

# Package ‘gitear’

September 18, 2020

**Title** Client to the 'gitea' API

**Version** 1.0.0

**Description** 'Gitea' is a community managed, lightweight code hosting solution where projects and their respective git repositories can be managed <<https://gitea.io>>. This package gives an interface to the 'Gitea' API to access and manage repositories, issues and organizations directly in R.

**Depends** R (>= 3.5.0), magrittr

**Imports** httr, jsonlite, dplyr, tidyr, tibble, stringr, mockery, graphics, Rcpp

**License** GPL-3

**URL** <https://ixpantia.github.io/gitear/>

**BugReports** <https://github.com/ixpantia/gitear/issues>

**Encoding** UTF-8

**LazyData** true

**Suggests** testthat, knitr, rmarkdown, covr

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** ixpantia, SRL [aut],  
Frans van Dunné [cre, aut],  
Francisco Sácida [aut],  
Ronny Hernández Mora [aut],  
Patrick Santamaría Guzmán [ctb]

**Maintainer** Frans van Dunné <[frans@ixpantia.com](mailto:frans@ixpantia.com)>

**Repository** CRAN

**Date/Publication** 2020-09-18 08:00:07 UTC

**R topics documented:**

add_tracked_time_issue . . . . .	2
create_comment_issue . . . . .	3
create_issue . . . . .	4
edit_comment . . . . .	5
edit_issue . . . . .	5
get_admin_organizations . . . . .	6
get_an_organization . . . . .	7
get_branches . . . . .	7
get_commits . . . . .	8
get_forks . . . . .	9
get_issues . . . . .	9
get_issues_closed_state . . . . .	10
get_issues_open_state . . . . .	11
get_label_issue . . . . .	11
get_list_comments_issue . . . . .	12
get_list_comments_repository . . . . .	13
get_list_org_members . . . . .	13
get_list_org_teams . . . . .	14
get_list_repos_org . . . . .	14
get_list_users . . . . .	15
get_milestones . . . . .	16
get_organizations . . . . .	16
get_org_hook . . . . .	17
get_org_list_hooks . . . . .	17
get_pull_requests . . . . .	18
get_releases . . . . .	19
get_repositories . . . . .	19
get_times_issue . . . . .	20
get_version . . . . .	21
gitgear . . . . .	21
<b>Index</b>	<b>22</b>

---

add\_tracked\_time\_issue

*Add tracked time to an issue*

---

**Description**

Add a tracked time to an issue

**Usage**

add\_tracked\_time\_issue(base\_url, api\_key, owner, repo, id\_issue, time)

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_issue	Index of the issue to add tracked time to
time	The time in seconds to add to the issue

**Examples**

```
## Not run:
add_tracked_time_issue(base_url = "https://example.gitea.service.com",
                       api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                       owner = "company",
                       repo = "test_repo",
                       id_issue = 2,
                       time = 15)

## End(Not run)
```

---

create\_comment\_issue *Add a comment to an issue*

---

**Description**

Add a comment to an issue in a gitea server

**Usage**

```
create_comment_issue(base_url, api_key, owner, repo, id_issue, body)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_issue	Index of the issue to create a comment
body	The text that is added as a comment to the issue

**Examples**

```
## Not run:
create_comment_issue(base_url = "https://example.gitea.service.com",
                    api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                    owner = "company",
                    repo = "test_repo",
                    id_issue = 2,
                    body = "my first comment on this issue")

## End(Not run)
```

---

create_issue	<i>Create a new issue</i>
--------------	---------------------------

---

**Description**

Create an new issue in a specific repository

**Usage**

```
create_issue(base_url, api_key, owner, repo, title, body)
```

**Arguments**

base_url	URL prefix for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
title	The title of the issue
body	The body text of the issue

**Value**

list with results of the new issue

**Examples**

```
## Not run:
create_issue(base_url = "https://example.gitea.service.com",
            api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
            owner = "company",
            repo = "test_repo",
            title = "Perform clean code task",
            body = "Perform in an orderly manner and document steps")

## End(Not run)
```

---

edit_comment	<i>Edit a comment</i>
--------------	-----------------------

---

**Description**

Edit a comment in a specific issue

**Usage**

```
edit_comment(base_url, api_key, owner, repo, id_comment, body)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_comment	Id of the comment to edit
body	The text to replace the old comment

**Examples**

```
## Not run:
edit_comment(base_url = "https://example.gitea.service.com",
             api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
             owner = "company",
             repo = "test_repo",
             id_comment = 612,
             body = "This is the correction of my comment")

## End(Not run)
```

---

edit_issue	<i>Edit an issue</i>
------------	----------------------

---

**Description**

Edit an issue

**Usage**

```
edit_issue(base_url, api_key, owner, repo, id_issue, title, body, state)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repo
repo	The name of the repo
id_issue	Index of the issue to edit
title	The new issue title text
body	The new issue body text
state	The issue state

**Examples**

```
## Not run:
edit_issue(base_url = "https://example.gitea.service.com",
           api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
           owner = "company",
           repo = "test_repo",
           id_issue = 3,
           title = "My new title for this issue",
           body = "My new comment starts on this issue",
           state = "open")

## End(Not run)
```

---

get\_admin\_organizations

*Returns organizations for an administrator user*

---

**Description**

Returns the list of organizations for a user with an administrator role

**Usage**

```
get_admin_organizations(base_url, api_key)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service

**Details**

This function works only in the case that the 'api\_key' is associated with a user with administrator role

### Examples

```
## Not run:  
get_admin_organizations(base_url = "https://example.gitea.service.com",  
                        api_key = "b6026f861fd41a94c3389d54293de9d04bde6f7c")  
  
## End(Not run)
```

---

get\_an\_organization     *Returns an organization*

---

### Description

Get information from an organization

### Usage

```
get_an_organization(base_url, api_key, org)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization

### Examples

```
## Not run:  
get_an_organization(base_url = "https://example.gitea.service.com",  
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",  
                  org = "company")  
  
## End(Not run)
```

---

get\_branches     *Returns the branches of a repository*

---

### Description

Returns branches in a specific repository

### Usage

```
get_branches(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:
get_branches(base_url = "https://example.gitea.service.com",
             api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
             owner = "company",
             repo = "test_repo")

## End(Not run)
```

---

get_commits	<i>Returns repository commits</i>
-------------	-----------------------------------

---

**Description**

Returns the repository commits of a Gitea application

**Usage**

```
get_commits(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:
get_commits(base_url = "https://example.gitea.service.com",
            api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
            owner = "company",
            repo = "test_repo")

## End(Not run)
```

---

get\_forks                      *Returns the forks of a repository*

---

**Description**

Returns forks in a specific repository

**Usage**

```
get_forks(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:
get_forks(base_url = "https://example.gitea.service.com",
          api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
          owner = "company",
          repo = "test_repo")

## End(Not run)
```

---

get\_issues                      *Returns open issues from an specific repository*

---

**Description**

Returns open issues in an specific repository

**Usage**

```
get_issues(base_url, api_key, owner, repo, full_info = FALSE)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
full_info	TRUE or FALSE value. If FALSE this will select specific columns from the issues data



---

get\_issues\_open\_state *Returns the repository issues in open state*

---

### Description

Returns the issues in open state of a repository

### Usage

```
get_issues_open_state(base_url, api_key, owner, repo)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

### Examples

```
## Not run:
get_issues_open_state (base_url = "https://example.gitea.service.com",
                      api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                      owner = "company",
                      repo = "test_repo")

## End(Not run)
```

---

get\_label\_issue *Returns an issue's labels*

---

### Description

Returns an issue's labels

### Usage

```
get_label_issue(base_url, api_key, owner, repo, id_issue)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_issue	Index of the issue

## Examples

```
## Not run:
get_label_issue(base_url = "https://example.gitea.service.com",
                api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                owner = "company",
                repo = "test_repo",
                id_issue = 3)

## End(Not run)
```

---

get\_list\_comments\_issue

*Returns all comments on an issue*

---

## Description

Returns a list of all comments on an issue

## Usage

```
get_list_comments_issue(base_url, api_key, owner, repo, id_issue)
```

## Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_issue	Index of the issue to get comments

## Examples

```
## Not run:
get_list_comments_issue(base_url = "https://example.gitea.service.com",
                        api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                        owner = "company",
                        repo = "test_repo",
                        id_issue = 3)

## End(Not run)
```

---

get\_list\_comments\_repository  
*Returns all comments in a repository*

---

**Description**

Returns a list of all comments in a repository

**Usage**

```
get_list_comments_repository(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:  
get_list_comments_repository(base_url = "https://example.gitea.service.com",  
                             api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",  
                             owner = "company",  
                             repo = "test_repo")  
  
## End(Not run)
```

---

get\_list\_org\_members *Returns organization's members*

---

**Description**

List an organization's members

**Usage**

```
get_list_org_members(base_url, api_key, org)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization

### Examples

```
## Not run:
get_list_org_members(base_url = "https://example.gitea.service.com",
                    api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                    org = "company")

## End(Not run)
```

---

get\_list\_org\_teams      *Returns organization's teams*

---

### Description

Get organization's teams

### Usage

```
get_list_org_teams(base_url, api_key, org)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization

### Examples

```
## Not run:
get_list_org_teams(base_url = "https://example.gitea.service.com",
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                  org = "company")

## End(Not run)
```

---

get\_list\_repos\_org      *Returns organization's repositories*

---

### Description

Get a list of an organization's repositories

### Usage

```
get_list_repos_org(base_url, api_key, org)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization

**Examples**

```
## Not run:
get_list_repos_org(base_url = "https://example.gitea.service.com",
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                  org = "company")

## End(Not run)
```

---

get_list_users	<i>Returns users of a gitea server</i>
----------------	--

---

**Description**

User list for a gitea server

**Usage**

```
get_list_users(base_url, api_key)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service

**Examples**

```
## Not run:
get_list_users(base_url = "https://example.gitea.service.com",
              api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288")

## End(Not run)
```

---

get\_milestones      *Returns open milestones of a repository*

---

### Description

Returns open milestones in a specific repository

### Usage

```
get_milestones(base_url, api_key, owner, repo)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

### Examples

```
## Not run:
get_milestones(base_url = "https://example.gitea.service.com",
               api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
               owner = "company",
               repo = "test_repo")

## End(Not run)
```

---

get\_organizations      *Returns organizations of the user*

---

### Description

Returns list the current user's organizations

### Usage

```
get_organizations(base_url, api_key)
```

### Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service

**Examples**

```
## Not run:
get_organizations(base_url = "https://example.gitea.service.com",
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288")

## End(Not run)
```

---

get_org_hook	<i>Returns a hook</i>
--------------	-----------------------

---

**Description**

Get a hook information of a organizations

**Usage**

```
get_org_hook(base_url, api_key, org, id_hook)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization
id_hook	Id of the hook to get information

**Examples**

```
## Not run:
get_org_hook(base_url = "https://example.gitea.service.com",
              api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
              org = "company",
              id_hook = 2)

## End(Not run)
```

---

get_org_list_hooks	<i>Returns organization's webhooks</i>
--------------------	--

---

**Description**

Returns a list of organization's webhooks

**Usage**

```
get_org_list_hooks(base_url, api_key, org)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
org	Name of the organization

**Examples**

```
## Not run:
get_org_list_hooks(base_url = "https://example.gitea.service.com",
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                  org = "company")

## End(Not run)
```

---

get\_pull\_requests      *Returns pull requests of a repository*

---

**Description**

Returns open and closed pull requests in a specific repository

**Usage**

```
get_pull_requests(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:
get_pull_requests(base_url = "https://example.gitea.service.com",
                  api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
                  owner = "company",
                  repo = "test_repo")

## End(Not run)
```

---

get_releases	<i>Returns releases of a repository</i>
--------------	---

---

**Description**

Returns releases in a specific repository

**Usage**

```
get_releases(base_url, api_key, owner, repo)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository

**Examples**

```
## Not run:  
get_releases(base_url = "https://example.gitea.service.com",  
             api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",  
             owner = "company",  
             repo = "test_repo")  
  
## End(Not run)
```

---

get_repositories	<i>Returns repositories of a gitea service</i>
------------------	--

---

**Description**

Returns the repositories of the Gitea application

**Usage**

```
get_repositories(base_url, api_key)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service

## Examples

```
## Not run:
get_repositories(base_url = "https://example.gitea.service.com",
                 api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288")

## End(Not run)
```

---

get_times_issue	<i>Returns issue's tracked times</i>
-----------------	--------------------------------------

---

## Description

Returns a data frame of an issue's tracked times

## Usage

```
get_times_issue(base_url, api_key, owner, repo, id_issue)
```

## Arguments

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service
owner	The owner of the repository
repo	The name of the repository
id_issue	Index of the issue

## Examples

```
## Not run:
get_times_issue(base_url = "https://example.gitea.service.com",
               api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288",
               owner = "company",
               repo = "test_repo",
               id_issue = 3)

## End(Not run)
```

---

get_version	Returns gitea service version
-------------	-------------------------------

---

**Description**

Returns the version of the Gitea application

**Usage**

```
get_version(base_url, api_key)
```

**Arguments**

base_url	The base URL for your gitea server (no trailing '/')
api_key	The user's API token key for the gitea service

**Examples**

```
## Not run:  
get_version(base_url = "https://example.gitea.service.com",  
            api_key = "ccaf5c9a22e854856d0c5b1b96c81e851bafb288")  
  
## End(Not run)
```

---

gitear	<i>gitear package</i>
--------	-----------------------

---

**Description**

'Gitea' is a community managed, lightweight code hosting solution where projects and their respective git repositories can be managed access and manage repositories, issues and organizations directly in R.

# Index

`add_tracked_time_issue`, 2

`create_comment_issue`, 3  
`create_issue`, 4

`edit_comment`, 5  
`edit_issue`, 5

`get_admin_organizations`, 6  
`get_an_organization`, 7  
`get_branches`, 7  
`get_commits`, 8  
`get_forks`, 9  
`get_issues`, 9  
`get_issues_closed_state`, 10  
`get_issues_open_state`, 11  
`get_label_issue`, 11  
`get_list_comments_issue`, 12  
`get_list_comments_repository`, 13  
`get_list_org_members`, 13  
`get_list_org_teams`, 14  
`get_list_repos_org`, 14  
`get_list_users`, 15  
`get_milestones`, 16  
`get_org_hook`, 17  
`get_org_list_hooks`, 17  
`get_organizations`, 16  
`get_pull_requests`, 18  
`get_releases`, 19  
`get_repositories`, 19  
`get_times_issue`, 20  
`get_version`, 21  
`gitear`, 21