

Package ‘bskyr’

October 25, 2025

Title Interact with 'Bluesky' Social

Version 0.4.0

Description Collect data from and make posts on 'Bluesky' Social via the Hypertext Transfer Protocol (HTTP) Application Programming Interface (API), as documented at <https://atproto.com/specs/xrpc>. This further supports broader queries to the Authenticated Transfer (AT) Protocol <https://atproto.com/> which 'Bluesky' Social relies on. Data is returned in a tidy format and posts can be made using a simple interface.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports cli, curl, dplyr, fs, httr2, lubridate, magick, mime, opengraph, purrr, rlang, stringi, stringr, tibble, tidyr

Suggests emoji, jsonlite, knitr, rmarkdown, spelling, testthat (>= 3.0.0), vcr (>= 2.0.0), withr

URL <https://github.com/christopherkenny/bskyr>,
<http://christophertkenny.com/bskyr/>

BugReports <https://github.com/christopherkenny/bskyr/issues>

Config/testthat/edition 3

Language en-US

Depends R (>= 4.2.0)

VignetteBuilder knitr

NeedsCompilation no

Author Christopher T. Kenny [aut, cre] (ORCID:
<https://orcid.org/0000-0002-9386-6860>)

Maintainer Christopher T. Kenny <ctkenny@proton.me>

Repository CRAN

Date/Publication 2025-10-25 07:30:12 UTC

Contents

bs_accept_convo	4
bs_add_reaction	5
bs_auth	6
bs_block	7
bs_created_at	8
bs_create_record	8
bs_delete_block	9
bs_delete_follow	11
bs_delete_like	12
bs_delete_list	13
bs_delete_list_item	14
bs_delete_message_for_self	15
bs_delete_post	16
bs_delete_record	17
bs_delete_repost	18
bs_delete_starter_pack	19
bs_describe_repo	20
bs_extract_record_key	21
bs_follow	21
bs_get_actor_lists	22
bs_get_actor_starter_packs	23
bs_get_actor_suggestions	24
bs_get_author_feed	25
bs_get_blocked_lists	26
bs_get_blocks	27
bs_get_convo	28
bs_get_convo_availability	29
bs_get_convo_for_members	30
bs_get_convo_log	31
bs_get_feed	32
bs_get_feeds	33
bs_get_feed_generator	34
bs_get_feed_generators	35
bs_get_feed_suggestions	36
bs_get_followers	37
bs_get_follows	38
bs_get_follow_suggestions	39
bs_get_likes	40
bs_get_list	41
bs_get_list_feed	42
bs_get_messages	43
bs_get_muted_lists	44
bs_get_mutes	45
bs_get_notifications	46
bs_get_notification_count	47
bs_get_posts	48

- bs_get_post_likes 49
- bs_get_post_thread 50
- bs_get_preferences 51
- bs_get_profile 52
- bs_get_quotes 53
- bs_get_record 54
- bs_get_relationships 55
- bs_get_reposts 56
- bs_get_starter_pack 57
- bs_get_starter_packs 58
- bs_get_timeline 59
- bs_leave_convo 60
- bs_like 61
- bs_list_convos 62
- bs_list_records 63
- bs_mute_convo 64
- bs_new_embed_external 65
- bs_new_list 66
- bs_new_list_item 67
- bs_new_starter_pack 68
- bs_post 69
- bs_remove_reaction 71
- bs_repost 73
- bs_resolve_handle 74
- bs_search_actors 75
- bs_search_posts 76
- bs_send_message 77
- bs_send_message_batch 78
- bs_unmute_convo 80
- bs_update_all_read 81
- bs_update_read 82
- bs_upload_blob 83
- bs_uri_to_url 84
- bs_url_to_uri 84
- pass 85
- set_bluesky_pass 86
- set_bluesky_user 86
- user 87

bs_accept_convo	<i>Accept a conversation (direct message) invitation or request</i>
-----------------	---

Description

Accept a conversation (direct message) invitation or request

Usage

```
bs_accept_convo(  
  convo_id,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.acceptConvo.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_accept_convo(convo_id = '3ku7w6h4vog2d')
```

bs_add_reaction	<i>Add a reaction (e.g. emoji) to a message in a conversation</i>
-----------------	---

Description

Add a reaction (e.g. emoji) to a message in a conversation

Usage

```
bs_add_reaction(  
  convo_id,  
  message_id,  
  value,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
message_id	Character, length 1. Message ID.
value	Character, length 1. Reaction value (e.g. an emoji).
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.addReaction.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_add_reaction(convo_id = '3ku7w6h4vog2d', message_id = '3lphbnrx7l32l', value = '\U0001F44D')
```

bs_auth	<i>Authenticate a user</i>
---------	----------------------------

Description

Authenticate a user

Usage

```
bs_auth(user, pass, save_auth = TRUE)
```

Arguments

user	Character. User name to log in with.
pass	Character. App password to log in with.
save_auth	Logical. Should the authentication information be saved? If TRUE, it tries to reload from the cache. If a file is over 10 minutes old, it will not be read. Set save_auth = NULL to force the token to refresh and save the results.

Value

a list of authentication information

Lexicon references

[server/createSession.json \(2023-09-30\)](#)

Function introduced

v0.0.1 (2023-09-30)

Examples

```
bs_auth(user = get_bluesky_user(), pass = get_bluesky_pass())
```

bs_block	<i>Block an account</i>
----------	-------------------------

Description

Block an account

Usage

```
bs_block(  
  subject,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

subject	Character, length 1. Subject to act on, as a handle or did.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of follow information

Lexicon references

[graph/block.json \(2024-12-03\)](#) [repo/createRecord.json \(2024-12-02\)](#)

Function introduced

v0.2.0 (2024-12-03)

Examples

```
bs_block(subject = 'nytimes.com')
```

bs_created_at	<i>Get current time in Bluesky format</i>
---------------	---

Description

Get current time in Bluesky format

Usage

```
bs_created_at()
```

Value

a length 1 character vector

Function introduced

v0.1.0 (2023-11-25)

Examples

```
bs_created_at()
```

bs_create_record	<i>Create a record in a repo</i>
------------------	----------------------------------

Description

Create a record in a repo

Usage

```
bs_create_record(  
  collection,  
  record,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```


Arguments

collection	Character, length 1. The NSID of the record collection.
record	List, length 1. Description of a record.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of record information

Lexicon references

[repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
# get info about a record
post_rcd <- bs_get_record('https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
# create a record, to like the post
like <- list(
  subject = list(
    uri = post_rcd$uri,
    cid = post_rcd$cid
  ),
  createdAt = bs_created_at()
)

bs_create_record(collection = 'app.bsky.feed.like', record = like)
```

bs_delete_block

Delete a block

Description

Delete a block

Usage

```
bs_delete_block(  
  rkey,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

```
bs_unblock(  
  rkey,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an `httr2` status code

Lexicon references

[graph/list.json \(2024-12-03\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-03)

Examples

```
blck <- bs_block(subject = 'nytimes.com')  
bs_delete_block(bs_extract_record_key(blck$uri))
```

bs_delete_follow	<i>Delete a follow (un-follow someone)</i>
------------------	--

Description

Delete a follow (un-follow someone)

Usage

```
bs_delete_follow(  
    rkey,  
    user = get_bluesky_user(),  
    pass = get_bluesky_pass(),  
    auth = bs_auth(user, pass)  
)  
  
bs_unfollow(  
    rkey,  
    user = get_bluesky_user(),  
    pass = get_bluesky_pass(),  
    auth = bs_auth(user, pass)  
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an http2 status code

Lexicon references

[graph/list.json \(2024-12-03\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-03)

Examples

```
foll <- bs_follow(subject = 'chriskenny.bsky.social')
bs_delete_follow(bs_extract_record_key(foll$uri))
# obviously, you deleted this by mistake and want to follow me
foll <- bs_follow(subject = 'chriskenny.bsky.social')
```

bs_delete_like	<i>Delete a like (un-like something)</i>
----------------	--

Description

Delete a like (un-like something)

Usage

```
bs_delete_like(
  rkey,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)

bs_unlike(
  rkey,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an httr2 status code

Lexicon references

[graph/list.json \(2024-12-03\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-03)

Examples

```
like <- bs_like(post = 'https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
bs_delete_like(bs_extract_record_key(like$uri))
```

bs_delete_list	<i>Delete a list</i>
----------------	----------------------

Description

Delete a list

Usage

```
bs_delete_list(  
  rkey,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).

Value

an httr2 status code

Lexicon references[graph/list.json \(2024-12-01\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)**Function introduced**

v0.2.0 (2024-12-01)

Examples

```
lst <- bs_new_list(name = 'test list bskyr', purpose = 'curatelist')
bs_delete_list(bs_extract_record_key(lst$uri))
```

bs_delete_list_item *Delete a list item*

Description

Delete a list item

Usage

```
bs_delete_list_item(
  rkey,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an httr2 status code

Lexicon references

[graph/listitem.json \(2024-12-01\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-01)

Examples

```
lst <- bs_new_list(name = 'test list bskyr', purpose = 'curatelist')
itm <- bs_new_list_item(subject = 'bskyr.bsky.social', uri = lst$uri)
bs_delete_list_item(bs_extract_record_key(itm$uri))
bs_delete_list(bs_extract_record_key(lst$uri))
```

`bs_delete_message_for_self`

Remove a message from your view of a conversation (does not delete it for others)

Description

Remove a message from your view of a conversation (does not delete it for others)

Usage

```
bs_delete_message_for_self(  
  convo_id,  
  message_id,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>convo_id</code>	Character, length 1. ID of the conversation to get.
<code>message_id</code>	Character, length 1. Message ID.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.deleteMessageForSelf.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_delete_message_for_self(convo_id = '3ku7w6h4vog2d', message_id = '3lpi4fcbnxv2l')
```

bs_delete_post	<i>Delete a post</i>
----------------	----------------------

Description

Delete a post

Usage

```
bs_delete_post(  
  rkey,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an httr2 status code

Lexicon references

[feed/post.json \(2025-03-20\)](#) [repo/deleteRecord.json \(2025-03-20\)](#)

Function introduced

v0.3.0 (2025-03-20)

Examples

```
pst <- bs_post('a test post to be deleted')  
bs_delete_post(bs_extract_record_key(pst$suri))
```

bs_delete_record	<i>Delete a record in a repo</i>
------------------	----------------------------------

Description

Delete a record in a repo

Usage

```
bs_delete_record(  
  collection,  
  rkey,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

collection	Character, length 1. The NSID of the record collection.
rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an http2 status code

Lexicon references

[repo/deleteRecord.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
# get info about a record  
post_rcd <- bs_get_record('https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')  
# create a record, to like the post  
like <- list(  
  subject = list(  
    uri = post_rcd$uri,  
    cid = post_rcd$cid
```

```

    ),
    createdAt = bs_created_at()
  )

rec <- bs_create_record(collection = 'app.bsky.feed.like', record = like)
bs_delete_record(
  collection = 'app.bsky.feed.like',
  rkey = bs_extract_record_key(rec$uri)
)

```

bs_delete_repost	<i>Delete a repost</i>
------------------	------------------------

Description

Delete a repost

Usage

```

bs_delete_repost(
  rkey,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)

```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an httr2 status code

Lexicon references

[feed/repost.json \(2023-11-25\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-03)

Examples

```
repo <- bs_repost('https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
bs_delete_repost(bs_extract_record_key(repo$uri))
```

bs_delete_starter_pack

Delete a starter pack

Description

Delete a starter pack

Usage

```
bs_delete_starter_pack(
  rkey,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)
```

Arguments

rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

an httr2 status code

Lexicon references

[graph/starterpack.json \(2024-12-04\)](#) [repo/deleteRecord.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-04)

Examples

```
starter <- bs_new_starter_pack('bskyr test')
bs_delete_starter_pack(bs_extract_record_key(starter$uri))
```

bs_describe_repo	<i>Describe a repo</i>
------------------	------------------------

Description

Describe a repo

Usage

```
bs_describe_repo(  
  repo,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

repo	Character, length 1. The handle or DID of the repo.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of record information

Lexicon references

[repo/describeRepo.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
bs_describe_repo('chriskenny.bsky.social')
```

bs_extract_record_key *Extract Record Key from a link*

Description

Extract Record Key from a link

Usage

```
bs_extract_record_key(url)
```

Arguments

url Character, length 1. URL for record to get.

Value

character vector of record keys

Examples

```
bs_extract_record_key('https://bsky.app/profile/chriskenny.bsky.social/post/3lc5d6zspys2c')
```

bs_follow *Follow an account*

Description

Follow an account

Usage

```
bs_follow(
  subject,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

subject Character, length 1. Subject to act on, as a handle or did.
user Character. User name to log in with. Defaults to get_bluesky_user().
pass Character. App password to log in with. Defaults to get_bluesky_pass().
auth Authentication information. Defaults to bs_auth(user, pass).
clean Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of follow information

Lexicon references

[graph/list.json \(2024-12-02\) repo/createRecord.json \(2024-12-02\)](#)

Function introduced

v0.2.0 (2024-12-02)

Examples

```
bs_follow(subject = 'chriskenny.bsky.social')
```

`bs_get_actor_lists` *Get a list of lists that belong to an actor.*

Description

Get a list of lists that belong to an actor.

Usage

```
bs_get_actor_lists(  
  actor,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>actor</code>	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
<code>cursor</code>	Character, length 1. A cursor property from a prior response. Default: NULL.
<code>limit</code>	Integer. Number of records to request. If over 100, multiple requests are made.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of lists

Lexicon references

[graph/getLists.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_actor_lists('profmusgrave.bsky.social')
bs_get_actor_lists('chriskenny.bsky.social')
bs_get_actor_lists('pfrazee.com')
```

bs_get_actor_starter_packs

Get starter packs created by an actor

Description

Get starter packs created by an actor

Usage

```
bs_get_actor_starter_packs(
  actor,
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 50, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of starter packs

Lexicon references

[graph/getActorStarterPacks.json \(2024-11-20\)](#)

Function introduced

v0.2.0 (2024-11-20)

Examples

```
bs_get_actor_starter_packs('chriskenny.bsky.social')
bs_get_actor_starter_packs('pfrazee.com')
```

bs_get_actor_suggestions

Get a list of actors suggested for following

Description

Get a list of actors suggested for following

Usage

```
bs_get_actor_suggestions(  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of suggested accounts to follow

Lexicon references

[actor/getSuggestions.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_actor_suggestions()
```

bs_get_author_feed	<i>Retrieve posts on an actor's feed</i>
--------------------	--

Description

Retrieve posts on an actor's feed

Usage

```
bs_get_author_feed(  
  actor,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of posts

Lexicon references

[feed/getAuthorFeed.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_author_feed('chriskenny.bsky.social')
```

`bs_get_blocked_lists` *Retrieve a user's (self) muted lists*

Description

Retrieve a user's (self) muted lists

Usage

```
bs_get_blocked_lists(  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>cursor</code>	Character, length 1. A cursor property from a prior response. Default: NULL.
<code>limit</code>	Integer. Maximum number to request.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references[graph/getListMutes.json \(2023-10-02\)](#)**Function introduced**

v0.0.1 (2023-10-02)

Examples`bs_get_blocked_lists()`

bs_get_blocks	<i>Retrieve user (self) blocks</i>
---------------	------------------------------------

Description

Retrieve user (self) blocks

Usage

```
bs_get_blocks(
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Valuea `tibble::tibble` of blocked accounts**Lexicon references**[graph/getBlocks.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

bs_get_blocks()

`bs_get_convo`*Retrieve detailed information about a conversation by its ID*

Description

Retrieve detailed information about a conversation by its ID

Usage

```
bs_get_convo(
  convo_id,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

<code>convo_id</code>	Character, length 1. ID of the conversation to get.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Valuea `tibble::tibble` or a list if `clean = FALSE`**Lexicon references**[chat.bsky.convo.getConvo.json \(2025-05-16\)](#)**Function introduced**

v0.4.0 (2025-05-16)

Examples

```
bs_get_convo(convo_id = '3ku7w6h4vog2d')
```

`bs_get_convo_availability`*Check conversation availability with specified members*

Description

Check conversation availability with specified members

Usage

```
bs_get_convo_availability(  
  actors,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>actors</code>	character vector of actor(s), such as 'chriskenny.bsky.social'
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

A `tibble::tibble` or a list if `clean = FALSE`.

Lexicon references

[chat.bsky.convo.getConvoAvailability.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_get_convo_availability(actors = 'chriskenny.bsky.social')
```

`bs_get_convo_for_members`*Retrieve conversation shared among specified members*

Description

Retrieve conversation shared among specified members

Usage

```
bs_get_convo_for_members(  
  actors,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>actors</code>	character vector of actor(s), such as 'chriskenny.bsky.social'
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

A `tibble::tibble` or a list if `clean = FALSE`.

Lexicon references

[chat.bsky.convo.getConvoForMembers.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_get_convo_for_members(actors = c('bskyr.bsky.social', 'chriskenny.bsky.social'))
```

bs_get_convo_log	<i>Retrieve the chat event log for the authenticated user</i>
------------------	---

Description

Retrieve the chat event log for the authenticated user

Usage

```
bs_get_convo_log(  
  cursor = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.getLog.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_get_convo_log()
```

`bs_get_feed`*Build feed from user's feed generator*

Description

Build feed from user's feed generator

Usage

```
bs_get_feed(  
  feed,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>feed</code>	Character, length 1. Feed to get.
<code>cursor</code>	Character, length 1. A cursor property from a prior response. Default: NULL.
<code>limit</code>	Integer. Number of records to request. If over 100, multiple requests are made.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of posts

Lexicon references

[feed/getFeed.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_feed('at://did:plc:z72i7hdynmk6r22z27h6tvur/app.bsky.feed.generator/bsky-team')
```

bs_get_feeds	<i>Retrieve a list of feeds created by a given actor</i>
--------------	--

Description

Retrieve a list of feeds created by a given actor

Usage

```
bs_get_feeds(  
  actor,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of feeds

Lexicon references

[feed/getActorFeeds.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_feeds('chriskenny.bsky.social')
```

bs_get_feed_generator *Get specific information about one feed generator*

Description

Get specific information about one feed generator

Usage

```
bs_get_feed_generator(  
  feed,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

feed	Character, length 1. Feed to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of feeds

Lexicon references

[feed/getFeedGenerator.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

See Also

[bs_get_feed_generators\(\)](#) for less detailed information about multiple feed generators.

Examples

```
bs_get_feed_generator('at://did:plc:z72i7hdynmk6r22z27h6tvur/app.bsky.feed.generator/bsky-team')
```

`bs_get_feed_generators`*Get information about a list of feed generators*

Description

Get information about a list of feed generators

Usage

```
bs_get_feed_generators(  
  feeds,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>feeds</code>	Character. Vector of feeds to get.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of feeds

Lexicon references

[feed/getFeedGenerators.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

See Also

[bs_get_feed_generators\(\)](#) for more detailed information about one feed generator.

Examples

```
bs_get_feed_generators('at://did:plc:z72i7hdynmk6r22z27h6tvur/app.bsky.feed.generator/bsky-team')
bs_get_feed_generators(c(
  'at://did:plc:z72i7hdynmk6r22z27h6tvur/app.bsky.feed.generator/bsky-team',
  'at://did:plc:z72i7hdynmk6r22z27h6tvur/app.bsky.feed.generator/whats-hot'
))
```

bs_get_feed_suggestions

Get a list of feed suggestions

Description

Get a list of feed suggestions

Usage

```
bs_get_feed_suggestions(
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of suggested feeds

Lexicon references

[feed/getSuggestedFeeds.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_feed_suggestions()
```

bs_get_followers	<i>Retrieve an actor's followers</i>
------------------	--------------------------------------

Description

Retrieve an actor's followers

Usage

```
bs_get_followers(  
  actor,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[graph/getFollowers.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_followers('chriskenny.bsky.social')
```

bs_get_follows	<i>Retrieve an actor's follows</i>
----------------	------------------------------------

Description

Retrieve an actor's follows

Usage

```
bs_get_follows(
  actor,
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[graph/getFollows.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_follows('chriskenny.bsky.social')
```

```
bs_get_follow_suggestions
```

Get suggested follows related to a given actor

Description

Get suggested follows related to a given actor

Usage

```
bs_get_follow_suggestions(  
  actor,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[graph/getSuggestedFollowsByActor.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_follow_suggestions('chriskenny.bsky.social')
```

bs_get_likes	<i>Retrieve posts liked by an actor (self)</i>
--------------	--

Description

Retrieve posts liked by an actor (self)

Usage

```
bs_get_likes(  
  actor,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of likes

Lexicon references

[feed/getActorLikes.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_likes(bs_get_user())
```

bs_get_list	<i>Get a view of a list</i>
-------------	-----------------------------

Description

Get a view of a list

Usage

```
bs_get_list(  
  list,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

list	Character vector, length 1. Reference of the list record to get.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of lists

Lexicon references

[graph/getList.json \(2025-03-20\)](#)

Function introduced

v0.2.0 (2024-11-25)

Examples

```
bs_get_list('at://did:plc:ragtj2j2vknwkz3zp4oxrd/app.bsky.graph.list/3kmokjyuf1k2g')  
bs_get_list('at://did:plc:hgyz2hn6zxpqokmp5c2xrdo/app.bsky.graph.list/3laygnmmfc2x')
```

bs_get_list_feed *Retrieve recent posts from actors in a list*

Description

Retrieve recent posts from actors in a list

Usage

```
bs_get_list_feed(  
  list,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

list	Character vector, length 1. Reference of the list record to get.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of likes

Lexicon references

[feed/getListFeed.json \(2025-03-20\)](#)

Function introduced

v0.3.0 (2025-03-20)

Examples

```
bs_get_list_feed('at://did:plc:ragtj2j2vknwkz3zp4oxrd/app.bsky.graph.list/3kmokjyuf1k2g')
```

bs_get_messages	<i>Retrieve messages from a conversation, optionally paginated</i>
-----------------	--

Description

Retrieve messages from a conversation, optionally paginated

Usage

```
bs_get_messages(  
  convo_id,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.getMessages.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_get_messages(convo_id = '3ku7w6h4vog2d', limit = 10)
```

bs_get_muted_lists *Retrieve a user's (self) muted lists*

Description

Retrieve a user's (self) muted lists

Usage

```
bs_get_muted_lists(  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[graph/getListMutes.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_muted_lists()
```

bs_get_mutes	<i>Retrieve a user's (self) muted accounts</i>
--------------	--

Description

Retrieve a user's (self) muted accounts

Usage

```
bs_get_mutes(  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[graph/getMutes.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_mutes()
```

bs_get_notifications *Get the user's (self) notifications*

Description

Get the user's (self) notifications

Usage

```
bs_get_notifications(  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a tibble with notifications

Lexicon references

[notification/listNotifications.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_notifications()
```

`bs_get_notification_count`*Get the user's (self) number of unread notifications*

Description

Get the user's (self) number of unread notifications

Usage

```
bs_get_notification_count(  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a tibble with a single column and row for the count

Lexicon references

[notification/getUnreadCount.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_notification_count()
```

`bs_get_posts`*Retrieve thread of posts*

Description

Retrieve thread of posts

Usage

```
bs_get_posts(  
  uris,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>uris</code>	Character. Vector of URIs for posts to get.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of posts

Lexicon references

[feed/getPosts.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_posts('at://did:plc:ic6zqvuw5ulmfpjiwnhsr2ns/app.bsky.feed.post/3k7qmjev51r2s')  
bs_get_posts('https://bsky.app/profile/chriskenny.bsky.social/post/3lc5d6zspys2c')
```

bs_get_post_likes	<i>Retrieve likes on a post</i>
-------------------	---------------------------------

Description

Retrieve likes on a post

Usage

```
bs_get_post_likes(  
  uri,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

uri	Character, length 1. URI for post to get.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of likes

Lexicon references

[feed/getLikes.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_post_likes('at://did:plc:ic6zqvuw5ulmfpjiwnhsr2ns/app.bsky.feed.post/3k7qmjev51r2s')
```

bs_get_post_thread *Retrieve thread of posts*

Description

Retrieve thread of posts

Usage

```
bs_get_post_thread(  
  uri,  
  depth = NULL,  
  parent_height = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

uri	Character, length 1. URI for post to get.
depth	Integer. Maximum depth to request. Maximum: 1000
parent_height	Integer. Maximum parent height to request.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of posts

Lexicon references

[feed/getPostThread.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_post_thread('at://did:plc:ic6zqvuw5ulmfpjiwnhsr2ns/app.bsky.feed.post/3k7qmjev51r2s')
```

bs_get_preferences *Get (Self) Preferences*

Description

Get (Self) Preferences

Usage

```
bs_get_preferences(  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of preferences

Lexicon references

[actor/getPreferences.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_preferences()
```

`bs_get_profile`*Get Profile for a Bluesky Social User*

Description

Get Profile for a Bluesky Social User

Usage

```
bs_get_profile(  
  actors,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

<code>actors</code>	character vector of actor(s), such as 'chriskenny.bsky.social'
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a tibble with a row for each actor

Lexicon references

[actor/getProfiles.json \(2023-10-01\)](#) [actor/getProfile.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_get_profile('chriskenny.bsky.social')  
bs_get_profile(actors = c('chriskenny.bsky.social', 'simko.bsky.social'))
```

bs_get_quotes	<i>Retrieve a list of quotes for a given post</i>
---------------	---

Description

Retrieve a list of quotes for a given post

Usage

```
bs_get_quotes(  
  uri,  
  cid,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

uri	Character, length 1. URI for post to get.
cid	Optional, character. Filters to quotes of specific version (by CID) of the post record
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of quote posts

Lexicon references

[feed/getQuotes.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-01)

Examples

```
bs_get_quotes('at://did:plc:5c2r73erhng4bszmx1fdtscf/app.bsky.feed.post/3lc5c5qv72r2w')
```

bs_get_record
Get an arbitrary record from a repo

Description

Get an arbitrary record from a repo

Usage

```
bs_get_record(
  repo = NULL,
  collection = NULL,
  rkey = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

repo	Character, length 1. The handle or DID of the repo.
collection	Character, length 1. The NSID of the record collection.
rkey	Character, length 1. The CID of the version of the record. If not specified, then return the most recent version.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a [tibble::tibble](#) of upload blob information

Lexicon references

[repo/getRecord.json \(2023-11-24\)](#)

Function introduced

v0.1.0 (2023-11-24)

Examples

```
bs_get_record('https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
```

bs_get_relationships *Get relationships between an account and other users*

Description

Get relationships between an account and other users

Usage

```
bs_get_relationships(  
  actor,  
  others,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

actor	Character, length 1. name of 1 actor, such as 'chriskenny.bsky.social'
others	Optional, character vector of other users to look up relationships
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of relationships

Lexicon references

[graph/getRelationships.json \(2024-12-01\)](#)

Function introduced

v0.2.0 (2024-12-01)

Examples

```
bs_get_relationships('chriskenny.bsky.social', 'bskyr.bsky.social')
```

bs_get_reposts	<i>Retrieve actors who reposted a post</i>
----------------	--

Description

Retrieve actors who reposted a post

Usage

```
bs_get_reposts(  
  uri,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

uri	Character, length 1. URI for post to get.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of actors

Lexicon references

[feed/getRepostedBy.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_reposts('at://did:plc:ic6zqvuw5ulmfpjiwnhsr2ns/app.bsky.feed.post/3kaa2gxjh2a')
```

bs_get_starter_pack *Get information on one starter pack*

Description

Get information on one starter pack

Usage

```
bs_get_starter_pack(  
  starter_pack,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

starter_pack	Character vector, length 1. URI of starter pack to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of starter packs

Lexicon references

[graph/getStarterPack.json \(2024-11-20\)](#)

Function introduced

v0.2.0 (2024-11-20)

Examples

```
bs_get_starter_pack(  
  'at://did:plc:wpe35pganb6d4pg4ekmfy6u5/app.bsky.graph.starterpack/31b3g5veo2z2r'  
)
```

bs_get_starter_packs *Get information about a list of starter packs*

Description

Get information about a list of starter packs

Usage

```
bs_get_starter_packs(  
  starter_packs,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

starter_packs	Character vector. Vector of URIs of starter packs to get.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of starter packs

Lexicon references

[graph/getStarterPacks.json \(2024-11-20\)](#)

Function introduced

v0.2.0 (2024-11-20)

Examples

```
bs_get_starter_packs(  
  'at://did:plc:wpe35pganb6d4pg4ekmfy6u5/app.bsky.graph.starterpack/31b3g5veo2z2r'  
)  
bs_get_starter_packs(  
  c(  
    'at://did:plc:wpe35pganb6d4pg4ekmfy6u5/app.bsky.graph.starterpack/31b3g5veo2z2r',  
    'at://did:plc:bmc56x6ksb7o7sdkq2fgm7se/app.bsky.graph.starterpack/3laywns2q2v27'  
  )  
)
```

```
)
```

bs_get_timeline	<i>Retrieve the user's home timeline</i>
-----------------	--

Description

Retrieve the user's home timeline

Usage

```
bs_get_timeline(  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of posts

Lexicon references

[feed/getTimeline.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_get_timeline()
```

bs_leave_convo	<i>Exit a conversation so you no longer receive messages or see it in your inbox</i>
----------------	--

Description

Exit a conversation so you no longer receive messages or see it in your inbox

Usage

```
bs_leave_convo(  
  convo_id,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.leaveConvo.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_leave_convo(convo_id = '31pidxucy2g27')
```

bs_like	<i>Like an existing post</i>
---------	------------------------------

Description

Like an existing post

Usage

```
bs_like(  
  post,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

post	Character vector, length 1. Link to a post.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of post information

Lexicon references

[feed/like.json \(2023-11-25\)](#) [repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
bs_like(post = 'https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
```

bs_list_convos	<i>List the conversations (direct message threads) for the authenticated user</i>
----------------	---

Description

List the conversations (direct message threads) for the authenticated user

Usage

```
bs_list_convos(
  read_state = NULL,
  status = NULL,
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

read_state	Character, optional. Filter by read state, one of c('unread'). Default: NULL.
status	Character, optional. Filter by conversation status, one of c('accepted', 'request'). Default: NULL.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.listConvos.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_list_convos(limit = 5, status = 'accepted')
```

bs_list_records	<i>List records in a repo</i>
-----------------	-------------------------------

Description

List records in a repo

Usage

```
bs_list_records(  
  repo,  
  collection,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

repo	Character, length 1. The handle or DID of the repo.
collection	Character, length 1. The NSID of the record collection.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of record information

Lexicon references

[repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
bs_list_records(repo = 'chriskenny.bsky.social', collection = 'app.bsky.feed.post')
```

bs_mute_convo
Mute a conversation so you no longer receive notifications.

Description

Mute a conversation so you no longer receive notifications.

Usage

```
bs_mute_convo(
  convo_id,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.muteConvo.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_mute_convo(convo_id = '3ku7w6h4vog2d')
```

bs_new_embed_external *Embed external media in a post*

Description

Embeds are not designed as standalone records, but rather as part of a post. This will create a list representation of an external embed.

Usage

```
bs_new_embed_external(  
  uri,  
  title,  
  description,  
  thumb,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

uri	a link to embed
title	the title for the link
description	a description of the link
thumb	Optional. A thumbnail for the link
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

a list representation of an external embed

Lexicon references

[embed/external.json \(2024-12-05\)](#)

Function introduced

v0.2.0 (2024-12-05)

Examples

```
bs_new_embed_external(
  uri = 'https://christophertkenny.com/bskyr/',
  title = 'Interact with Bluesky Social',
  description = 'An R package for using Bluesky Social'
)

bs_new_embed_external(
  uri = 'https://christophertkenny.com/bskyr/'
)
```

`bs_new_list`*Create a list*

Description

Create a list

Usage

```
bs_new_list(
  name,
  purpose,
  description,
  avatar,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

<code>name</code>	Character. Display name for list.
<code>purpose</code>	Purpose of the list. One of 'modlist', 'curatelist', or 'referencelist'
<code>description</code>	Optional character. Description of the list.
<code>avatar</code>	Optional character. Path to image to use as avatar. PNG or JPEG recommended.
<code>user</code>	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
<code>pass</code>	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
<code>auth</code>	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
<code>clean</code>	Logical. Should output be cleaned into a tibble? Default: TRUE.

Valuea `tibble::tibble` of list information

Lexicon references

[graph/list.json \(2024-12-01\)](#) [graph/defs.json \(2024-12-01\)](#) [repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.2.0 (2024-12-01)

Examples

```
bs_new_list(name = 'test list bskyr', purpose = 'curatelist')
bs_new_list(
  name = 'test list bskyr w avatar',
  description = 'to be deleted, just for testing bskyr',
  avatar = fs::path_package('bskyr', 'man/figures/logo.png'),
  purpose = 'curatelist'
)
```

bs_new_list_item	<i>Add a subject to a list</i>
------------------	--------------------------------

Description

Add a subject to a list

Usage

```
bs_new_list_item(
  subject,
  uri,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

subject	Character, length 1. Subject to act on, as a handle or did.
uri	Character, length 1. URI of the list to add the subject to.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of list item information

Lexicon references

[graph/listitem.json \(2024-12-01\)](#) [repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.2.0 (2024-12-01)

Examples

```
lst <- bs_new_list(name = 'test list listitem bskyr', purpose = 'curatelist')
bs_new_list_item(subject = 'chriskenny.bsky.social', uri = lst$uri)
# see the list item
bs_get_list(lst$uri)
```

bs_new_starter_pack *Create a new starter pack*

Description

Create a new starter pack

Usage

```
bs_new_starter_pack(
  name,
  list,
  description,
  feeds,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

name	Character. Display name for starter pack
list	Character. List to base the starter pack on. If not provided, a new list will be created.
description	Optional character. Description of the list.
feeds	Optional character. List of feed items to include in starter pack.

user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of post information

Lexicon references

[graph/starterpack.json \(2024-12-04\)](#) [repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.2.0 (2024-12-04)

Examples

```
bs_new_starter_pack('bskyr test')
```

bs_post	<i>Make a post on Bluesky Social</i>
---------	--------------------------------------

Description

Make a post on Bluesky Social

Usage

```
bs_post(
  text,
  images,
  images_alt,
  video,
  video_alt,
  langs,
  reply,
  quote,
  embed = TRUE,
  emoji = TRUE,
  max_tries,
  created_at = bs_created_at(),
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

text	Text of post
images	Character vector of paths to images to attach to post
images_alt	Character vector of alt text for images. Must be same length as images if used.
video	Character vector of path for up to one video to attach to post
video_alt	Character vector, length one, of alt text for video, if used.
langs	Character vector of languages in BCP-47 format
reply	Character vector with link to the parent post to reply to
quote	Character vector with link to a post to quote
embed	Logical. Default is TRUE. Should a link card be embedded?
emoji	Logical. Default is TRUE. Should <code>:emoji:</code> style references be converted?
max_tries	Integer, ≥ 2 . Number of times to retry the request if the first fails.
created_at	Character, length 1 of the form "%Y-%m-%dT%H:%M:%OS6Z". Time to assign to a record. Default is <code>bs_created_at()</code> .
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of post information

Emoji parsing

`:emoji:` parsing is not a formally supported Bluesky feature. This package converts usages of this kind by identifying text within `:s`, here "emoji" and then matches them to the emoji package's list of emoji names. All supported emoji names and corresponding images can be seen with `emoji::emoji_name`. This feature was introduced in `v0.2.0`.

Embedding

Embedding is a feature that allows for a link card to be created when a URL or other media to be added as a preview to the post. This feature was introduced in `v0.2.0`.

Embeds are processed as follows:

1. If `is.list(embed)`, it is assumed to be an embed object. These should be created with `bs_new_embed_external()`, unless you are certain of the structure.
2. If `is.character(embed)`, it is assumed to be a URL. The function will use the open graph protocol to try to infer the embed from the URL.
3. If `isTRUE(embed)`, the *default*, it tries to infer the embed from the text.
4. First, if a Tenor Gif link is found in the text, it will be embedded.
5. Second, if a URL is found in the text, it will be embedded. Only the first URL found will be embedded.
6. If `isFALSE(embed)` or it does not match an earlier condition, no embed is created and the post is sent as is.

Lexicon references

[feed/post.json \(2024-11-29\)](#) [repo/createRecord.json \(2023-10-02\)](#)

Function introduced

v0.0.1 (2023-10-02)

Examples

```
bs_post('Test post from R CMD Check for r package `bskyr`
via @bskyr.bsky.social (https://christophertkenny.com/bskyr/)')
bs_post('Test self-reply from r package `bskyr`
via @bskyr.bsky.social (https://christophertkenny.com/bskyr/)',
  reply = 'https://bsky.app/profile/bskyr.bsky.social/post/3kexwuoyqj32g'
)
bs_post('Test quoting from r package `bskyr`
via @bskyr.bsky.social (https://christophertkenny.com/bskyr/)',
  quote = 'https://bsky.app/profile/bskyr.bsky.social/post/3kf24wd6cmb2a'
)
bs_post('Test quote and reply from r package `bskyr`
via @bskyr.bsky.social (https://christophertkenny.com/bskyr/)',
  reply = 'https://bsky.app/profile/bskyr.bsky.social/post/3kexwuoyqj32g',
  quote = 'https://bsky.app/profile/bskyr.bsky.social/post/3kf24wd6cmb2a'
)

bs_post('Test quote with :emoji: and :fire: and :confetti_ball: from r package
`bskyr` via @bskyr.bsky.social (https://christophertkenny.com/bskyr/)')

bs_post(
  text = 'Testing images and aspect ratios from R',
  images = fs::path_package('bskyr', 'man/figures/logo.png'),
  images_alt = 'hexagonal logo of the R package bskyr, with the text "bskyr" on a cloud'
)

bs_post(
  text = 'testing sending videos from R',
  video = fs::path_package('bskyr', 'man/figures/pkgs.mp4'),
  video_alt = 'a carousel of package logos, all hexagonal'
)
```

bs_remove_reaction *Remove a previously added reaction from a message*

Description

Remove a previously added reaction from a message

Usage

```
bs_remove_reaction(  
  convo_id,  
  message_id,  
  value,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
message_id	Character, length 1. Message ID.
value	Character, length 1. Reaction to remove.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a [tibble::tibble](#) or a list if clean = FALSE

Lexicon references

[chat.bsky.convo.removeReaction.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_remove_reaction(convo_id = '3ku7w6h4vog2d', message_id = '3lphbnrx71321', value = '\U0001F44D')
```

bs_repost	<i>Repost an existing post</i>
-----------	--------------------------------

Description

Repost an existing post

Usage

```
bs_repost(  
  post,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

post	Character vector, length 1. Link to a post.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of post information

Lexicon references

[feed/repost.json \(2023-11-25\)](#) [repo/createRecord.json \(2023-11-25\)](#)

Function introduced

v0.1.0 (2023-11-25)

Examples

```
bs_repost('https://bsky.app/profile/bskyr.bsky.social/post/3kf2577exva2x')
```

bs_resolve_handle *Resolve a Handle to Decentralized Identifier (DID)*

Description

Resolve a Handle to Decentralized Identifier (DID)

Usage

```
bs_resolve_handle(  
  handle,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

handle	Character, length 1. Handle, such as 'chriskenny.bsky.social'
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of decentralized identifier

Lexicon references

[identity/resolveHandle.json \(2023-11-24\)](#)

Function introduced

v0.1.0 (2023-11-24)

Examples

```
bs_resolve_handle('chriskenny.bsky.social')
```

bs_search_actors	<i>Find profiles matching search criteria</i>
------------------	---

Description

Find profiles matching search criteria

Usage

```
bs_search_actors(  
  query,  
  typeahead = FALSE,  
  cursor = NULL,  
  limit = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

query	character. search query, Lucene query syntax is recommended when typeahead = FALSE.
typeahead	logical. Use typeahead for search? Default is FALSE.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a [tibble::tibble](#) of suggested accounts to follow

Lexicon references

[actor/searchActors.json \(2023-10-01\)](#) [actor/searchActorsTypeahead.json \(2023-10-01\)](#)

Function introduced

v0.0.1 (2023-10-01)

Examples

```
bs_search_actors('political science')
```

bs_search_posts	<i>Find posts matching search criteria</i>
-----------------	--

Description

Find posts matching search criteria

Usage

```
bs_search_posts(
  query,
  sort = NULL,
  since = NULL,
  until = NULL,
  mentions = NULL,
  author = NULL,
  lang = NULL,
  domain = NULL,
  url = NULL,
  tag = NULL,
  cursor = NULL,
  limit = NULL,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass),
  clean = TRUE
)
```

Arguments

query	Character vector, length 1. character. Search query, Lucene query syntax is recommended.
sort	character. Order of results. Either 'top' or 'latest'
since	character. Filter results for posts on or after the indicated datetime or ISO date (YYYY-MM-DD).
until	character. Filter results for posts before the indicated datetime or ISO date (YYYY-MM-DD).
mentions	character. Filter to posts which mention the given account.
author	character. Filter to posts by the given account.
lang	character. Filter to posts in the given language.

domain	character. Filter to posts with URLs (facet links or embeds) linking to the given domain (hostname). Server may apply hostname normalization.
url	character. Filter to posts with links (facet links or embeds) pointing to this URL. Server may apply URL normalization or fuzzy matching.
tag	character. Filter to posts with the given tag (hashtag), based on rich-text facet or tag field. Do not include the hash (#) prefix. Multiple tags can be specified, with 'AND' matching.
cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Number of records to request. If over 100, multiple requests are made.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of suggested accounts to follow

Lexicon references

[feed/searchPosts.json \(2024-11-25\)](#)

Function introduced

v0.1.1 (2023-12-13)

Examples

```
bs_search_posts('redistricting')
bs_search_posts('ggplot2', tag = 'rstats', sort = 'latest')
```

bs_send_message	<i>Send a message in a conversation (DM thread)</i>
-----------------	---

Description

Send a message in a conversation (DM thread)

Usage

```
bs_send_message(  
  convo_id,  
  text,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
text	Character, length 1. Message text.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.sendMessage.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_send_message(convo_id = '3ku7w6h4vog2d', text = '[example] sent with bskyr')
```

`bs_send_message_batch` *Send multiple messages across different conversations*

Description

Send multiple messages across different conversations

Usage

```
bs_send_message_batch(  
  convo_id,  
  text,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
text	Character vector of message texts.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

A `tibble::tibble` or a list if `clean = FALSE`.

Lexicon references

[chat.bsky.convo.sendMessageBatch.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_send_message_batch(  
  convo_id = c('3ku7w6h4vog2d', '3lpidxucy2g27'),  
  text = c('Hello', 'Hi there')  
)
```

bs_unmute_convo	<i>Unmute a previously muted conversation</i>
-----------------	---

Description

Unmute a previously muted conversation

Usage

```
bs_unmute_convo(  
  convo_id,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.unmuteConvo.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_unmute_convo(convo_id = '3ku7w6h4vog2d')
```

bs_update_all_read *Mark all conversations as read for the authenticated user*

Description

Mark all conversations as read for the authenticated user

Usage

```
bs_update_all_read(  
  status = c("accepted", "request"),  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

status	Character, length 1. Conversation status, one of c("accepted", "request"). Default: NULL.
user	Character. User name to log in with. Defaults to get_bluesky_user().
pass	Character. App password to log in with. Defaults to get_bluesky_pass().
auth	Authentication information. Defaults to bs_auth(user, pass).
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.updateAllRead.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_update_all_read()
```

bs_update_read	<i>Mark a conversation (optionally up to a specific message) as read</i>
----------------	--

Description

Mark a conversation (optionally up to a specific message) as read

Usage

```
bs_update_read(  
  convo_id,  
  message_id = NULL,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

convo_id	Character, length 1. ID of the conversation to get.
message_id	Character, optional. Message ID up to which to mark as read.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` or a list if `clean = FALSE`

Lexicon references

[chat.bsky.convo.updateRead.json \(2025-05-16\)](#)

Function introduced

v0.4.0 (2025-05-16)

Examples

```
bs_update_read(convo_id = '3ku7w6h4vog2d')
```

bs_upload_blob	<i>Upload a blob to a repo</i>
----------------	--------------------------------

Description

Upload a blob to a repo

Usage

```
bs_upload_blob(  
  blob,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass),  
  clean = TRUE  
)
```

Arguments

blob	Character, files to upload to a repo.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .
clean	Logical. Should output be cleaned into a tibble? Default: TRUE.

Value

a `tibble::tibble` of upload blob information

Lexicon references

[repo/uploadBlob.json \(2023-11-24\)](#)

Function introduced

v0.1.0 (2023-11-24)

Examples

```
fig <- fs::path_package('bskyr', 'man/figures/logo.png')  
bs_upload_blob(fig)
```

bs_uri_to_url	<i>Convert Universal Resource Identifiers to Hypertext Transfer Protocol Secure URLs</i>
---------------	--

Description

Convert Universal Resource Identifiers to Hypertext Transfer Protocol Secure URLs

Usage

```
bs_uri_to_url(uri)
```

Arguments

uri Character, length 1. URI for post to get.

Value

character vector of HTTPS URLs

Function introduced

v0.1.0 (2023-11-24)

Examples

```
bs_uri_to_url('at://did:plc:ic6zqvuw5ulmfpjiwnhsr2ns/app.bsky.feed.post/3k7qmjev5lr2s')
```

bs_url_to_uri	<i>Convert Hypertext Transfer Protocol Secure URLs to Universal Resource Identifiers</i>
---------------	--

Description

Convert Hypertext Transfer Protocol Secure URLs to Universal Resource Identifiers

Usage

```
bs_url_to_uri(  
  url,  
  user = get_bluesky_user(),  
  pass = get_bluesky_pass(),  
  auth = bs_auth(user, pass)  
)
```

Arguments

url	Character, length 1. URL for record to get.
user	Character. User name to log in with. Defaults to <code>get_bluesky_user()</code> .
pass	Character. App password to log in with. Defaults to <code>get_bluesky_pass()</code> .
auth	Authentication information. Defaults to <code>bs_auth(user, pass)</code> .

Value

character vector of URIs

Function introduced

v0.2.0 (2024-11-30)

Examples

```
bs_url_to_uri('https://bsky.app/profile/chriskenny.bsky.social/post/3lc5d6zspys2c')
```

pass

Check or Get Bluesky App Password

Description

Check or Get Bluesky App Password

Usage

```
has_bluesky_pass()
```

```
get_bluesky_pass()
```

```
bs_get_pass()
```

```
bs_has_pass()
```

Value

logical if has, pass if get

Examples

```
has_bluesky_pass()
```

set_bluesky_pass *Add Entry to Renviron*

Description

Adds Bluesky App Password to .Renviron.

Usage

```
set_bluesky_pass(pass, overwrite = FALSE, install = FALSE, r_env = NULL)
```

```
bs_set_pass(pass, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

pass	Character. App Password to add to add.
overwrite	Defaults to FALSE. Boolean. Should existing BLUESKY_APP_PASS in Renviron be overwritten?
install	Defaults to FALSE. Boolean. Should this be added '~/.Renviron' file?
r_env	Path to install to if install is TRUE.

Value

pass, invisibly

Examples

```
example_env <- tempfile(fileext = '.Renviron')
set_bluesky_pass('1234-1234-1234-1234', r_env = example_env)
# r_env should likely be: file.path(Sys.getenv('HOME'), '.Renviron')
```

set_bluesky_user *Adds Bluesky User to .Renviron.*

Description

Adds Bluesky User to .Renviron.

Usage

```
set_bluesky_user(user, overwrite = FALSE, install = FALSE, r_env = NULL)
```

```
bs_set_user(user, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

user	Character. User to add to add.
overwrite	Defaults to FALSE. Boolean. Should existing BLUESKY_APP_USER in Renviron be overwritten?
install	Defaults to FALSE. Boolean. Should this be added '~/.Renviron' file?
r_env	Path to install to if install is TRUE.

Value

user, invisibly

Examples

```
example_env <- tempfile(fileext = '.Renviron')
set_bluesky_user('CRAN_EXAMPLE.bsky.social', r_env = example_env)
# r_env should likely be: file.path(Sys.getenv('HOME'), '.Renviron')
```

user *Check or Get Bluesky User*

Description

Check or Get Bluesky User

Usage

```
has_bluesky_user()

get_bluesky_user()

bs_get_user()

bs_has_user()
```

Value

logical if has, user if get

Examples

```
has_bluesky_user()
```

Index

* actor

- [bs_get_actor_suggestions, 24](#)
- [bs_get_preferences, 51](#)
- [bs_get_profile, 52](#)
- [bs_search_actors, 75](#)

* auth

- [bs_auth, 6](#)
- [pass, 85](#)
- [set_bluesky_pass, 86](#)
- [set_bluesky_user, 86](#)
- [user, 87](#)

* chat

- [bs_accept_convo, 4](#)
- [bs_add_reaction, 5](#)
- [bs_delete_message_for_self, 15](#)
- [bs_get_convo, 28](#)
- [bs_get_convo_availability, 29](#)
- [bs_get_convo_for_members, 30](#)
- [bs_get_convo_log, 31](#)
- [bs_get_messages, 43](#)
- [bs_leave_convo, 60](#)
- [bs_list_convos, 62](#)
- [bs_mute_convo, 64](#)
- [bs_remove_reaction, 71](#)
- [bs_send_message, 77](#)
- [bs_send_message_batch, 78](#)
- [bs_unmute_convo, 80](#)
- [bs_update_all_read, 81](#)
- [bs_update_read, 82](#)

* embed

- [bs_new_embed_external, 65](#)

* feed

- [bs_get_author_feed, 25](#)
- [bs_get_feed, 32](#)
- [bs_get_feed_generator, 34](#)
- [bs_get_feed_generators, 35](#)
- [bs_get_feed_suggestions, 36](#)
- [bs_get_feeds, 33](#)
- [bs_get_likes, 40](#)

- [bs_get_list_feed, 42](#)
- [bs_get_post_likes, 49](#)
- [bs_get_post_thread, 50](#)
- [bs_get_posts, 48](#)
- [bs_get_quotes, 53](#)
- [bs_get_reposts, 56](#)
- [bs_get_timeline, 59](#)
- [bs_search_posts, 76](#)

* graph

- [bs_get_actor_lists, 22](#)
- [bs_get_actor_starter_packs, 23](#)
- [bs_get_blocked_lists, 26](#)
- [bs_get_blocks, 27](#)
- [bs_get_follow_suggestions, 39](#)
- [bs_get_followers, 37](#)
- [bs_get_follows, 38](#)
- [bs_get_list, 41](#)
- [bs_get_muted_lists, 44](#)
- [bs_get_mutes, 45](#)
- [bs_get_relationships, 55](#)
- [bs_get_starter_pack, 57](#)
- [bs_get_starter_packs, 58](#)

* helper

- [bs_created_at, 8](#)

* identity

- [bs_resolve_handle, 74](#)

* notification

- [bs_get_notification_count, 47](#)
- [bs_get_notifications, 46](#)

* record

- [bs_block, 7](#)
- [bs_follow, 21](#)
- [bs_like, 61](#)
- [bs_new_list, 66](#)
- [bs_new_list_item, 67](#)
- [bs_new_starter_pack, 68](#)
- [bs_post, 69](#)
- [bs_repost, 73](#)

* repo

- bs_create_record, 8
 - bs_delete_block, 9
 - bs_delete_follow, 11
 - bs_delete_like, 12
 - bs_delete_list, 13
 - bs_delete_list_item, 14
 - bs_delete_post, 16
 - bs_delete_record, 17
 - bs_delete_repost, 18
 - bs_delete_starter_pack, 19
 - bs_describe_repo, 20
 - bs_get_record, 54
 - bs_list_records, 63
 - bs_upload_blob, 83
-
- bs_accept_convo, 4
 - bs_add_reaction, 5
 - bs_auth, 6
 - bs_block, 7
 - bs_create_record, 8
 - bs_created_at, 8
 - bs_delete_block, 9
 - bs_delete_follow, 11
 - bs_delete_like, 12
 - bs_delete_list, 13
 - bs_delete_list_item, 14
 - bs_delete_message_for_self, 15
 - bs_delete_post, 16
 - bs_delete_record, 17
 - bs_delete_repost, 18
 - bs_delete_starter_pack, 19
 - bs_describe_repo, 20
 - bs_extract_record_key, 21
 - bs_follow, 21
 - bs_get_actor_lists, 22
 - bs_get_actor_starter_packs, 23
 - bs_get_actor_suggestions, 24
 - bs_get_author_feed, 25
 - bs_get_blocked_lists, 26
 - bs_get_blocks, 27
 - bs_get_convo, 28
 - bs_get_convo_availability, 29
 - bs_get_convo_for_members, 30
 - bs_get_convo_log, 31
 - bs_get_feed, 32
 - bs_get_feed_generator, 34
 - bs_get_feed_generators, 35
 - bs_get_feed_generators(), 34, 35
 - bs_get_feed_suggestions, 36
 - bs_get_feeds, 33
 - bs_get_follow_suggestions, 39
 - bs_get_followers, 37
 - bs_get_follows, 38
 - bs_get_likes, 40
 - bs_get_list, 41
 - bs_get_list_feed, 42
 - bs_get_messages, 43
 - bs_get_muted_lists, 44
 - bs_get_mutes, 45
 - bs_get_notification_count, 47
 - bs_get_notifications, 46
 - bs_get_pass (pass), 85
 - bs_get_post_likes, 49
 - bs_get_post_thread, 50
 - bs_get_posts, 48
 - bs_get_preferences, 51
 - bs_get_profile, 52
 - bs_get_quotes, 53
 - bs_get_record, 54
 - bs_get_relationships, 55
 - bs_get_reposts, 56
 - bs_get_starter_pack, 57
 - bs_get_starter_packs, 58
 - bs_get_timeline, 59
 - bs_get_user (user), 87
 - bs_has_pass (pass), 85
 - bs_has_user (user), 87
 - bs_leave_convo, 60
 - bs_like, 61
 - bs_list_convos, 62
 - bs_list_records, 63
 - bs_mute_convo, 64
 - bs_new_embed_external, 65
 - bs_new_list, 66
 - bs_new_list_item, 67
 - bs_new_starter_pack, 68
 - bs_post, 69
 - bs_remove_reaction, 71
 - bs_repost, 73
 - bs_resolve_handle, 74
 - bs_search_actors, 75
 - bs_search_posts, 76
 - bs_send_message, 77
 - bs_send_message_batch, 78
 - bs_set_pass (set_bluesky_pass), 86
 - bs_set_user (set_bluesky_user), 86
 - bs_unblock (bs_delete_block), 9

`bs_unfollow (bs_delete_follow)`, 11
`bs_unlike (bs_delete_like)`, 12
`bs_unmute_convo`, 80
`bs_update_all_read`, 81
`bs_update_read`, 82
`bs_upload_blob`, 83
`bs_uri_to_url`, 84
`bs_url_to_uri`, 84

`get_bluesky_pass (pass)`, 85
`get_bluesky_user (user)`, 87

`has_bluesky_pass (pass)`, 85
`has_bluesky_user (user)`, 87

`pass`, 85

`set_bluesky_pass`, 86
`set_bluesky_user`, 86

`tibble::tibble`, 4, 5, 7, 9, 15, 20, 22–45,
48–51, 53–64, 66, 68–70, 72–75,
77–83

`user`, 87