

# Package ‘tumblr’

March 18, 2020

**Type** Package

**Title** Access to Tumblr V2 API

**Version** 1.2

**Date** 2020-03-15

**Author** Andrea Capozio <andreacapozio@gmail.com>

**Maintainer** Andrea Capozio <andreacapozio@gmail.com>

**Description** Provides an R-interface to the Tumblr web API (see Tumblr v2 API on <<https://www.tumblr.com/docs/en/api/v2>>). Tumblr is a microblogging platform and social networking website (<<https://www.tumblr.com>>).

**Depends** R (>= 3.6.0), httr (>= 1.4.1), RCurl (>= 1.98-1.1), stringr (>= 1.4.0), RJSONIO (>= 1.3-1.4)

**License** Artistic-2.0

**Collate** 'avatar.R' 'dashboard.R' 'def.postParams.R' 'follow.R' 'followers.R' 'http.connection.R' 'info.blog.R' 'like.post.R' 'likes.R' 'post.R' 'post.delete.R' 'post.edit.R' 'post.reblog.R' 'posts.R' 'posts.draft.R' 'posts.queue.R' 'posts.submission.R' 'tagged.R' 'unfollow.R' 'unlike.post.R' 'user.following.R' 'user.info.R' 'user.likes.R' 'compact.R' 'nonce.R' 'oauth.encode.R' 'oauth.encode1.R' 'sort.names.R'

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-03-18 15:20:05 UTC

## R topics documented:

avatar	2
dashboard	3
follow	5
followers	7
info.blog	8
like.post	10
likes	11

post . . . . .	12
post.delete . . . . .	15
post.edit . . . . .	17
post.reblog . . . . .	19
posts . . . . .	22
posts.draft . . . . .	24
posts.queue . . . . .	27
posts.submission . . . . .	29
tagged . . . . .	31
unfollow . . . . .	33
unlike.post . . . . .	34
user.following . . . . .	35
user.info . . . . .	37
user.likes . . . . .	39

<b>Index</b>	<b>41</b>
--------------	-----------

---

avatar	<i>Retrieve a Blog Avatar.</i>
--------	--------------------------------

---

## Description

Get the url of a blog's avatar.

## Usage

```
avatar(base_hostname = NA, size = 64)
```

## Arguments

base_hostname	The standard or custom blog hostname. See Details.
size	The size of the avatar (square, one value for both length and width). Must be one of the values: 16, 24, 30, 40, 48, 64, 96, 128, 512.

## Details

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

## Value

If the download succeeded, the url of the blog's avatar has returned, otherwise an error is encountered.

## Author(s)

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## you must specify a real blog for base_hostname

size <- 48
base_hostname <- "blogname.tumblr.com"

avatar(base_hostname = base_hostname, size = 48)

## End(Not run)
```

---

dashboard	<i>Retrieve a User's Dashboard.</i>
-----------	-------------------------------------

---

**Description**

Use this method to retrieve the dashboard that matches the OAuth credentials submitted with the request.

**Usage**

```
dashboard(limit = 20, offset = 0, type = NA, since_id = 0, reblog_info = FALSE,
notes_info = FALSE, token = NA, consumer_key = NA, consumer_secret = NA)
```

**Arguments**

<code>limit</code>	The number of results to return: 1-20, inclusive.
<code>offset</code>	Post number to start at. 0 is the first post.
<code>type</code>	The type of post to return. The available values are: text, photo, quote, link, chat, audio, video, answer. If no values are specified, all types are returned.
<code>since_id</code>	Return posts that have appeared after this ID.
<code>reblog_info</code>	Indicates whether to return reblog information (specify TRUE or FALSE). Returns the various reblogged_fields.
<code>notes_info</code>	Indicates whether to return notes information (specify TRUE or FALSE). Returns note count and note metadata.
<code>token</code>	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
<code>consumer_key</code>	The consumer key provided by your application.
<code>consumer_secret</code>	The consumer secret provided by your application.

**Details**

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

A serialized JSON object with the following fields:

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The unique ID of the post.
post_url	A string. The location of the post.
type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.
source_title	A string. The title of the source site. Exists only if there is a content source.
liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2#common-fields>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret and appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/aces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

dashboard(limit = 15, offset = 3, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

follow

*Follow a blog.*

---

**Description**

This function allows to follow a blog of other Tumblr users.

**Usage**

```
follow(url = NA, token = NA, consumer_key = NA, consumer_secret = NA)
```

**Arguments**

<code>url</code>	The URL of the blog to follow.
<code>token</code>	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
<code>consumer_key</code>	The consumer key provided by your application.
<code>consumer_secret</code>	The consumer secret provided by your application.

## Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

## Value

Returns 200: OK (blog successfully followed) or a 404 (blog was not found).

## Author(s)

Andrea Capozio

## References

<https://www.tumblr.com/docs/en/api/v2>

## Examples

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/aces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for url

url <- "blogname.tumblr.com"

follow(url = url, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

 followers

 Retrieve a Blog's Followers
 

---

### Description

Retrieve the followers of the user's blog.

### Usage

```
followers(base_hostname = NA, limit = 20, offset = 0, token = NA,
consumer_key = NA, consumer_secret = NA)
```

### Arguments

base_hostname	The standard or custom blog hostname. See Details.
limit	The number of results to return: 1-20, inclusive.
offset	Result to start at. 0 is the first follower.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry. The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

### Value

A list object with the following fields:

total_users	A number. The number of users currently following the blog.
users	An array. Each item is a follower, containing these fields:
name	A string. The user's name on tumblr.
following	A boolean. Whether the caller is following the user.
url	A string. The URL of the user's primary blog.
updated	A number. The time of the user's most recent post, in seconds since the epoch.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps
consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

followers(base_hostname = base_hostname, limit = 20, offset = 0, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

info.blog

*Retrieve Blog's Info.*

---

**Description**

Returns general information about the blog, such as the title, number of posts, and other high-level data.

**Usage**

```
info.blog(base_hostname = NA, api_key = NA)
```



**Arguments**

base\_hostname The standard or custom blog hostname. See Details.  
 api\_key Your OAuth Consumer Key.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry. The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

**Value**

A list object with the following fields:

title A string. The display title of the blog.  
 posts A number. The total number of posts to this blog.  
 name A string. The short blog name that appears before tumblr.com in a standard blog hostname (and before the domain in a custom blog hostname).  
 updated A number. The time of the most recent post, in seconds since the epoch.  
 description A string. The description of the blog.  
 ask A boolean. Indicates whether the blog allows questions.  
 ask\_anon A boolean. Indicates whether the blog allows anonymous questions. Returned only if ask is TRUE.  
 likes A number. Number of likes for this user. Returned only if this is the primary blog of the user and sharing of likes is enabled.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request,
## where api_key is fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

api_key <- "key"

## you must specify a real blog for base_hostname
```

```
base_hostname <- "blogname.tumblr.com"

info.blog(base_hostname = base_hostname, api_key = api_key)

## End(Not run)
```

---

like.post

*Like a Post.*

---

## Description

This function allows to like a post of other Tumblr users.

## Usage

```
like.post(id = NA, reblog_key = NA, token = NA, consumer_key = NA,
consumer_secret = NA)
```

## Arguments

id	The ID of the post to like.
reblog_key	The reblog key for the post id.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

## Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

## Value

Returns 200: OK (post successfully liked) or a 404 (post id or reblog\_key was not found).

## Author(s)

Andrea Capozio

## References

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/acces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

id <- 7504154594
reblog_key <- "HNvqLd5G"

like.post(id = id, reblog_key = reblog_key, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

likes

*Retrieve Blog's Likes*


---

**Description**

Retrieve the publicly exposed likes from a blog.

**Usage**

```
likes(base_hostname = NA, limit = 20, offset = 0, api_key = NA)
```

**Arguments**

base_hostname	The standard or custom blog hostname. See Details.
limit	The number of results to return: 1-20, inclusive.
offset	Liked post number to start at. 0 is the first post.
api_key	Your OAuth Consumer Key. See Details.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry. The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

**Value**

A list object with the following fields:

liked\_posts      An Array. An array of post objects (posts liked by the user).  
 liked\_count      A number. Total number of liked posts.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request,
## where api_key is fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps
api_key <- "key"

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

likes(base_hostname = base_hostname, limit = 20, offset = 0, api_key = api_key)

## End(Not run)
```

---

post

*Create a New Blog Post*

---

**Description**

This function allows to create a blog post.

**Usage**

```
post(base_hostname = NA, type = "text", state = "published", tags = NA,
     tweet = NA, date = as.character(Sys.time()), format = "html", slug = NA,
     title_text = NA, body = NA, caption_photo = NA, link = NA, source_photo = NA,
     data_photo = NA, quote = NA, source_quote = NA, url_link = NA, title_link = NA,
     description = NA, title_chat = NA, conversation = NA, external_url = NA,
     data_audio = NA, caption_audio = NA, embed = NA, data_video = NA,
     caption_video = NA, token = NA, consumer_key = NA, consumer_secret = NA)
```

**Arguments**

All Post types have the following parameters:

base_hostname	The standard or custom blog hostname. See Details.
type	The type of post to create. Specify one of the following: text, photo, quote, link, chat, audio, video. The default is setted as text.
state	The state of the post. Specify one of the following: published, draft, queue, private. The default is setted as published.
tags	Comma-separated tags for this post.
tweet	Manages the autotweet (if enabled) for this post: set to off for no tweet, or enter text to override the default tweet.
date	The GMT date and time of the post, as a string.
format	Sets the format type of post. Supported formats are: html, markdown. The default is setted as html.
slug	Add a short text summary to the end of the post URL.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

**Text Posts**

title_text	The optional title of the post, HTML entities must be escaped.
body	The full post body, HTML allowed.

**Photo Posts**

caption_photo	The user-supplied caption, HTML allowed.
link	The "click-through URL" for the photo.
source_photo	The photo source URL. (Either source_photo or data_photo)
data_photo	One or more image files (submit multiple times to create a slide show). (Either source_photo or data_photo)

**Quote Posts**

quote	The full text of the quote, HTML entities must be escaped.
source_quote	Cited source, HTML allowed.

**Link Posts**

<code>url_link</code>	The link.
<code>title_link</code>	The title of the page the link points to, HTML entities should be escaped.
<code>description</code>	A user-supplied description, HTML allowed.
<b>Chat Posts</b>	
<code>title_chat</code>	The title of the chat.
<code>conversation</code>	The text of the conversation/chat, with dialogue labels (no HTML).
<b>Audio Posts</b>	
<code>external_url</code>	The URL of the site that hosts the audio file (not tumblr). (Either <code>external_url</code> or <code>data_audio</code> )
<code>data_audio</code>	An audio file. (Either <code>external_url</code> or <code>data_audio</code> )
<code>caption_audio</code>	The user-supplied caption.
<b>Video Posts</b>	
<code>embed</code>	HTML embed code for the video. (Either <code>embed</code> or <code>data_video</code> )
<code>data_video</code>	A video file. (Either <code>embed</code> or <code>data_video</code> )
<code>caption_video</code>	The user-supplied caption.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + `.tumblr.com`. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your `api_key`. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

Returns 201: Created or an error code.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps
```

```
consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/aces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

post(base_hostname = base_hostname, type = "text", tags = "tumblr, api", body = "foo",
  token = token, consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

post.delete

*Delete a Post.*

---

## Description

This function allows to delete a post.

## Usage

```
post.delete(base_hostname = NA, id = NA, token = NA, consumer_key = NA,
  consumer_secret = NA)
```

## Arguments

base_hostname	The standard or custom blog hostname. See Details.
id	The ID of the post to delete.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your `api_key`. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

Returns 200: OK (successfully deleted) or an error code.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"
id <- 7504154594

post.delete(base_hostname = base_hostname, id = id, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```



---

 post.edit

*Edit a Blog Post.*


---

## Description

This function allows to editing a blog post.

## Usage

```
post.edit(base_hostname = NA, type = "text", state = "published", tags = NA,
  tweet = NA, date = as.character(Sys.time()), format = "html", slug = NA,
  title_text = NA, body = NA, caption_photo = NA, link = NA, source_photo = NA,
  data_photo = NA, quote = NA, source_quote = NA, url_link = NA, title_link = NA,
  description = NA, title_chat = NA, conversation = NA, external_url = NA,
  data_audio = NA, caption_audio = NA, embed = NA, data_video = NA,
  caption_video = NA, id = NA, token = NA, consumer_key = NA,
  consumer_secret = NA)
```

## Arguments

All Post types have the following parameters:

base_hostname	The standard or custom blog hostname. See Details.
type	The type of post to create. Specify one of the following: text, photo, quote, link, chat, audio, video. The default is setted as text.
state	The state of the post. Specify one of the following: published, draft, queue, private. The default is setted as published.
tags	Comma-separated tags for this post.
tweet	Manages the autotweet (if enabled) for this post: set to off for no tweet, or enter text to override the default tweet.
date	The GMT date and time of the post, as a string.
format	Sets the format type of post. Supported formats are: html, markdown. The default is setted as html.
slug	Add a short text summary to the end of the post URL.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.
<b>Text Posts</b>	
title_text	The optional title of the post, HTML entities must be escaped.
body	The full post body, HTML allowed.

### Photo Posts

caption_photo	The user-supplied caption, HTML allowed.
link	The "click-through URL" for the photo.
source_photo	The photo source URL. (Either source_photo or data_photo)
data_photo	One or more image files (submit multiple times to create a slide show). (Either source_photo or data_photo)
<b>Quote Posts</b>	
quote	The full text of the quote, HTML entities must be escaped.
source_quote	Cited source, HTML allowed.
<b>Link Posts</b>	
url_link	The link.
title_link	The title of the page the link points to, HTML entities should be escaped.
description	A user-supplied description, HTML allowed.
<b>Chat Posts</b>	
title_chat	The title of the chat.
conversation	The text of the conversation/chat, with dialogue labels (no HTML).
<b>Audio Posts</b>	
external_url	The URL of the site that hosts the audio file (not tumblr). (Either external_url or data_audio)
data_audio	An audio file. (Either external_url or data_audio)
caption_audio	The user-supplied caption.
<b>Video Posts</b>	
embed	HTML embed code for the video. (Either embed or data_video)
data_video	A video file. (Either embed or data_video)
caption_video	The user-supplied caption.
<b>Editing Parameter</b>	
id	The ID of the post to edit.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

Returns 200: OK (successfully edited) or an error code.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/accessToken'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"
id <- 97468713814

post.edit(base_hostname = base_hostname, type = "text", tags = "tumblr, api",
  title_text = "Title", body = "foo 2", id = id, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

post.reblog

*Reblog a Post.*

---

**Description**

This function allows to reblog a blog post.

**Usage**

```
post.reblog(base_hostname = NA, type = "text", state = "published", tags = NA,
  tweet = NA, date = as.character(Sys.time()), format = "html", slug = NA,
  title_text = NA, body = NA, caption_photo = NA, link = NA, source_photo = NA,
  data_photo = NA, quote = NA, source_quote = NA, url_link = NA, title_link = NA,
  description = NA, title_chat = NA, conversation = NA, external_url = NA,
  data_audio = NA, caption_audio = NA, embed = NA, data_video = NA,
  caption_video = NA, id = NA, reblog_key = NA, comment = NA, token = NA,
  consumer_key = NA, consumer_secret = NA)
```

**Arguments**

All Post types have the following parameters:

base_hostname	The standard or custom blog hostname. See Details.
type	The type of post to create. Specify one of the following: text, photo, quote, link, chat, audio, video. The default is setted as text.
state	The state of the post. Specify one of the following: published, draft, queue, private. The default is setted as published.
tags	Comma-separated tags for this post.
tweet	Manages the autotweet (if enabled) for this post: set to off for no tweet, or enter text to override the default tweet.
date	The GMT date and time of the post, as a string.
format	Sets the format type of post. Supported formats are: html, markdown. The default is setted as html.
slug	Add a short text summary to the end of the post URL.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

**Text Posts**

title_text	The optional title of the post, HTML entities must be escaped.
body	The full post body, HTML allowed.

**Photo Posts**

caption_photo	The user-supplied caption, HTML allowed.
link	The "click-through URL" for the photo.
source_photo	The photo source URL. (Either source_photo or data_photo)
data_photo	One or more image files (submit multiple times to create a slide show). (Either source_photo or data_photo)

**Quote Posts**

quote	The full text of the quote, HTML entities must be escaped.
-------	------------------------------------------------------------

source_quote	Cited source, HTML allowed.
<b>Link Posts</b>	
url_link	The link.
title_link	The title of the page the link points to, HTML entities should be escaped.
description	A user-supplied description, HTML allowed.
<b>Chat Posts</b>	
title_chat	The title of the chat.
conversation	The text of the conversation/chat, with dialogue labels (no HTML).
<b>Audio Posts</b>	
external_url	The URL of the site that hosts the audio file (not tumblr). (Either external_url or data_audio)
data_audio	An audio file. (Either external_url or data_audio)
caption_audio	The user-supplied caption.
<b>Video Posts</b>	
embed	HTML embed code for the video. (Either embed or data_video)
data_video	A video file. (Either embed or data_video)
caption_video	The user-supplied caption.
<b>Reblogging Parameters</b>	
id	The ID of the reblogged post on tumblelog.
reblog_key	The reblog key for the reblogged post - get the reblog key with a /posts request.
comment	A comment added to the reblogged post.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

Returns 201: Created or an error code.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/accessToken'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"
id <- 97468713814

post.reblog(base_hostname = base_hostname, type = "text", tags = "tumblr, api",
  title_text = "Title", body = "foo 2", id = id, reblog_key="2F0Pxe0a",
  token = token, consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

posts

*Retrieve Published Posts.***Description**

This function retrieves published posts.

**Usage**

```
posts(base_hostname = NA, limit = 20, offset = 0, api_key = NA, type = NA,
  id = NA, tag = NA, reblog_info = FALSE, notes_info = FALSE, filter = "HTML")
```

**Arguments**

base_hostname	The standard or custom blog hostname. See Details.
limit	The number of results to return: 1-20, inclusive.
offset	Result to start at. 0 is the first follower.
api_key	The consumer secret provided by your application. See Details.

type	The type of post to return. The available values are: text, photo, quote, link, chat, audio, video, answer. If no values are specified, all types are returned.
id	A specific post ID. Returns the single post specified or (if not found) a 404 error.
tag	Limits the response to posts with the specified tag.
reblog_info	Indicates whether to return reblog information (specify TRUE or FALSE). Returns the various reblogged_fields. See Details.
notes_info	Indicates whether to return notes information (specify TRUE or FALSE). Returns note count and note metadata. See Details.
filter	Specifies the post format to return, other than HTML: text - Plain text, no HTML; raw - As entered by the user (no post-processing); if the user writes in Markdown, the Markdown will be returned rather than HTML.

### Details

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

reblog\_info and notes\_info are false by default because of the server impact involved in retrieving the data.

### Value

Each response includes a blog object that is equivalent of an info.blog response. Posts are returned as an array attached to the posts field. All post types have the following common response.

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The post's unique ID.
post_url	A string. The location of the post.
type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.

source_title	A string. The title of the source site. Exists only if there is a content source.
liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

For a specified type, other response fields are returned. See References for more details.

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

### Examples

```
## Not run:
## An example of an authenticated request,
## where api_key is fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps
api_key <- "key"

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

posts(base_hostname = base_hostname, type = "text", api_key = api_key)

## End(Not run)
```

---

posts.draft

*Retrieve Draft Posts.*

---

### Description

This function retrieves draft posts.

### Usage

```
posts.draft(base_hostname = NA, before_id = 0, filter = "HTML", token = NA,
consumer_key = NA, consumer_secret = NA)
```



**Arguments**

base_hostname	The standard or custom blog hostname. See Details.
before_id	Returns posts that have appeared after this ID.
filter	Specifies the post format to return, other than HTML: text - Plain text, no HTML; raw - As entered by the user (no post-processing); if the user writes in Markdown, the Markdown will be returned rather than HTML.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

**Details**

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

All post types have the following common response.

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The post's unique ID.
post_url	A string. The location of the post.
type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.
source_title	A string. The title of the source site. Exists only if there is a content source.

liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

For a specified type, other response fields are returned. See References for more details.

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

### Examples

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

posts.draft(base_hostname = base_hostname, filter = "HTML", token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

posts.queue	<i>Retrieve Queued Posts.</i>
-------------	-------------------------------

---

## Description

This function retrieves queued drafts.

## Usage

```
posts.queue(base_hostname = NA, limit = 20, offset = 0, filter = "HTML",
            token = NA, consumer_key = NA, consumer_secret = NA)
```

## Arguments

base_hostname	The standard or custom blog hostname. See Details.
limit	The number of results to return: 1-20, inclusive.
offset	Result to start at. 0 is the first follower.
filter	Specifies the post format to return, other than HTML: text - Plain text, no HTML; raw - As entered by the user (no post-processing); if the user writes in Markdown, the Markdown will be returned rather than HTML.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

## Details

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

## Value

All post types have the following common response.

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The post's unique ID.
post_url	A string. The location of the post.

type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.
source_title	A string. The title of the source site. Exists only if there is a content source.
liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

For a specified type, other response fields are returned. See References for more details.

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

### Examples

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
```

```

sig <- sign_oauth1.0(app,
token = token$credentials$oauth_token,
token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- "blogname.tumblr.com"

posts.queue(base_hostname = base_hostname, filter = "raw", token = token,
consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)

```

---

posts.submission	<i>Retrieve Submission Posts.</i>
------------------	-----------------------------------

---

### Description

This function retrieves submission posts.

### Usage

```
posts.submission(base_hostname = NA, offset = 0, filter = "HTML", token = NA,
consumer_key = NA, consumer_secret = NA)
```

### Arguments

base_hostname	The standard or custom blog hostname. See Details.
offset	Result to start at. 0 is the first follower.
filter	Specifies the post format to return, other than HTML: text - Plain text, no HTML; raw - As entered by the user (no post-processing); if the user writes in Markdown, the Markdown will be returned rather than HTML.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

Each blog has a unique hostname. The hostname can be standard or custom. Standard hostname: the blog short name + .tumblr.com. Custom hostname: Anything at all, as determined by a DNS CNAME entry.

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your api\_key. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

All post types have the following common response.

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The post's unique ID.
post_url	A string. The location of the post.
type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.
source_title	A string. The title of the source site. Exists only if there is a content source.
liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

For a specified type, other response fields are returned. See References for more details.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
```

```

appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/acces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for base_hostname
base_hostname <- base_hostname

posts.submission(base_hostname = base_hostname, filter = "HTML", token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)

```

tagged

*Get Posts with Tag.***Description**

Retrieve the posts with a set of tags.

**Usage**

```
tagged(api_key = NA, tag = NA, before = as.integer(Sys.time()),
  limit = 20, filter = "HTML")
```

**Arguments**

<code>api_key</code>	Your OAuth Consumer Key. See Details.
<code>tag</code>	The tag on the posts you'd like to retrieve.
<code>before</code>	The timestamp of when you'd like to see posts before. Current timestamp is the default.
<code>limit</code>	The number of results to return: 1-20, inclusive.
<code>filter</code>	Specifies the post format to return, other than HTML: text - plain text, no HTML; raw - as entered by the user(no post-processing); if the user writes in Markdown, the Markdown will be returned rather than HTML.

**Details**

The API uses three different levels of authentication, depending on the method. None: No authentication. Anybody can query the method. API key: Requires an API key. Use your OAuth Consumer Key as your `api_key`. OAuth: Requires a signed request that meets the OAuth 1.0a Protocol.

**Value**

A list object with the following fields:

blog_name	A string. The short name used to uniquely identify a blog.
id	A number. The unique ID of the post.
post_url	A string. The location of the post.
type	A string. The type of post.
timestamp	A number. The time of the post, in seconds since the epoch.
date	A string. The GMT date and time of the post, as a string.
format	A string. The post format: html or markdown.
reblog_key	A string. The key used to reblog this post.
tags	An array (string). Tags applied to the post.
bookmarklet	A boolean. Indicates whether the post was created via the Tumblr bookmarklet. Exists only if true.
mobile	A boolean. Indicates whether the post was created via mobile/email publishing. Exists only if true.
source_url	A string. The URL for the source of the content for quotes, reblogs, etc.. Exists only if there is a content source.
source_title	A string. The title of the source site. Exists only if there is a content source.
liked	A boolean. Indicates if a user has already liked a post or not. Exists only if the request is fully authenticated with OAuth.
state	A string. Indicates the current state of the post. States are: published, queued, draft and private.
total_posts	A number. The total number of post available for this request, useful for paginating through results.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## An example of an authenticated request,
## where api_key is fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps
api_key <- "key"

tag <- "api"

tagged(api_key = api_key, tag = tag)
```



---

unfollow	<i>Unfollow a blog.</i>
----------	-------------------------

---

### Description

This function allows to unfollow a blog of other Tumblr users.

### Usage

```
unfollow(url = NA, token = NA, consumer_key = NA, consumer_secret = NA)
```

### Arguments

url	The URL of the blog to unfollow.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

### Value

Returns 200: OK (blog successfully unfollowed) or a 404 (blog was not found).

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

### Examples

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
```

```

accessTokenURL <- 'https://www.tumblr.com/oauth/acces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

## you must specify a real blog for url
url <- "blogname.tumblr.com"

unfollow(url = url, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)

```

---

 unlike.post

*Unlike a Post.*


---

### Description

This function allows to unlike a post of other Tumblr users.

### Usage

```
unlike.post(id = NA, reblog_key = NA, token = NA, consumer_key = NA,
  consumer_secret = NA)
```

### Arguments

id	The ID of the post to unlike.
reblog_key	The reblog key for the post id.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

### Value

Returns 200: OK (post successfully unliked) or a 404 (post id or reblog\_key was not found).

**Author(s)**

Andrea Capozio

**References**<https://www.tumblr.com/docs/en/api/v2>**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/aces_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

id <- 7504154594
reblog_key <- "HNvqLd5G"

unlike.post(id = id, reblog_key = reblog_key, token = token,
  consumer_key = consumer_key, consumer_secret = consumer_secret)

## End(Not run)
```

---

`user.following`*Retrieve the blogs a user is following.*

---

**Description**

Use this method to retrieve the blogs followed by the user whose OAuth credentials are submitted with the request.

**Usage**

```
user.following(limit = 20, offset = 0, token = NA, consumer_key = NA,
  consumer_secret = NA)
```

**Arguments**

limit	The number of results to return: 1-20, inclusive.
offset	Result to start at. 0 is the first follower.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

**Details**

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

**Value**

A serialized JSON object with the following fields:

total_blogs	A number. The number of blogs the user is following.
blogs	An array. Each item is a blog that is being followed, containing these fields:
name	A string. The user name attached the blog that is being followed.
url	A string. The URL of the blog that is being followed.
updated	A number. The time of the most recent post, in seconds since the epoch.
title	A string. The title of the blog.
description	A string. The description of the blog.

**Author(s)**

Andrea Capozio

**References**

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'
```

```

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
token = token$credentials$oauth_token,
token_secret = token$credentials$oauth_token_secret)

user_following(token = token, consumer_key = consumer_key,
consumer_secret = consumer_secret)

## End(Not run)

```

---

user.info

*Get a User's Information.*


---

### Description

Use this method to retrieve the user's account information that matches the OAuth credentials submitted with the request.

### Usage

```
user.info(token = NA, consumer_key = NA, consumer_secret = NA)
```

### Arguments

token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

### Value

A list object with the following fields:

following	A number. The number of blogs the user is following.
default_post_format	A string. The default posting format -html, markdown.
name	A string. The user's tumblr short name.
likes	A number. The total count of the user's like.

blogs	An array. Each item is a blog that is being followed, containing these fields:
name	A string. The short name of the blog.
url	A string. The URL of the blog.
primary	A boolean. Indicates if this is the user's primary blog.
title	A string. The title of the blog.
followers	A number. Total count of followers for this blog.
tweet	A number. Indicates if posts are tweeted (auto, Y, N).
facebook	Indicates if posts are sent to Facebook Y, N.
type	Indicates whether a blog is public or private.

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

### Examples

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/access_token'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

user.info(token = token, consumer_key = consumer_key,
  consumer_secret = consumer_secret)

## End(Not run)
```

---

user.likes	<i>Retrieve a User's Likes.</i>
------------	---------------------------------

---

### Description

Use this method to retrieve the liked posts that match the OAuth credentials submitted with the request.

### Usage

```
user.likes(limit = 20, offset = 0, token = NA, consumer_key = NA,  
consumer_secret = NA)
```

### Arguments

limit	The number of results to return: 1-20, inclusive.
offset	Result to start at. 0 is the first follower.
token	Represents the complete set of data needed for OAuth access: an app, an endpoint, cached credentials and parameters. See Details.
consumer_key	The consumer key provided by your application.
consumer_secret	The consumer secret provided by your application.

### Details

The API supports the OAuth 1.0a Protocol, accepting parameters via the Authorization header, with the HMAC-SHA1 signature method only.

### Value

A serialized JSON object with the following fields:

liked_posts	An array. An array of posts objects(posts liked by the user).
liked_count	A number.Total number of liked posts.

### Author(s)

Andrea Capozio

### References

<https://www.tumblr.com/docs/en/api/v2>

**Examples**

```
## Not run:
## An example of an authenticated request using the httr package,
## where consumer_key, consumer_secret, appname are fictitious.
## You can obtain your own at https://www.tumblr.com/oauth/apps

consumer_key <- 'key'
consumer_secret <- 'secret'
appname <- Tumblr_App
tokenURL <- 'https://www.tumblr.com/oauth/request_token'
accessTokenURL <- 'https://www.tumblr.com/oauth/accessToken'
authorizeURL <- 'https://www.tumblr.com/oauth/authorize'

app <- oauth_app(appname, consumer_key, consumer_secret)
endpoint <- oauth_endpoint(tokenURL, authorizeURL, accessTokenURL)
token <- oauth1.0_token(endpoint, app)
sig <- sign_oauth1.0(app,
  token = token$credentials$oauth_token,
  token_secret = token$credentials$oauth_token_secret)

user.likes(token = token, consumer_key = consumer_key,
  consumer_secret = consumer_secret)

## End(Not run)
```



# Index

- \*Topic **avatar**
    - avatar, 2
  - \*Topic **dashboard**
    - dashboard, 3
  - \*Topic **delete**
    - post.delete, 15
  - \*Topic **draft**
    - posts.draft, 24
  - \*Topic **edit**
    - post.edit, 17
  - \*Topic **followers**
    - followers, 7
  - \*Topic **following**
    - user.following, 35
  - \*Topic **follow**
    - follow, 5
  - \*Topic **info.blog**
    - info.blog, 8
  - \*Topic **likes**
    - likes, 11
    - user.likes, 39
  - \*Topic **like**
    - like.post, 10
  - \*Topic **posts**
    - posts, 22
    - posts.draft, 24
    - posts.queue, 27
    - posts.submission, 29
  - \*Topic **post**
    - post, 12
    - post.delete, 15
    - post.edit, 17
    - post.reblog, 19
  - \*Topic **queue**
    - posts.queue, 27
  - \*Topic **reblog**
    - post.reblog, 19
  - \*Topic **submission**
    - posts.submission, 29
  - \*Topic **tagged**
    - tagged, 31
  - \*Topic **unfollow**
    - unfollow, 33
  - \*Topic **unlike**
    - unlike.post, 34
  - \*Topic **user's info**
    - user.info, 37
- avatar, 2
- dashboard, 3
- follow, 5
- followers, 7
- info.blog, 8
- like.post, 10
- likes, 11
- post, 12
- post.delete, 15
- post.edit, 17
- post.reblog, 19
- posts, 22
- posts.draft, 24
- posts.queue, 27
- posts.submission, 29
- tagged, 31
- unfollow, 33
- unlike.post, 34
- user.following, 35
- user.info, 37
- user.likes, 39