

Package ‘clockify’

May 8, 2026

Type Package

Title A Wrapper for the 'Clockify' API

Version 0.1.7

Description A wrapper for the Clockify API <<https://docs.clockify.me/>>, making it possible to query, insert and update time keeping data.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

Imports anytime, dplyr, httr, janitor, logger, lubridate, methods, purrr, rlist, stringi, tibble, tidyr

Suggests here, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/testthat/start-first api-key, workspace

NeedsCompilation no

Author Andrew B. Collier [aut, cre]

Maintainer Andrew B. Collier <andrew.b.collier@gmail.com>

Repository CRAN

Date/Publication 2026-01-09 07:00:02 UTC

Contents

| | |
|-------------------------------|---|
| client | 3 |
| client-parameters | 4 |
| clients | 4 |
| client_create | 5 |
| client_delete | 5 |
| client_update | 6 |
| custom_fields | 6 |
| custom_field_delete | 7 |
| custom_field_update | 7 |
| get_api_key | 8 |

| | |
|--|----|
| paginate | 8 |
| project | 9 |
| project-update | 9 |
| project-update-estimate | 10 |
| projects | 11 |
| project_create | 12 |
| project_delete | 12 |
| project_update_billable_rate | 12 |
| project_update_cost_rate | 13 |
| project_update_memberships | 13 |
| reports-parameters | 14 |
| reports_detailed | 14 |
| reports_summary | 15 |
| reports_weekly | 16 |
| set_api_key | 16 |
| shared-reports-parameters | 17 |
| shared_report | 17 |
| shared_reports | 18 |
| shared_report_create | 18 |
| shared_report_delete | 19 |
| shared_report_update | 19 |
| tag | 20 |
| tags | 20 |
| tag_create | 21 |
| tag_delete | 21 |
| tag_update | 22 |
| task | 22 |
| task-create | 23 |
| tasks | 24 |
| task_delete | 24 |
| task_update | 25 |
| task_update_billable_rate | 26 |
| task_update_cost_rate | 26 |
| time-entry-parameters | 27 |
| time_entries | 27 |
| time_entry | 28 |
| time_entry_create | 29 |
| time_entry_delete | 30 |
| time_entry_invoiced | 31 |
| time_entry_set | 31 |
| time_entry_stop | 32 |
| user | 32 |
| user-update-role | 33 |
| users | 33 |
| user_create | 34 |
| user_delete | 35 |
| user_delete_role | 35 |
| user_groups | 36 |

| | |
|-----------------------------------|-----------|
| <i>client</i> | 3 |
| user_group_create | 36 |
| user_group_delete | 37 |
| user_group_update | 37 |
| user_group_user_add | 37 |
| user_group_user_remove | 38 |
| user_update_cost_rate | 38 |
| user_update_hourly_rate | 39 |
| user_update_status | 39 |
| workspace | 40 |
| workspaces | 40 |
| Index | 42 |

| | |
|---------------|-------------------|
| <i>client</i> | <i>Get client</i> |
|---------------|-------------------|

Description

Get client

Usage

```
client(client_id, concise = TRUE)
```

Arguments

| | |
|------------------------|-------------------------|
| <code>client_id</code> | Client ID |
| <code>concise</code> | Generate concise output |

Value

A data frame with one record per client

Examples

```
## Not run:
client("63a5493591ed63165538976d")

## End(Not run)
```

client-parameters *Parameters for client functions*

Description

Parameters for client functions

Arguments

| | |
|-----------|-------------------------|
| client_id | Client ID |
| concise | Generate concise output |

clients *Get clients*

Description

Get clients

Usage

```
clients(concise = TRUE)
```

Arguments

| | |
|---------|-------------------------|
| concise | Generate concise output |
|---------|-------------------------|

Value

A data frame with one record per client.

Examples

```
## Not run:  
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))  
  
clients()  
  
## End(Not run)
```

| | |
|---------------|--------------------------------------|
| client_create | <i>Add a new client to workspace</i> |
|---------------|--------------------------------------|

Description

Add a new client to workspace

Usage

```
client_create(name, concise = TRUE)
```

Arguments

| | |
|---------|-------------------------|
| name | Client name |
| concise | Generate concise output |

Value

A data frame with one row per record.

Examples

```
## Not run:  
client_create("RStudio")  
  
## End(Not run)
```

| | |
|---------------|---------------------------------------|
| client_delete | <i>Delete a client from workspace</i> |
|---------------|---------------------------------------|

Description

A client must first be archived before it can be deleted.

Usage

```
client_delete(client_id, archive = FALSE)
```

Arguments

| | |
|-----------|---------------------------------|
| client_id | Client ID |
| archive | Archive client before deleting. |

Value

A Boolean: TRUE on success or FALSE on failure.

| | |
|---------------|------------------------|
| client_update | <i>Update a client</i> |
|---------------|------------------------|

Description

Update a client

Usage

```
client_update(client_id, name = NULL, note = NULL, archived = NULL)
```

Arguments

| | |
|-----------|-----------------------------------|
| client_id | Client ID |
| name | Client name |
| note | Note about client |
| archived | Whether or not client is archived |

Value

A data frame with one record for the updated client.

| | |
|---------------|--------------------------|
| custom_fields | <i>Get custom fields</i> |
|---------------|--------------------------|

Description

Custom fields are only listed for specific projects if the default values for those fields have been modified for those projects.

Usage

```
custom_fields(project_id = NULL)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
|------------|------------|

Details

Custom fields are only available on the Pro and Enterprise plans.

custom_field_delete *Remove a custom field from a project*

Description

Remove a custom field from a project

Usage

```
custom_field_delete(project_id, custom_field_id)
```

Arguments

| | |
|-----------------|-----------------|
| project_id | Project ID |
| custom_field_id | Custom field ID |

custom_field_update *Update a custom field on a project*

Description

Update a custom field on a project

Usage

```
custom_field_update(  
  project_id,  
  custom_field_id,  
  default_value = NULL,  
  status = NULL  
)
```

Arguments

| | |
|-----------------|-------------------------------|
| project_id | Project ID |
| custom_field_id | Custom field ID |
| default_value | A default value for the field |
| status | Status |

get_api_key

Get API key

Description

Get API key

Usage

```
get_api_key()
```

Value

The API key.

Examples

```
## Not run:  
CLOCKIFY_API_KEY <- Sys.getenv("CLOCKIFY_API_KEY")  
set_api_key(CLOCKIFY_API_KEY)  
get_api_key()  
  
## End(Not run)
```

paginate

Title

Description

Title

Usage

```
paginate(path, query = NULL, pages = NULL, page_size = 50)
```

Arguments

| | |
|-----------|---------------------------------------|
| path | The path of the endpoint. |
| query | The query parameters. |
| pages | Maximum number of pages to retrieve. |
| page_size | Number of results requested per page. |

Value

Paginated response from API.

| | |
|---------|--------------------|
| project | <i>Get project</i> |
|---------|--------------------|

Description

Get project

Usage

```
project(project_id, concise = TRUE)
```

Arguments

| | |
|------------|-------------------------|
| project_id | Project ID |
| concise | Generate concise output |

Value

A data frame with one record per project

Examples

```
## Not run:  
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))  
  
project("612b16c0bc325f120a1e5099")  
  
## End(Not run)
```

| | |
|----------------|-----------------------|
| project-update | <i>Update project</i> |
|----------------|-----------------------|

Description

Adjust the project characteristics.

Usage

```
project_update(project_id, name = NULL, client_id = NULL, archived = NULL)  
  
project_update_template(project_id, is_template = TRUE)
```

Arguments

| | |
|-------------|--------------------------------------|
| project_id | Project ID |
| name | Project name |
| client_id | Client ID |
| archived | Whether or not project is archived |
| is_template | Whether or not project is a template |

Details

These functions enable the following functionality:

- change the project name
- change the client ID associated with the project
- toggle whether project is archived and
- toggle whether project is a template (paid plan only).

project-update-estimate

Update project time & budget estimates

Description

Update project time & budget estimates

Only available on a paid plan.

Usage

```
project_update_estimate_time(  
  project_id,  
  estimate = NULL,  
  manual = TRUE,  
  active = TRUE,  
  monthly = FALSE  
)
```

```
project_update_estimate_budget(  
  project_id,  
  estimate = NULL,  
  manual = TRUE,  
  active = TRUE,  
  monthly = FALSE  
)
```

Arguments

| | |
|------------|---|
| project_id | Project ID |
| estimate | Updated estimate |
| manual | Is the estimate for the whole project (TRUE) or should task-base estimate be enabled (FALSE). |
| active | Activate this estimate. Only one of either time or budget estimate may be active. |
| monthly | Should estimate be reset monthly? |

Examples

```
## Not run:
project_update_estimate_time("612b16c0bc325f120a1e5099", "PT1H0M0S", TRUE, TRUE)

## End(Not run)
## Not run:
project_update_estimate_budget("612b16c0bc325f120a1e5099", 1000, TRUE, TRUE)

## End(Not run)
```

| | |
|----------|---------------------|
| projects | <i>Get projects</i> |
|----------|---------------------|

Description

Get projects

Usage

```
projects(concise = TRUE)
```

Arguments

| | |
|---------|-------------------------|
| concise | Generate concise output |
|---------|-------------------------|

Value

A data frame with one record per project

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

projects()

## End(Not run)
```

| | |
|----------------|-----------------------|
| project_create | <i>Create project</i> |
|----------------|-----------------------|

Description

Create project

Usage

```
project_create(name, client_id = NULL)
```

Arguments

| | |
|-----------|--------------|
| name | Project name |
| client_id | Client ID |

| | |
|----------------|-----------------------|
| project_delete | <i>Delete project</i> |
|----------------|-----------------------|

Description

An active project cannot be deleted. Archive the project first.

Usage

```
project_delete(project_id)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
|------------|------------|

| | |
|------------------------------|---|
| project_update_billable_rate | <i>Update user billable rate on project</i> |
|------------------------------|---|

Description

Update user billable rate on project

Usage

```
project_update_billable_rate(project_id, user_id, rate, since = NULL)
```

Arguments

| | |
|------------|--|
| project_id | Project ID |
| user_id | User ID |
| rate | Rate |
| since | New rate will be applied to all time entries after this time |

project_update_cost_rate
Update user cost rate on project

Description

Only available on a paid plan.

Usage

project_update_cost_rate(project_id, user_id, rate, since = NULL)

Arguments

| | |
|------------|--|
| project_id | Project ID |
| user_id | User ID |
| rate | Rate |
| since | New rate will be applied to all time entries after this time |

project_update_memberships
Update project memberships

Description

Update project memberships

Usage

project_update_memberships(project_id, user_id)

Arguments

| | |
|------------|----------------------|
| project_id | Project ID |
| user_id | One or more user IDs |

| | |
|--------------------|---------------------------|
| reports-parameters | <i>Reports Parameters</i> |
|--------------------|---------------------------|

Description

These are parameters which occur commonly across functions for reports.

Arguments

| | |
|-------|------------|
| start | Start time |
| end | End time |

| | |
|------------------|------------------------|
| reports_detailed | <i>Detailed report</i> |
|------------------|------------------------|

Description

Detailed report

Usage

```
reports_detailed(start, end, extra_args = list())
```

Arguments

| | |
|------------|---|
| start | Start time |
| end | End time |
| extra_args | Extra arguments to be passed to the API . Example: <code>extra_args = list(rounding = TRUE)</code> . |

Value

A data frame with detailed time entries for the specified time period.

Examples

```
## Not run:  
report <- reports_detailed("2022-08-01", "2022-09-01")  
  
## End(Not run)
```

| | |
|-----------------|-----------------------|
| reports_summary | <i>Summary report</i> |
|-----------------|-----------------------|

Description

Summary report

Usage

```
reports_summary(start, end, extra_args = list())
```

Arguments

| | |
|------------|---|
| start | Start time |
| end | End time |
| extra_args | Extra arguments to be passed to the API . Example: <code>extra_args = list(rounding = TRUE)</code> . |

Value

A data frame with summarised time entries for the specified time period.

Examples

```
## Not run:
report <- reports_summary("2022-08-01", "2022-09-01")

# Summary per user.
report
# Summary per client/project.
report %>%
  select(-duration, -amount, -amounts) %>%
  unnest(projects)
# Summary per time entry.
report %>%
  select(-duration, -amount, -amounts) %>%
  unnest(projects) %>%
  select(-duration, -amount) %>%
  unnest(entries)

## End(Not run)
```

| | |
|----------------|----------------------|
| reports_weekly | <i>Weekly report</i> |
|----------------|----------------------|

Description

Weekly report

Usage

```
reports_weekly(start, end, extra_args = list())
```

Arguments

| | |
|------------|---|
| start | Start time |
| end | End time |
| extra_args | Extra arguments to be passed to the API . Example: <code>extra_args = list(rounding = TRUE)</code> . |

Value

A data frame with a weekly summary of time entries for the specified time period.

Examples

```
## Not run:
report <- reports_weekly("2022-08-01", "2022-08-08")

report %>%
  select(-duration, -amount) %>%
  unnest(projects)

## End(Not run)
```

| | |
|-------------|--------------------|
| set_api_key | <i>Set API key</i> |
|-------------|--------------------|

Description

Set API key

Usage

```
set_api_key(api_key)
```

Arguments

| | |
|---------|--------------------|
| api_key | A Clockify API key |
|---------|--------------------|

Value

The API key.

Examples

```
## Not run:
CLOCKIFY_API_KEY <- Sys.getenv("CLOCKIFY_API_KEY")
set_api_key(CLOCKIFY_API_KEY)

## End(Not run)
```

shared-reports-parameters

Shared Reports Parameters

Description

These are parameters which occur commonly across functions for shared reports.

Arguments

| | |
|------------------|---|
| shared_report_id | Identifier for a specific shared report |
| name | Name of the report |
| start | Start time |
| end | End time |
| is_public | Is this a public report? |
| fixed_date | Are the dates fixed? |

shared_report

Get a specific shared report

Description

Get a specific shared report

Usage

```
shared_report(shared_report_id)
```

Arguments

| | |
|------------------|---|
| shared_report_id | Identifier for a specific shared report |
|------------------|---|

Examples

```
## Not run:
# Get all shared reports.
shared_reports()
# Get specific shared report by shared report ID.
shared_report("6307f29f1bbd1d34e56b9eb7")

## End(Not run)
```

| | |
|----------------|-------------------------------|
| shared_reports | <i>Get all shared reports</i> |
|----------------|-------------------------------|

Description

Get all shared reports

Usage

```
shared_reports()
```

| | |
|----------------------|-------------------------------|
| shared_report_create | <i>Create a shared report</i> |
|----------------------|-------------------------------|

Description

Create a shared report

Usage

```
shared_report_create(name, start, end, is_public = TRUE, fixed_date = FALSE)
```

Arguments

| | |
|------------|--------------------------|
| name | Name of the report |
| start | Start time |
| end | End time |
| is_public | Is this a public report? |
| fixed_date | Are the dates fixed? |

Examples

```
## Not run:
shared_report_create("Sample Report", "2022-03-01", "2022-04-01")

## End(Not run)
```

shared_report_delete *Delete a shared report*

Description

Delete a shared report

Usage

```
shared_report_delete(shared_report_id)
```

Arguments

shared_report_id
Identifier for a specific shared report

Examples

```
## Not run:  
shared_report_delete("6307f29f1bbd1d34e56b9eb7", name = "Test Report")  
  
## End(Not run)
```

shared_report_update *Update a shared report*

Description

Update a shared report

Usage

```
shared_report_update(  
  shared_report_id,  
  name = NULL,  
  is_public = NULL,  
  fixed_date = NULL  
)
```

Arguments

shared_report_id
Identifier for a specific shared report

name
Report name

is_public
Is this a public report?

fixed_date
Are the dates fixed?

Examples

```
## Not run:
shared_report_update("6307f29f1bbd1d34e56b9eb7", name = "Test Report")

## End(Not run)
```

| | |
|-----|----------------|
| tag | <i>Get tag</i> |
|-----|----------------|

Description

Get tag

Usage

```
tag(tag_id)
```

Arguments

| | |
|--------|--------|
| tag_id | Tag ID |
|--------|--------|

Value

A data frame with one record per tag

Examples

```
## Not run:
tag("5f2d9bc659badb2a849c027e")

## End(Not run)
```

| | |
|------|-----------------|
| tags | <i>Get tags</i> |
|------|-----------------|

Description

Get tags

Usage

```
tags()
```

Value

A data frame.

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

tags()

## End(Not run)
```

| | |
|------------|-------------------|
| tag_create | <i>Create tag</i> |
|------------|-------------------|

Description

Create tag

Usage

```
tag_create(name)
```

Arguments

| | |
|------|----------|
| name | Tag name |
|------|----------|

Examples

```
## Not run:
tag_create("Size: S")
tag_create("Size: M")
tag_create("Size: L")
tag_create("Size: XL")

## End(Not run)
```

| | |
|------------|-------------------|
| tag_delete | <i>Delete tag</i> |
|------------|-------------------|

Description

Delete tag

Usage

```
tag_delete(tag_id)
```

Arguments

| | |
|--------|--------|
| tag_id | Tag ID |
|--------|--------|

Examples

```
## Not run:
tag_delete("5f2d9bc659badb2a849c027e")

## End(Not run)
```

| | |
|------------|-------------------|
| tag_update | <i>Update tag</i> |
|------------|-------------------|

Description

Update tag

Usage

```
tag_update(tag_id, name = NULL, archived = NULL)
```

Arguments

| | |
|----------|---------------------------------|
| tag_id | Tag ID |
| name | Tag name |
| archived | Whether or not item is archived |

Examples

```
## Not run:
tag_update("5f2d9bc659badb2a849c027e", "Size: Large")
tag_update("5f2d9bc659badb2a849c027e", archived = TRUE)
tag_update("5f2d9bc659badb2a849c027e", "Size: L", FALSE)

## End(Not run)
```

| | |
|------|-----------------|
| task | <i>Get task</i> |
|------|-----------------|

Description

Get task

Usage

```
task(project_id, task_id)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
| task_id | Task ID |

Value

A data frame.

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

workspace("61343c45ab05e02be2c8c1fd")
tasks("61343c9ba15c1d53ad33369f")

## End(Not run)
```

| | |
|-------------|----------------------|
| task-create | <i>Create a task</i> |
|-------------|----------------------|

Description

Create a task

Usage

```
task_create(project_id, name)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
| name | Task name |

Examples

```
## Not run:
task_create("630ce53290cfd8789366fd49", "tests")
task_create("630ce53290cfd8789366fd49", "docs")

## End(Not run)
```

| | |
|-------|------------------|
| tasks | <i>Get tasks</i> |
|-------|------------------|

Description

Get tasks

Usage

```
tasks(project_id)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
|------------|------------|

Value

A data frame.

Examples

```
## Not run:  
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))  
  
workspace("61343c45ab05e02be2c8c1fd")  
tasks("61343c9ba15c1d53ad33369f")  
  
## End(Not run)
```

| | |
|-------------|--------------------|
| task_delete | <i>Delete task</i> |
|-------------|--------------------|

Description

Delete task

Usage

```
task_delete(project_id, task_id)
```

Arguments

| | |
|------------|------------|
| project_id | Project ID |
| task_id | Task ID |

Examples

```
## Not run:
task_delete("630ce53290cfd8789366fd49", "630ce57e25e863294e5c6cf2")

## End(Not run)
```

| | |
|-------------|----------------------|
| task_update | <i>Update a task</i> |
|-------------|----------------------|

Description

Update a task

Usage

```
task_update(
  project_id,
  task_id,
  name,
  billable = NULL,
  status = NULL,
  assignee_id = NULL
)
```

Arguments

| | |
|-------------|-----------------------------|
| project_id | Project ID |
| task_id | Task ID |
| name | Task name |
| billable | Is the task billable? |
| status | Is the task ACTIVE or DONE? |
| assignee_id | Assignee ID |

Examples

```
## Not run:
task_update("630ce53290cfd8789366fd49", "630ce57e25e863294e5c6cf2", "Tests")
task_create("630ce53290cfd8789366fd49", "630ce80a7f07da44c14ca9a2", "Docs", FALSE)

## End(Not run)
```

task_update_billable_rate

Update task billable rate

Description

This feature is only available on the Standard, Pro and Enterprise plans.

Usage

task_update_billable_rate(project_id, task_id, rate, since = NULL)

Arguments

| | |
|------------|--|
| project_id | Project ID |
| task_id | Task ID |
| rate | Rate |
| since | New rate will be applied to all time entries after this time |

task_update_cost_rate *Update task cost rate*

Description

This feature is only available on the Pro and Enterprise plans.

Usage

task_update_cost_rate(project_id, task_id, rate, since = NULL)

Arguments

| | |
|------------|--|
| project_id | Project ID |
| task_id | Task ID |
| rate | Rate |
| since | New rate will be applied to all time entries after this time |

time-entry-parameters *Time Entry Parameters*

Description

These are parameters which occur commonly across functions for time entries.

Arguments

| | |
|---------------|---------------|
| time_entry_id | Time entry ID |
| project_id | Project ID |
| start | Start time |
| end | End time |
| description | Description |

time_entries *Get time entries*

Description

You send time according to your account's timezone (from Profile Settings) and get response with time in UTC.

Usage

```
time_entries(  
  user_id = NULL,  
  start = NULL,  
  end = NULL,  
  description = NULL,  
  project_id = NULL,  
  task = NULL,  
  tags = NULL,  
  finished = TRUE,  
  concise = TRUE,  
  ...  
)
```

Arguments

| | |
|-------------|---|
| user_id | User ID. If not specified then use authenticated user. |
| start | If provided, only time entries that started after the specified datetime will be returned. |
| end | If provided, only time entries that started before the specified datetime will be returned. |
| description | If provided, time entries will be filtered by description. |
| project_id | If provided, time entries will be filtered by project. |
| task | If provided, time entries will be filtered by task. |
| tags | If provided, time entries will be filtered by tags. You can provide one or more tags. |
| finished | Whether to include only finished time intervals (intervals with both start and end time). |
| concise | Generate concise output |
| ... | Further arguments passed to paginate . |

Value

A data frame with one record per time entry.

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

USER_ID <- "612b15a4f4c3bf0462192676"

# Specify number of results per page (default: 50).
time_entries(USER_ID, page_size = 200)
# Specify number of pages.
time_entries(USER_ID, pages = 3)

## End(Not run)
```

time_entry

Get a specific time entry on workspace

Description

Get a specific time entry on workspace

Usage

```
time_entry(time_entry_id, concise = TRUE)
```

Arguments

| | |
|---------------|-------------------------|
| time_entry_id | Time entry ID |
| concise | Generate concise output |

Value

A data frame with one record per time entry.

Examples

```
## Not run:
time_entry("61343d27ab05e02be2c8c266")

## End(Not run)
```

| | |
|-------------------|----------------------------|
| time_entry_create | <i>Create a time entry</i> |
|-------------------|----------------------------|

Description

Creating time entries for other users is a paid feature.

Usage

```
time_entry_create(
  user_id = NULL,
  project_id = NULL,
  start,
  end = NULL,
  description = NULL,
  task_id = NULL
)
```

Arguments

| | |
|-------------|-------------|
| user_id | User ID |
| project_id | Project ID |
| start | Start time |
| end | End time |
| description | Description |
| task_id | Task ID |

Value

A time entry ID.

Examples

```
## Not run:
# Create a time entry for the authenticated user.
time_entry_create(
  project_id = "600e73263e207962449a2c13",
  start = "2021-01-02 08:00:00",
  end = "2021-01-02 10:00:00",
  description = "Doing stuff"
)
# Create a time entry for another user (paid feature).
time_entry_create(
  "5df56293df753263139e60c5",
  "600e73263e207962449a2c13",
  "2021-01-02 10:00:00",
  "2021-01-02 12:00:00",
  "Doing other stuff"
)

## End(Not run)
```

| | |
|-------------------|----------------------------|
| time_entry_delete | <i>Delete a time entry</i> |
|-------------------|----------------------------|

Description

Delete a time entry

Usage

```
time_entry_delete(time_entry_id = NULL)
```

Arguments

time_entry_id Time entry ID

Value

A Boolean: TRUE on success or FALSE on failure.

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

time_entry_delete("612c7bd2a34530476ab25c67")

## End(Not run)
```

time_entry_invoiced *Mark time entries as invoiced*

Description

The time_entry_invoiced() function will only work on a paid plan.

Usage

```
time_entry_invoiced(time_entry_id, invoiced = TRUE)
```

Arguments

| | |
|---------------|------------------------------------|
| time_entry_id | Time entry ID |
| invoiced | Has this time entry been invoiced? |

time_entry_set *Replace a time entry*

Description

This does not update the time entry. It uses the same time entry ID but sets all details from scratch.

Usage

```
time_entry_set(  
  time_entry_id,  
  project_id = NULL,  
  start,  
  end = NULL,  
  description = NULL  
)
```

Arguments

| | |
|---------------|---------------|
| time_entry_id | Time entry ID |
| project_id | Project ID |
| start | Start time |
| end | End time |
| description | Description |

| | |
|-----------------|-------------------------------------|
| time_entry_stop | <i>Stop currently running timer</i> |
|-----------------|-------------------------------------|

Description

Stop currently running timer

Usage

```
time_entry_stop(user_id = NULL, end = NULL)
```

Arguments

| | |
|---------|--|
| user_id | User ID. If not specified then use authenticated user. |
| end | End time |

Examples

```
## Not run:  
# Start timer running.  
time_entry_create(  
  user_id = "5df56293df753263139e60c5",  
  project_id = "600e73263e207962449a2c13",  
  start = "2022-09-02 14:00:00",  
  description = "Doing other stuff"  
)  
# Stop timer.  
time_entry_stop(  
  user_id = "5df56293df753263139e60c5",  
  end = "2022-09-02 15:00:00"  
)  
  
## End(Not run)
```

| | |
|------|---|
| user | <i>Get information for authenticated user</i> |
|------|---|

Description

Get information for authenticated user

Usage

```
user(concise = TRUE)
```

Arguments

concise Generate concise output

Value

A data frame with details of user profile.

Examples

```
## Not run:
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))

user()

## End(Not run)
```

user-update-role *Update user roles*

Description

Update user roles

Usage

```
user_update_role(user_id, role, entity_id)
```

Arguments

user_id User ID

role One of "TEAM_MANAGER", "PROJECT_MANAGER" or "WORKSPACE_ADMIN".

entity_id Depending on role, this is a user ID (for "TEAM_MANAGER"), a project ID (for "PROJECT_MANAGER") or a workspace ID (for "WORKSPACE_ADMIN").

users *Get list of users in active workspace*

Description

Get list of users in active workspace

Usage

```
users(active = NULL, concise = TRUE)
```

Arguments

| | |
|---------|---------------------------|
| active | Only include active users |
| concise | Generate concise output |

Value

A data frame with one record per user.

Examples

```
## Not run:  
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))  
  
# Show only active users.  
users()  
# Show all users.  
users(active = FALSE)  
# Show active & default workspace for each user.  
users(concise = FALSE)  
  
## End(Not run)
```

| | |
|-------------|----------------------|
| user_create | <i>Create a user</i> |
|-------------|----------------------|

Description

Create a user

Usage

```
user_create(email, send_email = TRUE)
```

Arguments

| | |
|------------|-------------------------------|
| email | Email address for user |
| send_email | Whether to send email to user |

| | |
|-------------|--------------------|
| user_delete | <i>Delete user</i> |
|-------------|--------------------|

Description

Delete user

Usage

```
user_delete(user_id)
```

Arguments

| | |
|---------|---------|
| user_id | User ID |
|---------|---------|

| | |
|------------------|--------------------------|
| user_delete_role | <i>Delete user roles</i> |
|------------------|--------------------------|

Description

Delete user roles

Usage

```
user_delete_role(user_id, role, entity_id)
```

Arguments

| | |
|-----------|--|
| user_id | User ID |
| role | One of "TEAM_MANAGER", "PROJECT_MANAGER" or "WORKSPACE_ADMIN". |
| entity_id | Depending on role, this is a user ID (for "TEAM_MANAGER"), a project ID (for "PROJECT_MANAGER") or a workspace ID (for "WORKSPACE_ADMIN"). |

| | |
|-------------|------------------------|
| user_groups | <i>Get user groups</i> |
|-------------|------------------------|

Description

Get user groups

Usage

```
user_groups()
```

Value

A data frame with one record per user group.

Examples

```
## Not run:  
user_groups()  
  
## End(Not run)
```

| | |
|-------------------|----------------------------|
| user_group_create | <i>Create a user group</i> |
|-------------------|----------------------------|

Description

Create a user group

Usage

```
user_group_create(name)
```

Arguments

| | |
|------|--------------------|
| name | Name of user group |
|------|--------------------|

| | |
|-------------------|----------------------------|
| user_group_delete | <i>Delete a user group</i> |
|-------------------|----------------------------|

Description

Delete a user group

Usage

```
user_group_delete(group_id)
```

Arguments

| | |
|----------|---------------|
| group_id | User group ID |
|----------|---------------|

| | |
|-------------------|----------------------------|
| user_group_update | <i>Update a user group</i> |
|-------------------|----------------------------|

Description

Update a user group

Usage

```
user_group_update(group_id, name)
```

Arguments

| | |
|----------|--------------------|
| group_id | User group ID |
| name | Name of user group |

| | |
|---------------------|-----------------------------------|
| user_group_user_add | <i>Add a user to a user group</i> |
|---------------------|-----------------------------------|

Description

Add a user to a user group

Usage

```
user_group_user_add(group_id, user_id)
```

Arguments

| | |
|----------|---------------|
| group_id | User group ID |
| user_id | User ID |

`user_group_user_remove`*Remove a user from a user group*

Description

Remove a user from a user group

Usage

```
user_group_user_remove(group_id, user_id)
```

Arguments

| | |
|-----------------------|---------------|
| <code>group_id</code> | User group ID |
| <code>user_id</code> | User ID |

`user_update_cost_rate` *Update cost rate*

Description

For this to work you need to enable expenses (under the *General* tab in *Workspace Settings*). It's only available on the PRO plan.

Usage

```
user_update_cost_rate(user_id, rate, since = NULL)
```

Arguments

| | |
|----------------------|--|
| <code>user_id</code> | User ID |
| <code>rate</code> | Rate |
| <code>since</code> | New rate will be applied to all time entries after this time |

user_update_hourly_rate
Update hourly rate

Description

Update hourly rate

Usage

```
user_update_hourly_rate(user_id, rate, since = NULL)
```

Arguments

| | |
|---------|--|
| user_id | User ID |
| rate | Rate |
| since | New rate will be applied to all time entries after this time |

user_update_status *Update status*

Description

Update status

Usage

```
user_update_status(user_id, active)
```

Arguments

| | |
|---------|---|
| user_id | User ID |
| active | A Boolean indicating whether or not user is active. Can also specify either "ACTIVE" or "INACTIVE". |

| | |
|-----------|---------------------------------------|
| workspace | <i>Get or set active workspace ID</i> |
|-----------|---------------------------------------|

Description

Get or set active workspace ID

Usage

```
workspace(workspace_id = NULL)
```

Arguments

workspace_id A workspace ID

Value

The ID of the active workspace.

Examples

```
## Not run:  
# Select default workspace for authenticated user.  
workspace()  
# Select a specific workspace.  
workspace("612b15a5f4c3bf0462192677")  
  
## End(Not run)
```

| | |
|------------|---------------------------------|
| workspaces | <i>Get a list of workspaces</i> |
|------------|---------------------------------|

Description

Get a list of workspaces

Usage

```
workspaces()
```

Value

A data frame with one record per workspace.

Examples

```
## Not run:  
set_api_key(Sys.getenv("CLOCKIFY_API_KEY"))  
  
workspaces()  
  
## End(Not run)
```

Index

- client, 3
- client-parameters, 4
- client_create, 5
- client_delete, 5
- client_update, 6
- clients, 4
- custom_field_delete, 7
- custom_field_update, 7
- custom_fields, 6

- get_api_key, 8

- paginate, 8, 28
- project, 9
- project-update, 9
- project-update-estimate, 10
- project_create, 12
- project_delete, 12
- project_update (project-update), 9
- project_update_billable_rate, 12
- project_update_cost_rate, 13
- project_update_estimate_budget
 (project-update-estimate), 10
- project_update_estimate_time
 (project-update-estimate), 10
- project_update_memberships, 13
- project_update_template
 (project-update), 9
- projects, 11

- reports-parameters, 14
- reports_detailed, 14
- reports_summary, 15
- reports_weekly, 16

- set_api_key, 16
- shared-reports-parameters, 17
- shared_report, 17
- shared_report_create, 18
- shared_report_delete, 19

- shared_report_update, 19
- shared_reports, 18

- tag, 20
- tag_create, 21
- tag_delete, 21
- tag_update, 22
- tags, 20
- task, 22
- task-create, 23
- task_create (task-create), 23
- task_delete, 24
- task_update, 25
- task_update_billable_rate, 26
- task_update_cost_rate, 26
- tasks, 24
- time-entry-parameters, 27
- time_entries, 27
- time_entry, 28
- time_entry_create, 29
- time_entry_delete, 30
- time_entry_invoiced, 31
- time_entry_set, 31
- time_entry_stop, 32

- user, 32
- user-update-role, 33
- user_create, 34
- user_delete, 35
- user_delete_role, 35
- user_group_create, 36
- user_group_delete, 37
- user_group_update, 37
- user_group_user_add, 37
- user_group_user_remove, 38
- user_groups, 36
- user_update_cost_rate, 38
- user_update_hourly_rate, 39
- user_update_role (user-update-role), 33
- user_update_status, 39

users, [33](#)

workspace, [40](#)

workspaces, [40](#)