

Package ‘telegram’

July 22, 2025

Title R Wrapper Around the Telegram Bot API

Version 0.7.1

Author Luca Braglia [aut, cre]

Maintainer Luca Braglia <lbraglia@gmail.com>

Description A simple wrapper around the Telegram Bot API (<<https://core.telegram.org/bots/api>>) to access Telegram's messaging facilities with ease (e.g. you send messages, images, files from R to your smartphone).

URL <https://github.com/lbraglia/telegram>

BugReports <https://github.com/lbraglia/telegram/issues>

Imports R6, httr, jsonlite, curl, utils

Depends R (>= 3.2.3)

License GPL-3

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2025-03-25 08:00:06 UTC

Contents

bot_token	2
forwardMessage	2
getFile	3
getMe	3
getUpdates	4
getUserProfilePhotos	4
group_id	5
proxy	5
sendAudio	6
sendChatAction	6
sendDice	7
sendDocument	7

sendLocation	8
sendMessage	9
sendPhoto	9
sendPoll	10
sendSticker	11
sendVideo	11
sendVoice	12
stopPoll	13
telegram	13
TGBot	14
user_id	19

Index	20
--------------	-----------

bot_token	<i>bot_token</i>
-----------	------------------

Description

Obtain token from system variables (in Renviron) set according to package naming conventions, that is R_TELEGRAM_BOT_ where X is bot’s name (first question answered to the botfather).

Usage

bot_token(x)

Arguments

x character of length 1 with the name of the bot; if NULL a menu to choose between bot is displayed and the proper value returned

Examples

```
## Not run:  bot_token('RBot')
```

forwardMessage	<i>forwardMessage</i>
----------------	-----------------------

Description

Forward messages of any kind

Usage

```
forwardMessage(  
    from_chat_id = NULL,  
    message_id = NULL,  
    disable_notification = NULL,  
    chat_id = NULL  
)
```

Arguments

from_chat_id	Unique identifier for the chat where the original message was sent (required)
message_id	Unique message identifier (required)
disable_notification	Sends the message silently. Users will receive a notification with no sound
chat_id	Unique identifier for the target chat or username of the target channel (required)

`getFile`*getFile*

Description

Get info about a file and download it

Usage

```
getFile(file_id, destfile = NULL)
```

Arguments

file_id	File identifier (required)
destfile	Destination path; if specified the file will be downloaded

`getMe`*getMe*

Description

Test your bot's auth token

Usage

```
getMe()
```

`getUpdates`*getUpdates*

Description

Receive incoming updates

Usage

```
getUpdates(offset = NULL, limit = NULL)
```

Arguments

<code>offset</code>	Identifier of the first update to be returned.
<code>limit</code>	Limits the number of updates to be retrieved. Values between 1-100 are accepted. Defaults to 100

`getUserProfilePhotos`*getUserProfilePhotos*

Description

Get a list of profile pictures for a user

Usage

```
getUserProfilePhotos(user_id = NULL, offset = NULL, limit = NULL)
```

Arguments

<code>user_id</code>	Unique identifier of the target user (required)
<code>offset</code>	Sequential number of the first photo to be returned. By default, all photos are returned
<code>limit</code>	Limits the number of photos to be retrieved. Values between 1-100 are accepted. Defaults to 100

group_id

*group_id***Description**

Obtain telegram user id from system variables (in Renviron) set according to package naming conventions, that is R_TELEGRAM_USER_X where X is the user's name .

Usage

```
group_id(x)
```

Arguments

x character of length 1 with the name of the user; if NULL a menu to choose between bot is displayed and the proper value returned

Examples

```
## Not run: group_id('test_group')
```

proxy

*proxy***Description**

Obtain telegram proxy from system variables (in Renviron) set according to package naming conventions, that is R_TELEGRAM_PROXY_X_Y where X is the proxy's name and Y are the proxy parameters.

Usage

```
proxy(x)
```

Arguments

x character of length 1 with the name of the proxy; if NULL a menu to choose between proxy is displayed and the proper value returned

Examples

```
## Not run: proxy('default')
```

`sendAudio`*sendAudio*

Description

Send mp3 files

Usage

```
sendAudio(  
    audio = NULL,  
    caption = NULL,  
    duration = NULL,  
    performer = NULL,  
    title = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Arguments

<code>audio</code>	path to audio file to send (required)
<code>caption</code>	audio caption
<code>duration</code>	duration of the audio in seconds
<code>performer</code>	performer
<code>title</code>	track name
<code>disable_notification</code>	Sends the message silently. Users will receive a notification with no sound
<code>reply_to_message_id</code>	If the message is a reply, ID of the original message
<code>chat_id</code>	Unique identifier for the target chat or username of the target channel (required)

`sendChatAction`*sendChatAction*

Description

Tell the user that something is happening on the bot's side

Usage

```
sendChatAction(action = NULL, chat_id = NULL)
```

Arguments

action	type of action to broadcast. Choose one, depending on what the user is about to receive: "typing" for text messages, "upload_photo" for photos, "record_video" or "upload_video" for videos, "record_voice" or "upload_voice" for voice notes, "upload_document" for general files, "find_location" for location data, "record_video_note" or "upload_video_note" for video notes.
chat_id	Unique identifier for the target chat or username of the target channel (required)

`sendDice`*sendDice*

Description

Use this method to send a dice, which will have a random value from 1 to 6.

Usage

```
sendDice(disable_notification = NULL, chat_id = NULL)
```

Arguments

disable_notification	Sends the message silently. Users will receive a notification with no sound
chat_id	Unique identifier for the target chat or username of the target channel (required)

`sendDocument`*sendDocument*

Description

Send general files

Usage

```
sendDocument(  
  document = NULL,  
  disable_notification = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Arguments

- document path to the file to send (required)
- disable_notification Sends the message silently. Users will receive a notification with no sound
- reply_to_message_id if the message is a reply, ID of the original message
- chat_id Unique identifier for the target chat or username of the target channel (required)

sendLocation	<i>sendLocation</i>
--------------	---------------------

Description

Send point on the map

Usage

```
sendLocation(  
  latitude = NULL,  
  longitude = NULL,  
  disable_notification = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Arguments

- latitude Latitude of location (required)
- longitude Longitude of location (required)
- disable_notification Sends the message silently. Users will receive a notification with no sound
- reply_to_message_id If the message is a reply, ID of the original message
- chat_id Unique identifier for the target chat or username of the target channel (required)

sendMessage	<i>sendMessage</i>
-------------	--------------------

Description

Send text messages

Usage

```
sendMessage(  
    text = NULL,  
    parse_mode = NULL,  
    disable_web_page_preview = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Arguments

text	Text of the message to be sent (required)
parse_mode	send 'Markdown' if you want Telegram apps to show bold, italic and inline URLs in your bot's message
disable_web_page_preview	Disables link previews for links in this message
disable_notification	Sends the message silently. Users will receive a notification with no sound
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

sendPhoto	<i>sendPhoto</i>
-----------	------------------

Description

Send image files

Usage

```
sendPhoto(  
    photo = NULL,  
    caption = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Arguments

photo	photo to send (required)
caption	photo caption
disable_notification	Sends the message silently. Users will receive a notification with no sound
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

 sendPoll

sendPoll

Description

Use this method to send a native poll.

Usage

```

sendPoll(
    question = NULL,
    options = NULL,
    is_anonymous = TRUE,
    type = c("regular", "quiz"),
    allows_multiple_answers = FALSE,
    correct_option_id = NULL,
    is_closed = FALSE,
    disable_notification = NULL,
    reply_to_message_id = NULL,
    chat_id = NULL
)

```

Arguments

question	Poll question, 1-255 characters
options	vector of possible answers: 2-10 strings 1-100 characters each
is_anonymous	if TRUE (default) the poll needs to be anonymous
type	poll type, 'quiz' or 'regular', defaults to 'regular'
allows_multiple_answers	if TRUE the poll allows multiple answers (default to FALSE and ignored for polls in quiz mode)
correct_option_id	0-based identifier of the correct answer option, required for polls in quiz mode (eg 0 set the first answer as correct)

is_closed	if TRUE the poll needs to be immediately closed. This can be useful for poll preview. (default to FALSE)
disable_notification	Sends the message silently. Users will receive a notification with no sound
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

sendSticker	<i>sendSticker</i>
-------------	--------------------

Description

Send .webp stickers

Usage

```
sendSticker(
  sticker = NULL,
  disable_notification = NULL,
  reply_to_message_id = NULL,
  chat_id = NULL
)
```

Arguments

sticker	sticker to send (required)
disable_notification	Sends the message silently. Users will receive a notification with no sound
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

sendVideo	<i>sendVideo</i>
-----------	------------------

Description

Send mp4 videos

Usage

```
sendVideo(  
    video = NULL,  
    duration = NULL,  
    caption = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Arguments

video	Video to send (required)
duration	Duration of sent video in seconds
caption	Video caption
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

sendVoice	<i>sendVoice</i>
-----------	------------------

Description

Send .ogg files encoded with OPUS

Usage

```
sendVoice(  
    voice = NULL,  
    duration = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Arguments

voice	Audio file to send (required)
duration	Duration of sent audio in seconds
disable_notification	Sends the message silently. Users will receive a notification with no sound
reply_to_message_id	If the message is a reply, ID of the original message
chat_id	Unique identifier for the target chat or username of the target channel (required)

stopPoll	<i>stopPoll</i>
----------	-----------------

Description

Use this method to stop a poll which was sent by the bot.

Usage

```
stopPoll(message_id = NULL, chat_id = NULL)
```

Arguments

message_id	Identifier of the original message with the poll
chat_id	Unique identifier for the target chat or username of the target channel (required)

telegram	<i>telegram.</i>
----------	------------------

Description

Simple R wrapper around the Telegram Bot API (<http://core.telegram.org/bots/api>) to access Telegram's messaging facilities with ease (eg you send messages, images, files from R to your smart-phone)

Author(s)

Maintainer: Luca Braglia <lbraglia@gmail.com>

See Also

Useful links:

- <https://github.com/lbraglia/telegram>
- Report bugs at <https://github.com/lbraglia/telegram/issues>

TGBot

TGBot

Description

Package main class (implementing the Telegram bot).

Format

An `R6Class` generator object.

API Methods

`forwardMessage` forward messages of any kind
`getFile` get info about a file and download it
`getMe` test your bot's auth token
`getUpdates` receive incoming updates
`getUserProfilePhotos` get a list of profile pictures for a user
`sendAudio` send mp3 files
`sendDocument` send general files
`sendLocation` send point on the map
`sendMessage` send text messages
`sendPhoto` send image files
`sendSticker` send .webp stickers
`sendVideo` send mp4 videos
`sendVoice` send ogg files encoded with OPUS
`sendPoll` send a telegram poll
`stopPoll` stop a telegram poll and obtain results

Methods

Public methods:

- `TGBot$new()`
- `TGBot$set_token()`
- `TGBot$set_proxy()`
- `TGBot$set_default_chat_id()`
- `TGBot$req()`
- `TGBot$last_request()`
- `TGBot$print()`
- `TGBot$forwardMessage()`
- `TGBot$getFile()`

- TGBot\$getMe()
- TGBot\$getUpdates()
- TGBot\$getUserProfilePhotos()
- TGBot\$sendAudio()
- TGBot\$sendChatAction()
- TGBot\$sendDocument()
- TGBot\$sendLocation()
- TGBot\$sendMessage()
- TGBot\$sendPhoto()
- TGBot\$sendSticker()
- TGBot\$sendVideo()
- TGBot\$sendVoice()
- TGBot\$sendDice()
- TGBot\$sendPoll()
- TGBot\$stopPoll()
- TGBot\$clone()

Method new():*Usage:*`TGBot$new(token, proxy)`**Method set_token():***Usage:*`TGBot$set_token(token)`**Method set_proxy():***Usage:*`TGBot$set_proxy(proxy)`**Method set_default_chat_id():***Usage:*`TGBot$set_default_chat_id(chat_id)`**Method req():***Usage:*`TGBot$req(method = NULL, body = NULL)`**Method last_request():***Usage:*`TGBot$last_request()`**Method print():***Usage:*`TGBot$print()`

Method forwardMessage():*Usage:*

```
TGBot$forwardMessage(  
  from_chat_id = NULL,  
  message_id = NULL,  
  disable_notification = NULL,  
  chat_id = NULL  
)
```

Method getFile():*Usage:*

```
TGBot$getFile(file_id, destfile = NULL)
```

Method getMe():*Usage:*

```
TGBot$getMe()
```

Method getUpdates():*Usage:*

```
TGBot$getUpdates(offset = NULL, limit = NULL)
```

Method getUserProfilePhotos():*Usage:*

```
TGBot$getUserProfilePhotos(user_id = NULL, offset = NULL, limit = NULL)
```

Method sendAudio():*Usage:*

```
TGBot$sendAudio(  
  audio = NULL,  
  caption = NULL,  
  duration = NULL,  
  performer = NULL,  
  title = NULL,  
  disable_notification = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Method sendChatAction():*Usage:*

```
TGBot$sendChatAction(action = NULL, chat_id = NULL)
```

Method sendDocument():*Usage:*

```
TGBot$sendDocument(  
    document = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Method sendLocation():*Usage:*

```
TGBot$sendLocation(  
    latitude = NULL,  
    longitude = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Method sendMessage():*Usage:*

```
TGBot$sendMessage(  
    text = NULL,  
    parse_mode = NULL,  
    disable_web_page_preview = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Method sendPhoto():*Usage:*

```
TGBot$sendPhoto(  
    photo = NULL,  
    caption = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Method sendSticker():*Usage:*

```
TGBot$sendSticker(  
    sticker = NULL,  
    disable_notification = NULL,  
    reply_to_message_id = NULL,  
    chat_id = NULL  
)
```

Method sendVideo():

Usage:

```
TGBot$sendVideo(  
  video = NULL,  
  duration = NULL,  
  caption = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Method sendVoice():

Usage:

```
TGBot$sendVoice(  
  voice = NULL,  
  duration = NULL,  
  disable_notification = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Method sendDice():

Usage:

```
TGBot$sendDice(disable_notification = NULL, chat_id = NULL)
```

Method sendPoll():

Usage:

```
TGBot$sendPoll(  
  question = NULL,  
  options = NULL,  
  is_anonymous = TRUE,  
  type = c("regular", "quiz"),  
  allows_multiple_answers = FALSE,  
  correct_option_id = NULL,  
  is_closed = FALSE,  
  disable_notification = NULL,  
  reply_to_message_id = NULL,  
  chat_id = NULL  
)
```

Method stopPoll():

Usage:

```
TGBot$stopPoll(message_id = NULL, chat_id = NULL)
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
TGBot$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

References

[Bots: An introduction for developers and Telegram Bot API](#)

Examples

```
## Not run:

## Without proxy
bot <- TGBot$new(token = bot_token('RBot'))

## With a proxy ...
prx <- list('url' = '123.45.6.78',
            'port' = 8080,
            'username' = 'user',
            'password' = 'password')
bot <- TGBot$new(token = bot_token('RBot'), proxy = prx)
## .. or
bot <- TGBot$new(token = bot_token('RBot'))
bot$set_proxy(proxy = prx)

## End(Not run)
```

user_id

user_id

Description

Obtain telegram user id from system variables (in Renviron) set according to package naming conventions, that is R_TELEGRAM_USER_X where is the user's name .

Usage

```
user_id(x)
```

Arguments

x character of length 1 with the name of the user; if NULL a menu to choose between bot is displayed and the proper value returned

Examples

```
## Not run: user_id('me')
```

Index

bot_token, [2](#)

forwardMessage, [2](#), [14](#)

getFile, [3](#), [14](#)

getMe, [3](#), [14](#)

getUpdates, [4](#), [14](#)

getUserProfilePhotos, [4](#), [14](#)

group_id, [5](#)

proxy, [5](#)

R6Class, [14](#)

sendAudio, [6](#), [14](#)

sendChatAction, [6](#)

sendDice, [7](#)

sendDocument, [7](#), [14](#)

sendLocation, [8](#), [14](#)

sendMessage, [9](#), [14](#)

sendPhoto, [9](#), [14](#)

sendPoll, [10](#), [14](#)

sendSticker, [11](#), [14](#)

sendVideo, [11](#), [14](#)

sendVoice, [12](#), [14](#)

stopPoll, [13](#), [14](#)

telegram, [13](#)

telegram-package (telegram), [13](#)

TGBot, [14](#)

user_id, [19](#)