Package: urlshorteneR (via r-universe)

October 11, 2024

Type Package

Title R Wrapper for the 'Bit.ly' and 'Is.gd'/'v.gd' URL Shortening Services

Description Allows using two URL shortening services, which also provide expanding and analytic functions. Specifically developed for 'Bit.ly' (which requires OAuth 2.0) and 'is.gd' (no API key).

Version 1.5.7

Date 2022-08-20

Maintainer John Malc <cincenko@outlook.com>

Encoding UTF-8

Depends R (>= 4.0)

Imports httr (>= 1.4.3), jsonlite (>= 1.8.0), stringr (>= 1.4.0), lubridate (>= 1.8.0), assertthat (>= 0.2.1), shiny (>= 1.7.2), clipr (>= 0.8.0), miniUI (>= 0.1.1.1), cli (>= 3.3.0)

Suggests roxygen2 (>= 7.2.1), knitr (>= 1.39), testthat (>= 3.1.4), rmarkdown (>= 2.14), httpuv (>= 1.6.5), stringi (>= 1.7.8), covr (>= 3.5.1), lintr (>= 3.0.0)

VignetteBuilder knitr

License Apache License 2.0

URL https://github.com/dmpe/urlshorteneR

BugReports https://github.com/dmpe/urlshorteneR/issues

RoxygenNote 7.2.1

NeedsCompilation no

Repository https://dmpe.r-universe.dev

RemoteUrl https://github.com/dmpe/urlshortener

RemoteRef HEAD

RemoteSha c9276e932e76c4f12b6a51bd6348b58399ee9442

Contents

`bitly_add_cust_bitlink` *Add Custom Bitlink (Premium)*

Description

Add a Keyword to a Bitlink

Usage

```
bitly_add_cust_bitlink(
  bitlink_id = NULL,
  custom_bitlink = NULL,
  showRequestURL = FALSE
)
```

Arguments

`bitlink_id`	- string
`custom_bitlink`	- A Custom Bitlink made of the domain and keyword
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Custom Bitlinks

Custom Bitlinks have both a branded short domain (BSD) AND a customized backend. For example, bit.ly/bitlinks would not be considered a Custom Bitlink because it does not have a branded short domain. es.pn/2yxklu would not be considered a custom Bitlink because while it has a branded short domain, it doesn't have a customized backhalf. An example of a link that would live in this section is es.pn/SuperBowl

See Also

https://dev.bitly.com/api-reference/#addCustomBitlink

Examples

```
## Not run:
bitly_add_cust_bitlink(custom_bitlink = "es.pn/SuperBowl", bitlink_id = "")

## End(Not run)
```

---

bitly_app_details *Retrieve OAuth App*

Description

Retrieve details for the provided OAuth App client ID

Usage

```
bitly_app_details(
  client_id = "be03aead58f23bc1aee6e1d7b7a1d99d62f0ede8",
  showRequestURL = F
)
```

Arguments

client_id	- The client ID of an OAuth app
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

---

bitly_auth *Assign bit.ly API tokens using OAuth2.0*

Description

There are 2 ways of how you can authenticate using this package.

1. The recommended practise for the end-user of this package is to use default API keys which are provided using this method.

2. Alternatively, you can register your own application via the web in order to get Client ID and Client Secret code.

For that go first to https://app.bitly.com/settings/integrations/. Click REGISTERED OAUTH APPLICATIONS, then REGISTER NEW APPLICATION followed by GET REGISTRATION CODE. Open your email that you will receive and click COMPLETE REGISTRATION. Make up an APPLICATION NAME that is unique. Unless you know to do otherwise, type "http://localhost:1410/" (slash at the end is important) in REDIRECT URIs. For APPLICATION LINK and APPLICATION DESCRIPTION you can type whatever you like.

Usage

```
bitly_auth(
  key = "be03aead58f23bc1aee6e1d7b7a1d99d62f0ede8",
  secret = "f9c6a3b18968e991e35f466e90c7d883cc176073",
  debug = F,
  token
)
```

Arguments

key	- Client ID
secret	- Client Secret
debug	- whether to print additional debug messages
token	- a Token object or a file path to an rds file containing a token.

However Important Information

Before choosing registering new application yourself, you can try using my API keys (the default option). No worries, no information is exposed to me at all: neither what you shorten nor who does it, etc. In fact, quote: "If you are shortening URLs on behalf of end-users, we ask that you use our OAuth 2 implementation to authenticate end-users before shortening. URLs shortened in this manner will be added to the specified end-user's history, allowing the end-user to manage and track the shortened URLs".

WARNING

If using RStudio in the browsers via RStudio Server, then authentication may not work well. In such case, use desktop RStudio application. Look for help at <https://support.rstudio.com/>.

See Also

See https://dev.bitly.com/api-reference

Examples

```
## Not run:
# overwrite keys - Variant 2
bitly_token <-
  bitly_auth(
    key = "be03aead58f23bc1aee6e1d7b7a1d99d62f0ede8",
    secret = "f9c6a3b18968e991e35f466e90c7d883cc176073"
  )

# default variant
bitly_token <- bitly_auth()
saveRDS(bitly_token, "bitly_token.rds")
# for non-interactive use:
bitly_auth(token = "bitly_token.rds")
```

```
## End(Not run)
```

`bitly_bsds` *Fetch all Branded Short Domains*

Description

BSDs is an acronym for branded short domains. This is a custom 15 character or less domain for bitlinks. This allows you to customize the domain to your brand.

Usage

```
bitly_bsds(showRequestURL = F)
```

Arguments

`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getBSDs

`bitly_bsds_overrides` *Branded Short Domains Group Overrides (Premium)*

Description

Retrieves all account overrides matching specified group_guid and bsd query filters.

Usage

```
bitly_bsds_overrides(group_id = NA, showRequestURL = F)
```

Arguments

`group_id`	- a required string | A GUID for a Bitly group
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getOverridesForGroups

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
bsds_over <- bitly_bsds_overrides(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_create_bitlink` *Create a short Bitlink*

Description

See https://dev.bitly.com/docs/getting-started/rate-limits and https://dev.bitly.com/api-reference/#createFullBitlink Convert a long url to a Bitlink and set additional parameters.

Usage

```
bitly_create_bitlink(
  long_url = NULL,
  domain = "bit.ly",
  title = NULL,
  tags = NULL,
  group_guid = NULL,
 deeplinks_list = list(app_uri_path = NULL, install_type = NULL, install_url = NULL,
    app_id = NULL),
  showRequestURL = FALSE
)
```

Arguments

`long_url`	- required, a long URL to be shortened (example: https://www.idnes.cz). Must contain http/https
`domain`	- (optional) the short domain to use; either bit.ly, j.mp, or bitly.com or a custom short domain. The default for this parameter is the short domain selected by each user in their bitly account settings. Passing a specific domain via this parameter will override the default settings.
`title`	- title of the bitlink
`tags`	- Array of string, use e.g. `c("test1", "test2")`
`group_guid`	- a GUID for a Bitly group
`deeplinks_list`	- a list containing parameters below
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.
`app_uri_path`	- app_uri_path

`install_type` - install_type

`install_url` - install_url

`app_id` - app_id

Value

id - a short bitly identifier for long_url which is unique to the given account.

long_url - This may not always be equal to the URL requested, as some URL normalization may occur (e.g., due to encoding differences, or case differences in the domain). This long_url will always be functionally identical the the request parameter.

link - an bitly id with http(s) prefix

Note

Look in the vignette for bulk shortening of URLs. Each call of this function == 1 API call. Take that into consideration due to limits etc.

The bitly API does not support shortening more than one long URL with a single API call. Meaning 1 Long URL = 1 Function call.

Long URLs should be URL-encoded. You can not include a longUrl in the request that has &, ?, #, or other reserved parameters without first encoding it.

The default value for the domain parameter is selected by each user from within their bitly account settings at <https://app.bitly.com/settings/api/>.

Long URLs should not contain spaces: any longUrl with spaces will be rejected. All spaces should be either percent encoded spaces are all indications of errors. Please remember to strip leading and trailing whitespace from any user input before shortening.

Examples

```
## Not run:
bitly_create_bitlink(long_url = "http://slovnik.seznam.cz/")

## End(Not run)
```

---

bitly_create_campaigns

Create Campaign (Premium)

Description

Create a new campaign

Usage

```
bitly_create_campaigns(
  group_guid = NULL,
  channel_guids = NULL,
  description = NULL,
  name = NULL,
  showRequestURL = T
)
```

Arguments

group_guid	- a GUID for a Bitly group
channel_guids	- a list of strings
description	- description of campaign
name	- its name
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#createCampaign

Examples

```
## Not run:
cc <- bitly_create_campaigns(
  group_guid = "testing", showRequestURL = TRUE, channel_guids = list("1", "2", "3"),
  description = "description", name = "name"
)

## End(Not run)
```

---

bitly_create_channel *Create channel (Premium)*

Description

Create a new channel

Usage

```
bitly_create_channel(
  group_guid = NULL,
  guid = NULL,
  name = NULL,
  modified = NULL,
  created = NULL,
  campaign_guid = NULL,
  bitlink_id = NULL,
  showRequestURL = T
)
```

Arguments

group_guid	- a GUID for a Bitly group
guid	- ID for a channel
name	- its name
modified	- string | ISO_TIMESTAMP
created	- string | ISO TIMESTAMP
campaign_guid	- string | A GUID for a Bitly campaign
bitlink_id	- string
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#createChannel

Examples

```
## Not run:
  gc <- bitly_create_channel(group_guid = "testing", ...)

## End(Not run)
```

bitly_expand_link *Expand a Bitlink*

Description

See https://dev.bitly.com/api-reference/#expandBitlink This endpoint returns public information for a Bitlink.

Usage

```
bitly_expand_link(bitlink_id = NULL, showRequestURL = FALSE)
```

Arguments

bitlink_id	- string
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Value

long_url - a full URL to which bitlink points to

Examples

```
## Not run:
bitly_expand_link(bitlink_id = "bit.ly/DPetrov")
bitly_expand_link(bitlink_id = "on.natgeo.com/1bEVhwE")

## manyHashes <- list("bit.ly/DPetrov", "bit.ly/1QU8CFm", "bit.ly/1R1LPSE", "bit.ly/1LNqqva")
## for (u in 1:length(manyHashes)) {
##   print(bitly_expand_link(bitlink_id = manyHashes[[u]], showRequestURL = TRUE))
## }

## End(Not run)
```

bitly_rate_limits *bitly_rate_limits*

Description

Provides bit.ly rate limits by endpoint. See https://dev.bitly.com/api-reference/#getPlatformLimitss

Usage

```
bitly_rate_limits(showRequestURL = F)
```

Arguments

showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Value

data.frame of end points and their rate limits by action

---

bitly_retrieve_bitlink *Retrieve a Bitlink*

Description

This endpoint returns information for a Bitlink.

Usage

```
bitly_retrieve_bitlink(bitlink = NULL, showRequestURL = FALSE)
```

Arguments

bitlink	- required, a Bitlink made of the domain and hash
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

See https://dev.bitly.com/api-reference/#getBitlink

Examples

```
## Not run:
bitly_retrieve_bitlink(bitlink = "cnn.it/2HomWGB")

## End(Not run)
```

`bitly_retrieve_bitlinks_by_groups` *Retrieve Bitlinks by Group*

Description

See https://dev.bitly.com/api-reference/#getBitlinksByGroup Retrieve a paginated collection of Bitlinks for a Group

Usage

```
bitly_retrieve_bitlinks_by_groups(
  group_guid = NULL,
  size = 50,
  page = 1,
  showRequestURL = FALSE,
  keyword = NULL,
  query_q = NULL,
  created_before = NULL,
  created_after = NULL,
  modified_after = NULL,
  archived = "both",
  deeplinks = "both",
  campaign_guid = NULL,
  channel_guid = NULL,
  custom_bitlink = "both",
  tags = NULL,
  encoding_login = NULL,
  domain_deeplinks = "both"
)
```

Arguments

`group_guid`	- a GUID for a Bitly group
`size`	- The quantity of items to be be returned
`page`	- Default: 1 | Integer specifying the numbered result at which to start
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.
`keyword`	- Custom keyword to filter on history entries
`query_q`	- a query to look for in bitlinks; acts a filter
`created_before`	- Timestamp as an integer unix epoch
`created_after`	- Timestamp as an integer unix epoch
`modified_after`	- Timestamp as an integer unix epoch, see `as_datetime` or `anytime`'s `as_datetime`

archived	- string | Default: "off" | Enum:"on" "off" "both" | Whether or not to include archived bitlinks
deeplinks	- string | Default: "both" | Enum:"on" "off" "both" | Filter to only Bitlinks that contain deeplinks
campaign_guid	- string | A GUID for a Bitly campaign
channel_guid	- Filter to return only links for the given channel GUID, can be provided, overrides all other parameters
custom_bitlink	- string | Default: "both" | Enum:"on" "off" "both"
tags	- Array of string, use e.g. `c("test1", "test2")`
encoding_login	- Array of string | Filter by the login of the authenticated user that created the Bitlink
domain_deeplinks	- string | Default: "both" | Enum:"on" "off" "both" | Filter to only Bitlinks that contain deeplinks configured with a custom domain

Examples

```
## Not run:
bitly_retrieve_bitlinks_by_groups(group_guid = "bit.ly/DPetrov", keyword = "novy titulek")

## End(Not run)
```

`bitly_retrieve_campaign` *Retrieve a Campaign*

Description

Retrive details for a campaign

Usage

```
bitly_retrieve_campaign(campaign_guid = NULL, showRequestURL = T)
```

Arguments

campaign_guid	- string | A GUID for a Bitly campaign
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#getCampaign

Examples

```
## Not run:
  gc <- bitly_retrieve_campaign(campaign_guid = "testing")

## End(Not run)
```

`bitly_retrieve_campaigns` *Retrieve campaigns (Premium)*

Description

Retrieve the campaigns for the current user

Usage

```
bitly_retrieve_campaigns(group_guid = NULL, showRequestURL = T)
```

Arguments

`group_guid`	- a GUID for a Bitly group
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#getCampaigns

Examples

```
## Not run:
  gc <- bitly_retrieve_campaigns(group_guid = "testing")

## End(Not run)
```

bitly_retrieve_channel

Get a Channel (Premium)

Description

Get a channel's details

Usage

```
bitly_retrieve_channel(channel_guid = NULL, showRequestURL = T)
```

Arguments

channel_guid	- GUID of a target channel
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#getChannel

Examples

```
## Not run:
  gc <- bitly_retrieve_channel(channel_guid = "testing")

## End(Not run)
```

bitly_retrieve_channels

Retrieve channels (Premium)

Description

Retrieve the channels available to a user

Usage

```
bitly_retrieve_channels(
  group_guid = NULL,
  campaign_guid = NULL,
  showRequestURL = T
)
```

Arguments

group_guid	- a GUID for a Bitly group
campaign_guid	- string	A GUID for a Bitly campaign
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#getChannels

Examples

```
## Not run:
  gc <- bitly_retrieve_channels(group_guid = "testing", campaign_guid = "test")

## End(Not run)
```

---

bitly_retrieve_clicks *Get Clicks for a Bitlink*

Description

See https://dev.bitly.com/api-reference/#getClicksForBitlink This will return the click counts for a specified Bitlink. This returns an array with clicks based on a date.

Usage

```
bitly_retrieve_clicks(
  bitlink = NULL,
  size = 50,
  unit_reference = NULL,
  unit = NULL,
  units = -1,
  showRequestURL = FALSE
)
```

Arguments

bitlink	- required, a Bitlink made of the domain and hash
size	- The quantity of items to be be returned
unit_reference	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
unit	- A unit of time
units	- An integer representing the time units to query data for. pass -1 to return all units of time.
showRequestURL	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
bitly_retrieve_clicks(bitlink = "cnn.it/2HomWGB", unit = "day", units = -1, size = 100)

## End(Not run)
```

---

bitly_retrieve_clicks_summary

Get Clicks Summary for a Bitlink

Description

See https://dev.bitly.com/api-reference/#getClicksSummaryForBitlink This will return the click counts for a specified Bitlink. This rolls up all the data into a single field of clicks.

Usage

```
bitly_retrieve_clicks_summary(
  bitlink = NULL,
  size = 50,
  unit_reference = NULL,
  unit = NULL,
  units = -1,
  showRequestURL = FALSE
)
```

Arguments

bitlink	- required, a Bitlink made of the domain and hash
size	- The quantity of items to be be returned
unit_reference	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
unit	- A unit of time

units - An integer representing the time units to query data for. pass -1 to return all units of time.

showRequestURL - show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
bitly_retrieve_clicks_summary(bitlink = "cnn.it/2HomWGB", unit = "day", units = -1, size = 100)

## End(Not run)
```

---

bitly_retrieve_cust_bitlink

Retrieve Custom Bitlink (Premium)

Description

Retrieve the details and history of a Custom Bitlink

Usage

```
bitly_retrieve_cust_bitlink(custom_bitlink = NULL, showRequestURL = FALSE)
```

Arguments

custom_bitlink - A Custom Bitlink made of the domain and keyword

showRequestURL - an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getCustomBitlink

Examples

```
## Not run:
bitly_retrieve_cust_bitlink(custom_bitlink = "es.pn/SuperBowl")

## End(Not run)
```

bitly_retrieve_cust_bitlink_clicks_history

Get Clicks for a Custom Bitlink's Entire History (Premium)

Description

Returns the click counts for the specified link. This returns an array with clicks based on a date.

Usage

```
bitly_retrieve_cust_bitlink_clicks_history(
  custom_bitlink = NULL,
  showRequestURL = FALSE
)
```

Arguments

custom_bitlink	- A Custom Bitlink made of the domain and keyword
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getClicksForCustomBitlink

Examples

```
## Not run:
bitly_retrieve_cust_bitlink_clicks_history(custom_bitlink = "es.pn/SuperBowl")

## End(Not run)
```

bitly_retrieve_cust_bitlink_metrics_destination

Get Metrics for a Custom Bitlink by Destination (Premium)

Description

Returns click metrics for the specified link by its historical destinations.

Usage

```
bitly_retrieve_cust_bitlink_metrics_destination(
  custom_bitlink = NULL,
  showRequestURL = FALSE
)
```

Arguments

custom_bitlink	- A Custom Bitlink made of the domain and keyword
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getCustomBitlinkMetricsByDestination

Examples

```
## Not run:
bitly_retrieve_cust_bitlink_metrics_destination(custom_bitlink = "es.pn/SuperBowl")

## End(Not run)
```

---

bitly_retrieve_destination_metrics

Get Metrics for a Custom Bitlink by destination (Premium)

Description

Get Click Metrics for a Custom Bitlink by historical Bitlink destinations

Usage

```
bitly_retrieve_destination_metrics(
  custom_bitlink = NULL,
  showRequestURL = FALSE
)
```

Arguments

custom_bitlink	- A Custom Bitlink made of the domain and keyword
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getCustomBitlinkMetricsByDestination

Examples

```
## Not run:
bitly_retrieve_destination_metrics(custom_bitlink = "es.pn/SuperBowl")

## End(Not run)
```

bitly_retrieve_group *Retrieve a single group*

Description

Retrive details for a specific group that a user belongs to.

Usage

```
bitly_retrieve_group(group_id = NA, showRequestURL = F)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroup

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group(group_guid = ui$default_group_guid)

## End(Not run)
```

bitly_retrieve_groups *Retrieve a list of all groups*

Description

Retrive details for all groups that a user belongs to.

Usage

```
bitly_retrieve_groups(organization_id = NULL, showRequestURL = F)
```

Arguments

organization_id
: - an optional string parameter | A GUID for a Bitly organization

showRequestURL
: - an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroups

Examples

```
## Not run:
rg <- bitly_retrieve_groups("") # will still work ok

## End(Not run)
```

---

bitly_retrieve_group_click_metrics_by_cities

Get Click Metrics for a specified group by city (Premium)

Description

Returns the geographic origins of click traffic by city for the specified group. Requires a premium account.

Usage

```
bitly_retrieve_group_click_metrics_by_cities(group_id = NA, showRequestURL = F)
```

Arguments

group_id
: - a required string | A GUID for a Bitly group

showRequestURL
: - an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference/#getGroupMetricsByCities

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group_click_metrics_by_cities(group_id = ui$default_group_guid[1])

## End(Not run)
```

---

bitly_retrieve_group_click_metrics_by_countries

Get Click Metrics for a Group by countries

Description

This endpoint will return metrics about the countries referring click traffic rolled up to a Group

Usage

```
bitly_retrieve_group_click_metrics_by_countries(
  group_id = NA,
  showRequestURL = F
)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroupMetricsByCountries

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group_click_metrics_by_countries(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_retrieve_group_click_metrics_by_devices` *Get Click Metrics for a specified group by devices (Premium)*

Description

Returns the device types generating click traffic to the specified group's links. Requires a premium account.

Usage

```
bitly_retrieve_group_click_metrics_by_devices(
  group_id = NA,
  showRequestURL = F
)
```

Arguments

`group_id`	- a required string | A GUID for a Bitly group
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference/#getGroupMetricsByDevices

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group_click_metrics_by_devices(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_retrieve_group_click_metrics_by_ref_networks` *Get Click Metrics for a Group by referring networks*

Description

This endpoint will return metrics about the referring network click traffic rolled up to a Group

Usage

```
bitly_retrieve_group_click_metrics_by_ref_networks(
  group_id = NA,
  showRequestURL = F
)
```

Arguments

`group_id`	- a required string | A GUID for a Bitly group
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#GetGroupMetricsByReferringNetworks

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group_click_metrics_by_ref_networks(group_id = ui$default_group_guid[1])

## End(Not run)
```

bitly_retrieve_group_pref
Retrieve Group Preferences

Description

Retrieve preferences for a specific group

Usage

```
bitly_retrieve_group_pref(group_id = NA, showRequestURL = F)
```

Arguments

group_id	- the group id the user belongs to
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroupPreferences

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
group_pref <- bitly_retrieve_group_pref(group_id = ui$default_group_guid[1])

## End(Not run)
```

bitly_retrieve_group_shorten_counts
Retrieve Group Shorten Counts

Description

Get all the shorten counts for a specific group

Usage

```
bitly_retrieve_group_shorten_counts(group_id = NA, showRequestURL = F)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroupShortenCounts

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_group_shorten_counts(group_id = ui$default_group_guid[1])

## End(Not run)
```

---

bitly_retrieve_links_grouped

Retrieve Bitlinks by Group

Description

Retrieve a paginated collection of Bitlinks for a Group

Usage

```
bitly_retrieve_links_grouped(
  group_id = NA,
  keyword = NULL,
  search_query = NULL,
  created_before = NULL,
  created_after = NULL,
  modified_after = NULL,
  archived = "off",
  deeplinks = "both",
  domain_deeplinks = "both",
  campaign_guid = NULL,
  channel_guid = NULL,
  custom_bitlink = "both",
  tags = NULL,
```

```
  encoding_login = NULL,
  page = 1,
  size = 50,
  showRequestURL = F
)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
keyword	- Custom keyword to filter on history entries
search_query	- string | the value that you would like to search
created_before	- Timestamp as an integer unix epoch
created_after	- Timestamp as an integer unix epoch
modified_after	- Timestamp as an integer unix epoch, see as_datetime or anytime's as_datetime
archived	- string | Default: "off" | Enum:"on" "off" "both" | Whether or not to include archived bitlinks
deeplinks	- string | Default: "both" | Enum:"on" "off" "both" | Filter to only Bitlinks that contain deeplinks
domain_deeplinks	- string | Default: "both" | Enum:"on" "off" "both" | Filter to only Bitlinks that contain deeplinks configured with a custom domain
campaign_guid	- Filter to return only links for the given campaign GUID, can be provided
channel_guid	- Filter to return only links for the given channel GUID, can be provided, overrides all other parameters
custom_bitlink	- string | Default: "both" | Enum:"on" "off" "both"
tags	- Array of string | filter by given tags
encoding_login	- Array of string | Filter by the login of the authenticated user that created the Bitlink
page	- Default: 1 | Integer specifying the numbered result at which to start
size	- string | Default: 50 | The quantity of items to be be returned
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getBitlinksByGroup

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_links_grouped(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_retrieve_metrics_by_countries` *Get Metrics for a Bitlink by countries*

Description

This endpoint will return metrics about the countries referring click traffic to a single Bitlink.

Usage

```
bitly_retrieve_metrics_by_countries(
  bitlink = NULL,
  size = 100,
  unit = NULL,
  unit_reference = NULL,
  units = -1,
  showRequestURL = FALSE
)
```

Arguments

`bitlink`	- required, a Bitlink made of the domain and hash
`size`	- The quantity of items to be be returned
`unit`	- A unit of time
`unit_reference`	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`units`	- An integer representing the time units to query data for. pass -1 to return all units of time.
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.

See Also

https://dev.bitly.com/api-reference/#getMetricsForBitlinkByCountries

Examples

```
## Not run:
bitly_retrieve_metrics_by_countries(bitlink = "bit.ly/DPetrov", unit = "day", units = -1,
size = 100)

## End(Not run)
```

`bitly_retrieve_metrics_by_referrers`
Get Metrics for a Bitlink by referrers

Description

This endpoint will return metrics about the referrers referring click traffic to a single Bitlink.

Usage

```
bitly_retrieve_metrics_by_referrers(
  bitlink = NULL,
  size = 100,
  unit = NULL,
  unit_reference = NULL,
  units = -1,
  showRequestURL = FALSE
)
```

Arguments

`bitlink`	- required, a Bitlink made of the domain and hash
`size`	- The quantity of items to be be returned
`unit`	- A unit of time
`unit_reference`	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`units`	- An integer representing the time units to query data for. pass -1 to return all units of time.
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.

See Also

https://dev.bitly.com/api-reference/#getMetricsForBitlinkByReferrers

Examples

```
## Not run:
bitly_retrieve_metrics_by_referrers(bitlink = "bit.ly/DPetrov", unit = "day",
units = -1, size = 100)

## End(Not run)
```

`bitly_retrieve_metrics_by_referrers_by_domain`
Get Metrics for a Bitlink by referrers by domain

Description

This endpoint will group referrers metrics about a single Bitlink.

Usage

```
bitly_retrieve_metrics_by_referrers_by_domain(
  bitlink = NULL,
  size = 50,
  unit_reference = NULL,
  unit = NULL,
  units = -1,
  showRequestURL = FALSE
)
```

Arguments

`bitlink`	- required, a Bitlink made of the domain and hash
`size`	- string | Default: 50 | The quantity of items to be be returned
`unit_reference`	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`unit`	- string | Default: "day", Enum: "minute" "hour" "day" "week" "month" | A unit of time
`units`	- integer | Default: -1 | An integer representing the time units to query data for. pass -1 to return all units of time.
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#getMetricsForBitlinkByReferrersByDomains

Examples

```
## Not run:
bitly_retrieve_metrics_by_referrers_by_domain(bitlink = "bit.ly/DPetrov", unit = "day",
units = -1, size = 100)

## End(Not run)
```

`bitly_retrieve_org` *Retrieve a single Organization*

Description

Retrive details for the specified organization.

Usage

```
bitly_retrieve_org(organization_id = NULL)
```

Arguments

`organization_id`
: - a required string | A GUID for a Bitly organization. You may also simply pass "" (double quotes), but this should be avoided at all costs.

Organizations

Organizations are part of our hierarchy. This is the top level where a group and user will belong.

See Also

https://dev.bitly.com/api-reference/#getOrganization

Examples

```
## Not run:
all_orgs <- bitly_retrieve_orgs()
ro <- bitly_retrieve_org(organization_id = all_orgs$guid)

## End(Not run)
```

bitly_retrieve_orgs *Retrieve all Organizations*

Description

Retrieve a list of organizations

Usage

```
bitly_retrieve_orgs()
```

Organizations

Organizations are part of our hierarchy. This is the top level where a group and user will belong.

See Also

https://dev.bitly.com/api-reference/#getOrganizations

Examples

```
## Not run:
ros <- bitly_retrieve_orgs()

## End(Not run)
```

bitly_retrieve_org_plan_limits *Get Plan Limits*

Description

Returns all plan limits and counts available for an organization.

Usage

```
bitly_retrieve_org_plan_limits(organization_id = NULL)
```

Arguments

`organization_id`
: - a required string | A GUID for a Bitly organization. You may also simply pass "" (double quotes), but this should be avoided at all costs.

Organizations

Organizations are part of our hierarchy. This is the top level where a group and user will belong.

See Also

https://dev.bitly.com/api-reference/#getPlanLimits

Examples

```
## Not run:
all_orgs <- bitly_retrieve_orgs()
plan <- bitly_retrieve_org_plan_limits(organization_id = all_orgs$guid)

## End(Not run)
```

---

bitly_retrieve_org_shorten_counts

Retrieve Organization Shorten Counts

Description

Retrieve all the shorten counts for a specific organization

Usage

```
bitly_retrieve_org_shorten_counts(organization_id = NULL)
```

Arguments

organization_id	- a required string | A GUID for a Bitly organization. You may also simply pass "" (double quotes), but this should be avoided at all costs.

Value

facet - Enum: "countries" "referrers" "referrers_by_domain" "referring_domains" "referring_networks" "shorten_counts"

Organizations

Organizations are part of our hierarchy. This is the top level where a group and user will belong.

See Also

https://dev.bitly.com/api-reference/#getOrganizationShortenCounts

Examples

```
## Not run:
all_orgs <- bitly_retrieve_orgs()
osc <- bitly_org_shorten_counts(organization_id = all_orgs$guid)
df_org_short_counts <- data.frame(osc, stringsAsFactors = FALSE)

## End(Not run)
```

`bitly_retrieve_sorted_bitlinks_by_groups`
Retrieve Sorted Bitlinks for Group

Description

See https://dev.bitly.com/api-reference/#getSortedBitlinks This will retrieve a paginated response for Bitlinks that are sorted for the Group

Usage

```
bitly_retrieve_sorted_bitlinks_by_groups(
  group_guid = NULL,
  unit = "day",
  units = -1,
  sort = "clicks",
  size = 50,
  unit_reference = NULL,
  showRequestURL = FALSE
)
```

Arguments

`group_guid`	- a GUID for a Bitly group
`unit`	- A unit of time
`units`	- An integer representing the time units to query data for. pass -1 to return all units of time.
`sort`	- required, Enum:"clicks" - The type of sorting that you would like to do
`size`	- The quantity of items to be be returned
`unit_reference`	- string	An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
bitly_retrieve_sorted_bitlinks_by_groups(group_guid = "", sort = "clicks")

## End(Not run)
```

`bitly_retrieve_sorted_links` *Retrieve Sorted Bitlinks for Group*

Description

This will retrieve a paginated response for Bitlinks that are sorted for the Group. This method returns a combined object which end-user (you) have to further process for your needs.

Usage

```
bitly_retrieve_sorted_links(
  group_id = NA,
  to_sort_by = "clicks",
  unit = "day",
  units = -1,
  unit_reference = NULL,
  size = 50,
  showRequestURL = F
)
```

Arguments

`group_id`	- a required string | A GUID for a Bitly group
`to_sort_by`	- a required string | Enum: "clicks" | The type of sorting that you would like to do
`unit`	- string | Default: "day", Enum: "minute" "hour" "day" "week" "month" | A unit of time
`units`	- integer | Default: -1 | An integer representing the time units to query data for. pass -1 to return all units of time.
`unit_reference`	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`size`	- string | Default: 50 | The quantity of items to be be returned
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getSortedBitlinks

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_sorted_links(group_id = ui$default_group_guid[1])

## End(Not run)
```

---

bitly_retrieve_tags *Retrieve Tags by Group*

Description

Retrieve the currently used tags for a group

Usage

```
bitly_retrieve_tags(group_id = NA, showRequestURL = F)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#getGroupTags

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
rg <- bitly_retrieve_tags(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_shorten_link` *Shorten a Link*

Description

See https://dev.bitly.com/api-reference/#createBitlink Convert a long url to a Bitlink

Usage

```
bitly_shorten_link(
  domain = "bit.ly",
  group_guid = NULL,
  long_url = NULL,
  showRequestURL = FALSE
)
```

Arguments

`domain`	- (optional) the short domain to use; either bit.ly, j.mp, or bitly.com or a custom short domain. The default for this parameter is the short domain selected by each user in their bitly account settings. Passing a specific domain via this parameter will override the default settings.
`group_guid`	- a GUID for a Bitly group
`long_url`	- required, a long URL to be shortened (example: https://www.idnes.cz). Must contain http/https
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
bitly_shorten_link(url = "http://www.seznam.cz/")
bitly_shorten_link(url = "http://www.seznam.cz/", showRequestURL = TRUE)

manyUrls <- list(
  "http://www.seznam.cz/", "http://www.seznamasdas.cz/",
  "http://www.seznam.cz/asadasd", "http://www.seznam.cz/adqwrewtregt"
)
for (u in 1:length(manyUrls)) {
  print(bitly_shorten_link(long_url = manyUrls[[u]], showRequestURL = TRUE))
}

## End(Not run)
```

`bitly_update_bitlink` *Update a Bitlink*

Description

See https://dev.bitly.com/api-reference/#updateBitlink Update fields in the Bitlink

Usage

```
bitly_update_bitlink(
  bitlink = NULL,
  archived = NULL,
  tags = NULL,
  showRequestURL = FALSE,
  created_at = NULL,
  title = NULL,
  created_by = NULL,
  long_url = NULL,
  client_id = NULL,
  custom_bitlinks = NULL,
  link = NULL,
  id = NULL,
 deeplinks = list(bitlink = NULL, install_url = NULL, created = NULL, modified = NULL,
   app_uri_path = NULL, install_type = NULL, app_guid = NULL, guid = NULL, os = NULL)
)
```

Arguments

`bitlink`	- required, a Bitlink made of the domain and hash
`archived`	- string | Default: "off" | Enum:"on" "off" "both" | Whether or not to include archived bitlinks
`tags`	- Array of string, use e.g. `c("test1", "test2")`
`showRequestURL`	- show URL which has been build and requested from server. For debug purposes.
`created_at`	- update created at parameter
`title`	- title of the bitlink
`created_by`	- update user
`long_url`	- required, a long URL to be shortened (example: https://www.idnes.cz). Must contain http/https
`client_id`	- The client ID of an OAuth app
`custom_bitlinks`	- update custom_bitlinks
`link`	- link
`id`	- id
`deeplinks`	- string | Default: "both" | Enum:"on" "off" "both" | Filter to only Bitlinks that contain deeplinks

Examples

```
## Not run:
bitly_update_bitlink(bitlink = "bit.ly/DPetrov", title = "novy titulek")

## hash is the one which is only returned. Dont use
bitly_update_bitlink(bitlink = "on.natgeo.com/1bEVhwE")

## manyHashes <- list("bit.ly/DPetrov", "bit.ly/1QU8CFm", "bit.ly/1R1LPSE", "bit.ly/1LNqqva")
## for (u in 1:length(manyHashes)) {
##   print(bitly_update_bitlink(bitlink = manyHashes[[u]],
##                 title = stri_rand_strings(1, 8, pattern = "[A-Za-z0-9]")))
## }

## End(Not run)
```

`bitly_update_campaign` *Update A Channel (Premium)*

Description

Update an existing Channel

Usage

```
bitly_update_campaign(
  campaign_guid = NULL,
  group_guid = NULL,
  channel_guids = NULL,
  description = NULL,
  name = NULL,
  showRequestURL = T
)
```

Arguments

`campaign_guid`	- string | A GUID for a Bitly campaign
`group_guid`	- a GUID for a Bitly group
`channel_guids`	- a list of strings
`description`	- description of campaign
`name`	- its name
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#updateCampaign

Examples

```
## Not run:
cc <- bitly_update_campaign(
  group_guid = "", channel_guids = list("1", "2", "3"),
  description = "description", name = "name"
)

## End(Not run)
```

---

bitly_update_channel *Update A Channel (Premium)*

Description

Update an existing Channel

Usage

```
bitly_update_channel(
  channel_guid = NULL,
  group_guid = NULL,
  guid = NULL,
  name = NULL,
  modified = NULL,
  created = NULL,
  campaign_guid = NULL,
  bitlink_id = NULL,
  showRequestURL = T
)
```

Arguments

channel_guid	- GUID of a target channel
group_guid	- a GUID for a Bitly group
guid	- ID for a channel
name	- its name
modified	- string | ISO_TIMESTAMP
created	- string | ISO TIMESTAMP
campaign_guid	- string | A GUID for a Bitly campaign
bitlink_id	- string
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Campaigns

Bitly Campaigns allows you to build and manage omnichannel campaigns.

See Also

https://dev.bitly.com/api-reference/#updateChannel

Examples

```
## Not run:
uc <- bitly_update_channel(channel_guid = "testing", group_guid = "", name = "name")

## End(Not run)
```

---

bitly_update_cust_bitlink

Update Custom Bitlink (Premium)

Description

Move a Keyword to a different Bitlink

Usage

```
bitly_update_cust_bitlink(
  custom_bitlink = NULL,
  bitlink_id = NULL,
  showRequestURL = FALSE
)
```

Arguments

custom_bitlink	- A Custom Bitlink made of the domain and keyword
bitlink_id	- string
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

See Also

https://dev.bitly.com/api-reference/#updateCustomBitlink

Examples

```
## Not run:
bitly_update_cust_bitlink(custom_bitlink = "es.pn/SuperBowl", bitlink_id = "")

## End(Not run)
```

`bitly_update_group` *Update a Group*

Description

Update the details of a group

Usage

```
bitly_update_group(
  group_id = NA,
  name = NA,
  organization_id = NA,
  showRequestURL = F
)
```

Arguments

`group_id`	- a required string | A GUID for a Bitly group
`name`	- username to change
`organization_id`	- an optional string parameter | A GUID for a Bitly organization
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#updateGroup

[bitly_update_user()]

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
up_group <- bitly_update_group(group_id = ui$default_group_guid[1], name = "New Group Name",
organization_id = "asd")

## End(Not run)
```

bitly_update_group_pref *Update Group Preferences*

Description

Update preferences for a specific group

Usage

```
bitly_update_group_pref(group_id = NA, domain_pref = NA, showRequestURL = F)
```

Arguments

group_id	- a required string | A GUID for a Bitly group
domain_pref	- string
showRequestURL	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Group

Groups are a subdivision within an organization. A user will belong to a group within an organization. Most actions on our API will be on behalf of a group. For example, when you shorten a link, it will be on behalf of a user and a group.

See Also

https://dev.bitly.com/api-reference#updateGroupPreferences

Examples

```
## Not run:
ui <- bitly_user_info(showRequestURL = TRUE)
group_pref <- bitly_update_group_pref(group_id = ui$default_group_guid[1])

## End(Not run)
```

`bitly_update_user` *Update your name and/or default group ID*

Description

This will overwrite your (display) username and/or group ID you belong to.

Usage

```
bitly_update_user(default_group_guid = NULL, name = "", showRequestURL = FALSE)
```

Arguments

`default_group_guid`	- group id to change, see NOTE
`name`	- username to change
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

User

User operations such as changing your name or fetching basic user information apply only to the authenticated user.

Note

Applies only to the authenticated user: Changing group/org ID is only permitted to premium users. Thus, if you are a "free" user and will try to change your default group id to something else, you will get an error. In that case, only changing display name is permitted.

See Also

https://dev.bitly.com/api-reference/#updateUser

Examples

```
## Not run:
# this applies only for "free" users
uu <- bitly_update_user(name = "Malc")

# if you are premium user, you can additionally adjust your group id
uug <- bitly_update_user(name = "Malc", default_group_guid = "TestGroupID")

## End(Not run)
```

`bitly_user_info` *Retrieve information for the current authenticated user*

Description

Retrieve information for the current authenticated user

Usage

```
bitly_user_info(showRequestURL = FALSE)
```

Arguments

`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Value

login - the specified bitly login or the login of the authenticated user

name - the user's full/display name

default_group_guid - a group to which user belongs

created - Timestamp for the moment the user signed up (uses `ymd_hms`)

is_active - whether a user profile is active

modified - Timestamp of the last modification to the user profile (uses `ymd_hms`)

is_sso_user - is Single-Sign-On enabled for the user (PREMIUM FEATURE)

is_2fa_enabled - is 2 Step verification enabled ?

email - user's emails

User

User operations such as changing your name or fetching basic user information apply only to the authenticated user.

See Also

https://dev.bitly.com/api-reference/#getUser

Examples

```
## Not run:
   ui <- bitly_user_info(showRequestURL = TRUE)

## End(Not run)
```

`bitly_user_metrics_referring_domains` *Get Metrics for a Bitlink by referring domains*

Description

This endpoint will rollup the click counts to a referrer about a single Bitlink.

Usage

```
bitly_user_metrics_referring_domains(
  bitlink = NULL,
  unit = "day",
  units = -1,
  size = 50,
  unit_reference = NULL,
  showRequestURL = FALSE
)
```

Arguments

`bitlink`	- required, a Bitlink made of the domain and hash
`unit`	- string | Default: "day", Enum: "minute" "hour" "day" "week" "month" | A unit of time
`units`	- integer | Default: -1 | An integer representing the time units to query data for. pass -1 to return all units of time.
`size`	- string | Default: 50 | The quantity of items to be be returned
`unit_reference`	- string | An ISO-8601 timestamp, indicating the most recent time for which to pull metrics. Will default to current time.
`showRequestURL`	- an optional T/F value to whether show URL which has been build and requested from server. For debug purposes, default FALSE.

Value

facet - string | One of "countries" "referrers" "referrers_by_domain" "referring_domains" "referring_networks" "shorten_counts"

See Also

See https://dev.bitly.com/api-reference/#getMetricsForBitlinkByReferringDomains

Examples

```
## Not run:
bitly_user_metrics_referring_domains(bitlink = "cnn.it/2HomWGB", unit = "month", units = -1,
size = 100)

## End(Not run)
```

`clipExpanderAddin` *Expand an URL from clipboard*

Description

Expand an URL from clipboard

Usage

```
clipExpanderAddin()
```

`clipShortenerAddin` *Shorten an URL from clipboard*

Description

Shorten an URL from clipboard

Usage

```
clipShortenerAddin()
```

`isgd_LinksExpand` *Expand a short URL to a longer one*

Description

See https://is.gd/apilookupreference.php

Usage

```
isgd_LinksExpand(shorturl = "", showRequestURL = FALSE)
```

Arguments

shorturl	- (optional) You can specify the shorturl parameter if you'd like to pick a shortened URL instead of having is.gd randomly generate one. These must be between 5 and 30 characters long and can only contain alphanumeric characters and underscores. Shortened URLs are case sensitive. Bear in mind that a desired short URL might already be taken (this is very often the case with common words) so if you're using this option be prepared to respond to an error and get an alternative choice from your app's user.
showRequestURL	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
isgd_LinksExpand(shorturl = "https://is.gd/4oIAXJ", showRequestURL = TRUE)

## End(Not run)
```

---

isgd_LinksShorten *Given a long URL, returns a short is.gd link.*

Description

See https://is.gd/apishorteningreference.php

Usage

```
isgd_LinksShorten(
  longUrl = "",
  logstats = "0",
  shorturl = NULL,
  showRequestURL = FALSE
)
```

Arguments

longUrl	- The url parameter is the address that you want to shorten.
logstats	- (optional) Adding the parameter logstats=1 turns on logging of detailed statistics when the shortened URL you create is accessed. This allows you to see how many times the link was accessed on a given day, what pages referred people to the link, what browser visitors were using etc. You can access these stats via the link preview page for your shortened URL (add a hyphen/dash to the end of the shortened URL to get to it). Creating links with statistics turned on has twice the "cost" towards our rate limit of other shortened links, so leave this parameter out of your API call if you don't require statistics on usage. See our usage limits page for more information on this https://is.gd/usagelimits.php.

shorturl - (optional) You can specify the shorturl parameter if you'd like to pick a shortened URL instead of having is.gd randomly generate one. These must be between 5 and 30 characters long and can only contain alphanumeric characters and underscores. Shortened URLs are case sensitive. Bear in mind that a desired short URL might already be taken (this is very often the case with common words) so if you're using this option be prepared to respond to an error and get an alternative choice from your app's user.

showRequestURL - show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
short_lin <- isgd_LinksShorten(longUrl = "https://novinky.cz/",showRequestURL = TRUE)

## End(Not run)
```

---

is_bitly_user_premium_holder

Check if authenticated user holds premium account

Description

Check if authenticated user holds premium account

Usage

```
is_bitly_user_premium_holder()
```

See Also

[bitly_user_info()]

---

shortenerAddin

Shorten an URL

Description

Call this function as an addin to prompt a url shortener. bit.ly accepts a custom domain, see bitly_shorten_link.

Usage

```
shortenerAddin()
```

vgd_LinksExpand *Expand a short URL to a longer one*

Description

See https://v.gd/apilookupreference.php

Usage

```
vgd_LinksExpand(shorturl = "", showRequestURL = FALSE)
```

Arguments

shorturl	- (optional) You can specify the shorturl parameter if you'd like to pick a shortened URL instead of having v.gd randomly generate one. These must be between 5 and 30 characters long and can only contain alphanumeric characters and underscores. Shortened URLs are case sensitive. Bear in mind that a desired short URL might already be taken (this is very often the case with common words) so if you're using this option be prepared to respond to an error and get an alternative choice from your app's user.
showRequestURL	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
isgd_LinksExpand(shorturl = "https://v.gd/4oIAXJ", showRequestURL = TRUE)

## End(Not run)
```

vgd_LinksShorten *Given a long URL, returns a short v.gd link.*

Description

See https://v.gd/apishorteningreference.php

Usage

```
vgd_LinksShorten(
  longUrl = "",
  logstats = "0",
  shorturl = NULL,
  showRequestURL = FALSE
)
```

Arguments

longUrl	- The url parameter is the address that you want to shorten.
logstats	- (optional) Adding the parameter logstats=1 turns on logging of detailed statistics when the shortened URL you create is accessed. This allows you to see how many times the link was accessed on a given day, what pages referred people to the link, what browser visitors were using etc. You can access these stats via the link preview page for your shortened URL (add a hyphen/dash to the end of the shortened URL to get to it). Creating links with statistics turned on has twice the "cost" towards our rate limit of other shortened links, so leave this parameter out of your API call if you don't require statistics on usage. See our usage limits page for more information on this https://v.gd/usagelimits.php.
shorturl	- (optional) You can specify the shorturl parameter if you'd like to pick a shortened URL instead of having v.gd randomly generate one. These must be between 5 and 30 characters long and can only contain alphanumeric characters and underscores. Shortened URLs are case sensitive. Bear in mind that a desired short URL might already be taken (this is very often the case with common words) so if you're using this option be prepared to respond to an error and get an alternative choice from your app's user.
showRequestURL	- show URL which has been build and requested from server. For debug purposes.

Examples

```
## Not run:
assd <- vgd_LinksShorten(longUrl = "https://novinky.cz/",showRequestURL = TRUE)

## End(Not run)
```

Index