base role. Possible values: 'student', 'teacher', 'ta', 'observer', or 'designer'.
sort	Optional. The column to sort results by. Possible values: 'username', 'email', 'sis_id', 'integration_id', 'last_login', 'id'.
order	Optional. The order to sort the given column by. Possible values: 'asc', 'desc'.
include_deleted_users	Optional. When set to TRUE, returns users who have deleted pseudonyms for the context.
uuids	Optional. A vector of UUIDs. When set, only return users with the specified UUIDs. UUIDs after the first 100 are ignored.
include	Optional. A vector of additional information to include. Possible values include: "email", "enrollments", "avatar_url", "bio", "last_login", "time_zone", "locale", "uuid".

Value

A data frame containing the users matching the specified criteria.

Examples

```
## Not run:
# Authenticate with Canvas
canvas <- canvas_authenticate()

# Get all users from the default account
users <- get_users(canvas)

# Get users with a search term
users <- get_users(canvas, search_term = "John")

# Get users with email addresses included
users <- get_users(canvas, include = c("email", "enrollments"))

## End(Not run)
```

get_user_course_assignment_data

Get user-in-a-course-level assignment data from the Canvas LMS API

Description

Retrieves user-in-a-course-level assignment data for a specific course and student from the Canvas LMS API.

Usage

```
get_user_course_assignment_data(canvas, course_id, student_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the assignment data.
student_id	The ID of the student for which to retrieve the assignment data.

Value

A list of user-in-a-course-level assignment data retrieved from the Canvas LMS API.

get_user_course_messaging_data

Get user-in-a-course-level messaging data from the Canvas LMS API

Description

Retrieves user-in-a-course-level messaging data for a specific course and student from the Canvas LMS API.

Usage

```
get_user_course_messaging_data(canvas, course_id, student_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the messaging data.
student_id	The ID of the student for which to retrieve the messaging data.

Value

A list of user-in-a-course-level messaging data retrieved from the Canvas LMS API.

get_user_course_participation_data

Get user-in-a-course-level participation data from the Canvas LMS API

Description

Retrieves user-in-a-course-level participation data for a specific course and student from the Canvas LMS API.

Usage

```
get_user_course_participation_data(canvas, course_id, student_id)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course for which to retrieve the participation data.
student_id	The ID of the student for which to retrieve the participation data.

Value

A list of user-in-a-course-level participation data retrieved from the Canvas LMS API.

get_user_folders

Retrieves folders of the current user.

Description

This function retrieves all folders associated with the user accessing the API.

Usage

```
get_user_folders(canvas)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the `canvas_authenticate` function.

Details

Returns a list of all folders 'owned by' the current user. Note that since `self` replaces `user_id`, the id of the user accessing the API does not need to be specified. The highest level folder name is `my folder`.

Value

Returns a data frame containing the details on the folders available to the user accessing the API.

See Also

`get_course_root_folder()` and `get_course_folders()`

---

list_all_enrollment_terms

List All Enrollment Terms

Description

This function retrieves a paginated list of all enrollment terms for a specified account in Canvas LMS.

Usage

```
list_all_enrollment_terms(canvas, account_id, per_page = 100)
```

Arguments

canvas	A list containing Canvas API configuration: • base_url: The base URL of your Canvas instance • api_key: Your Canvas API key
account_id	The ID of the account for which to retrieve enrollment terms
per_page	The number of terms to retrieve per page (default is 100)

Value

A dataframe containing all enrollment terms details

---

paginate

Helper to paginate Canvas API GET requests

Description

Follows 'next' links in the Link header to retrieve all pages. Returns a list of all responses.

Usage

```
paginate(initial_response, access_token, showProgress = TRUE)
```

Arguments

initial_response	The initial httr response object from the first GET request.
access_token	The Canvas API access token (string).
showProgress	Logical; whether to print progress as pages are fetched. Default is TRUE.

Value

A list of httr response objects, one for each page.

---

post_new_discussion *Post a New Discussion in Canvas LMS*

Description

Creates a new discussion topic in a specific course using the Canvas LMS API.

Usage

```
post_new_discussion(
  canvas,
  course_id,
  discussion_title,
  discussion_message,
  discussion_is_announcement = FALSE
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the discussion.
discussion_title	The title of the discussion topic.
discussion_message	The initial message content of the discussion topic.
discussion_is_announcement	(Optional) Whether the discussion should be an announcement. Defaults to FALSE.

Value

A confirmation message that the discussion has been created.

query_progress *Queries progress.*

Description

This function queries the progress of asynchronous API operations.

Usage

```
query_progress(canvas, progress_url, attempts = 10)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
progress_url	The url where a progress update can be retrieved.
attempts	The number of times it is allowed to check progress.

Details

It performs 10 attempts with a pause of 6 seconds between them.

Value

a message indicating completion and status information about the asynchronous job.

update_course_grades *Updates assignment grades in a course.*

Description

This function updates the assignment grades of a specific assignment in a specific course in the Canvas LMS API.

Usage

```
update_course_grades(
  canvas,
  course_id,
  assignment_id,
  student_ids,
  posted_grades
)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course for which to generate the gradebook.
assignment_id	The ID of the assignment for which the grades are updated.
student_ids	The IDs of the students for whom the grades are updated.
posted_grades	The values of the (updated) grades.

Details

It updates the grading on multiple students' assignment submissions in an asynchronous job. The user must have permission to manage grades in the course. The API documentation describes the following types of scores:

points A floating point or integral value, such as "13.5". The grade will be interpreted directly as the score of the assignment. Values above assignment.points_possible are allowed, for awarding extra credit.

percentage A floating point value appended with a percent sign, such as "40%". The grade will be interpreted as a percentage score on the assignment, where 100% == assignment.points_possible. Values above 100% are allowed, for awarding extra credit.

letter grade A letter grade, following the assignment's defined letter grading scheme. For example, "A-". The resulting score will be the high end of the defined range for the letter grade. For instance, if "B" is defined as 86% to 84%, a letter grade of "B" will be worth 86%. The letter grade will be rejected if the assignment does not have a defined letter grading scheme. For more fine-grained control of scores, pass in points or percentage rather than the letter grade.

pass/complete/fail/incomplete A string value of "pass" or "complete" will give a score of 100%. "fail" or "incomplete" will give a score of 0.

Value

A confirmation message indicating that the grades have been updated.

See Also

update_section_grades()

---

update_page *Update a Page in Canvas LMS*

Description

Updates an existing page in a specific course using the Canvas LMS API.

Usage

```
update_page(canvas, course_id, page_id, page_params)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course in which to create the page.
page_id	The ID of the page.
page_params	A named list of page parameters to update. This list can include:
	title (string) The title for the updated page.
	body (string) The content for the updated page.
	published (boolean) Whether the page is published (TRUE) or draft state (FALSE).
	published_at (DateTime) Schedule a future date/time to publish the page.

Value

A confirmation message that the page has been updated.

---

update_quiz *Modify an existing quiz.*

Description

This function modifies an existing quiz in a specific course in the Canvas LMS API.

Usage

```
update_quiz(canvas, course_id, quiz_id, quiz_params)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course containing the quiz to be modified.
quiz_id	The ID of the quiz to be modified.
quiz_params	A named list of quiz parameters to update. This list can include:
	title (string) The quiz title.
	description (string) A description of the quiz.
	quiz_type (string) The type of quiz. Allowed values: "practice_quiz", "assignment", "graded_survey", "survey".
	assignment_group_id (integer) The assignment group id to put the assignment in. Defaults to the top assignment group in the course. Only valid if the quiz is graded.
	time_limit (integer) Time limit to take this quiz, in minutes. Set to NULL for no time limit. Defaults to NULL.
	shuffle_answers (boolean) If TRUE, quiz answers for multiple choice questions will be randomized for each student. Defaults to FALSE.

hide_results (string) Dictates whether or not quiz results are hidden from students. Allowed values: `"always"`, `"until_after_last_attempt"`. Defaults to `NULL`.

show_correct_answers (boolean) If `FALSE`, hides correct answers from students when quiz results are viewed. Defaults to `TRUE`.

show_correct_answers_last_attempt (boolean) If `TRUE`, hides correct answers from students until they submit the last attempt for the quiz. Defaults to `FALSE`.

show_correct_answers_at (DateTime) The correct answers will be visible by students only after this date.

hide_correct_answers_at (DateTime) The correct answers will stop being visible once this date has passed.

allowed_attempts (integer) Number of times a student is allowed to take a quiz. Set to `-1` for unlimited attempts. Defaults to `1`.

scoring_policy (string) Scoring policy for a quiz that students can take multiple times. Allowed values: `"keep_highest"`, `"keep_latest"`.

one_question_at_a_time (boolean) If `TRUE`, shows quiz to student one question at a time. Defaults to `FALSE`.

cant_go_back (boolean) If `TRUE`, questions are locked after answering. Defaults to `FALSE`.

access_code (string) Restricts access to the quiz with a password. For no access code restriction, set to `NULL`.

ip_filter (string) Restricts access to the quiz to computers in a specified IP range.

due_at (DateTime) The day/time the quiz is due. Accepts times in ISO 8601 format, e.g., `"2011-10-21T18:48Z"`.

lock_at (DateTime) The day/time the quiz is locked for students.

unlock_at (DateTime) The day/time the quiz is unlocked for students.

published (boolean) Whether the quiz should be published.

one_time_results (boolean) Whether students should be prevented from viewing their quiz results past the first time. Defaults to `FALSE`.

only_visible_to_overrides (boolean) Whether this quiz is only visible to overrides. Defaults to `FALSE`.

notify_of_update (boolean) If `TRUE`, notifies users that the quiz has changed. Defaults to `TRUE`.

Value

A list representing the updated quiz object.

`update_section_grades` *Updates assignment grades in a section.*

Description

This function updates the assignment grades of a specific assignment in a specific section in the Canvas LMS API.

Usage

```
update_section_grades(
  canvas,
  section_id,
  assignment_id,
  student_ids,
  posted_grades
)
```

Arguments

`canvas`	An object containing the Canvas API key and base URL, obtained through the `canvas_authenticate` function.
`section_id`	The ID of the section for which to generate the gradebook.
`assignment_id`	The ID of the assignment for which the grades are updated.
`student_ids`	The IDs of the students for whom the grades are updated.
`posted_grades`	The values of the (updated) grades.

Details

It updates the grading on multiple students' assignment submissions in an asynchronous job. The user must have permission to manage grades in the section. The API documentation describes the following types of scores:

points A floating point or integral value, such as "13.5". The grade will be interpreted directly as the score of the assignment. Values above assignment.points_possible are allowed, for awarding extra credit.

percentage A floating point value appended with a percent sign, such as "40%". The grade will be interpreted as a percentage score on the assignment, where 100% == assignment.points_possible. Values above 100% are allowed, for awarding extra credit.

letter grade A letter grade, following the assignment's defined letter grading scheme. For example, "A-". The resulting score will be the high end of the defined range for the letter grade. For instance, if "B" is defined as 86% to 84%, a letter grade of "B" will be worth 86%. The letter grade will be rejected if the assignment does not have a defined letter grading scheme. For more fine-grained control of scores, pass in points or percentage rather than the letter grade.

pass/complete/fail/incomplete A string value of "pass" or "complete" will give a score of 100%. "fail" or "incomplete" will give a score of 0.

Value

A confirmation message indicating that the grades have been updated.

See Also

update_course_grades()

---

upload_file *Upload a file in Canvas LMS.*

Description

This function uploads a file to the files folder of a specific course.

Usage

```
upload_file(canvas, course_id, file_name)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
course_id	The ID of the course where the file will be uploaded.
file_name	The file to be uploaded.

Value

A confirmation message indicating that the file has been uploaded.

---

upload_folder_file *Uploads a file to a specific folder.*

Description

This function uploads a specific file to a specific folder.

Usage

```
upload_folder_file(canvas, folder_id, file_name, generate_message = TRUE)
```

Arguments

canvas	An object containing the Canvas API key and base URL, obtained through the canvas_authenticate function.
folder_id	The ID of the folder to which the file is uploaded.
file_name	The path to the local file that should be uploaded.
generate_message	A logical expressing whether a message should be shown after a succesful upload; defaults to TRUE.

Details

Note that the user should have rights to access the folder.

Value

If generate_message is TRUE, a confirmation message indicating that the file has been successfully uploaded.

See Also

get_course_root_folder() and get_user_folders()

---

upload_qti_file_with_migration

Upload QTI File with Content Migration

Description

This function uploads a QTI file to a specified course in Canvas LMS and initiates a content migration. It handles the process of creating the migration, uploading the file, and checking the migration status. Additionally, it allows for obtaining the ID of the resulting quiz within Canvas so that quiz settings are easily updated. In that case, because processing the QTI file (i.e. turning it into a quiz) takes some time, the execution of further commands need to be suspended. If the processing of the uploaded file has not yet resulted in a new quiz ID the user is asked if further waiting is required.

Usage

```
upload_qti_file_with_migration(
  canvas,
  course_id,
  qti_name,
  wait = FALSE,
  settings = list()
)
```

Arguments

canvas	A list containing the 'api_key' and 'base_url' for authentication.
course_id	The ID of the course where the QTI file will be uploaded.
qti_name	The name of the QTI file (without the '.zip' extension) being uploaded.
wait	logical or integer. Defaults to FALSE, if TRUE the function suspends retrieving information by 30 seconds. Integer values represent waiting time in seconds.
settings	A list of additional settings for the content migration. This can include options like question bank name or overwrite settings.

Value

A quiz ID (equal to NULL if wait = FALSE or if no new quiz ID was found) and a confirmation message indicating that the content migration has been completed.

Index