Package: bskyr (via r-universe)

October 4, 2026

Title Interact with 'Bluesky' Social

Version 0.5.0

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

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports cli, curl, dplyr, fs, httr2 (>= 1.0.0), mime, openssl, opengraph, purrr, rlang, stringi, stringr, tibble, tidyr

Suggests emoji, magick, 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

Config/Needs/website christopherkenny/ctktemplate

Language en-US

Depends R (>= 4.2.0)

VignetteBuilder knitr

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libxml2-dev libssl-dev

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

Date/Publication 2026-07-01 14:12:45 UTC

RemoteUrl https://github.com/christopherkenny/bskyr

RemoteRef HEAD

RemoteSha d6c54b891c85113e3834b77077f66be3bbab9ba6

Contents

appview . . . 4
bs_accept_convo . . . 4
bs_add_reaction . . . 5
bs_auth . . . 6
bs_block . . . 7
bs_create_record . . . 8
bs_created_at . . . 9
bs_delete_block . . . 10
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 . . . 33
bs_get_feed_generators . . . 34
bs_get_feed_suggestions . . . 35
bs_get_feeds . . . 36
bs_get_follow_suggestions . . . 37
bs_get_followers . . . 38
bs_get_follows . . . 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_notification_count . . . 46
bs_get_notifications . . . 47
bs_get_post_likes . . . 48
bs_get_post_thread . . . 49
bs_get_posts . . . 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_get_video_upload_limits . . . 60
bs_leave_convo . . . 61
bs_like . . . 62
bs_list_convos . . . 63
bs_list_records . . . 64
bs_mute_convo . . . 65
bs_new_embed_external . . . 66
bs_new_list . . . 67
bs_new_list_item . . . 68
bs_new_starter_pack . . . 69
bs_post . . . 70
bs_remove_reaction . . . 72
bs_repost . . . 74
bs_resolve_handle . . . 75
bs_search_actors . . . 76
bs_search_posts . . . 77
bs_send_message . . . 78
bs_send_message_batch . . . 79
bs_unmute_convo . . . 81
bs_update_all_read . . . 82
bs_update_read . . . 83
bs_upload_blob . . . 84
bs_upload_video . . . 85
bs_uri_to_url . . . 86
bs_url_to_uri . . . 86
pass . . . 87
pds . . . 88
set_bluesky_appview . . . 88
set_bluesky_pass . . . 89
set_bluesky_pds . . . 90
set_bluesky_user . . . 90

user 91

Index **92**

`appview` *Check or Get Bluesky AppView*

Description

Check or Get Bluesky AppView

Usage

```
has_bluesky_appview()

get_bluesky_appview()

bs_get_appview()

bs_has_appview()
```

Value

logical if `has`, AppView URL if `get`

Examples

```
has_bluesky_appview()
```

`bs_accept_convo` *Accept a conversation (direct message) invitation or request*

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 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.acceptConvo.json (2025-05-16)

Function introduced

v0.4.0 (2025-05-16)

Examples

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

---

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

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 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.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 *Authenticate a user*

Description

Authenticate a user

Usage

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

Arguments

user	Character. User name to log in with.
pass	Character. App password to log in with.
host	Character. Base URL of the user's Personal Data Server. Defaults to get_bluesky_pds().
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 *Block an account*

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 `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/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_create_record` *Create a record in a repo*

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_created_at *Get current time in Bluesky format*

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_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 `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/block.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 *Delete a follow (un-follow someone)*

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 `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/follow.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 *Delete a like (un-like something)*

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 `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

feed/like.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` *Delete a list*

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 `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/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

`convo_id`	Character, length 1. ID of the conversation to get.
`message_id`	Character, length 1. Message ID.
`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.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` *Delete a post*

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 `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

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$uri))
```

bs_delete_record *Delete a record in a repo*

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 `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

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` *Delete a repost*

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 `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

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 `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/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` *Describe a repo*

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 `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/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

`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 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 `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/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 `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/getSuggestions.json (2023-10-01)

Function introduced

`v0.0.1` (2023-10-01)

Examples

```
bs_get_actor_suggestions()
```

---

bs_get_author_feed *Retrieve posts on an actor's feed*

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 `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 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

cursor	Character, length 1. A cursor property from a prior response. Default: NULL.
limit	Integer. Maximum number to request.
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/getListMutes.json (2023-10-02)

Function introduced

`v0.0.1` (2023-10-02)

Examples

```
bs_get_blocked_lists()
```

`bs_get_blocks` *Retrieve user (self) blocks*

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 `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 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

convo_id	Character, length 1. ID of the conversation 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 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

`actors`	character vector of actor(s), 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 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

`actors`	character vector of actor(s), 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 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` *Retrieve the chat event log for the authenticated user*

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 `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.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

feed	Character, length 1. Feed 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 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_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 `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/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

`feeds`	Character. Vector of feeds 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 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_feeds` *Retrieve a list of feeds created by a given actor*

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_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_followers *Retrieve an actor's followers*

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 *Retrieve an actor's follows*

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_likes` *Retrieve posts liked by an actor (self)*

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 *Get a view of a list*

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 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 lists

Lexicon references

graph/getList.json (2025-03-20)

Function introduced

v0.2.0 (2024-11-25)

Examples

```
bs_get_list('at://did:plc:ragtjsm2j2vknwkz3zp4oxrd/app.bsky.graph.list/3kmokjyuflk2g')
bs_get_list('at://did:plc:hgyzg2hn6zxpqokmp5c2xrdo/app.bsky.graph.list/3laygnmmcfc2x')
```

`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:ragtjsm2j2vknwkz3zp4oxrd/app.bsky.graph.list/3kmokjyuflk2g')
```

`bs_get_messages` *Retrieve messages from a conversation, optionally paginated*

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 `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/getListMutes.json (2023-10-02)

Function introduced

`v0.0.1` (2023-10-02)

Examples

```
bs_get_muted_lists()
```

bs_get_mutes *Retrieve a user's (self) muted accounts*

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_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

`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 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_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_post_likes` *Retrieve likes on a post*

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 `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/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/3k7qmjev5lr2s')
```

`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 `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 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/3k7qmjev5lr2s')
```

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

uris	Character. Vector of URIs for posts 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 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/3k7qmjev5lr2s')
bs_get_posts('https://bsky.app/profile/chriskenny.bsky.social/post/3lc5d6zspys2c')
```

`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 `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 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

actors	character vector of actor(s), 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 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` *Retrieve a list of quotes for a given post*

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 `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 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:5c2r73erhng4bszmxlfdtscf/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 `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 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` *Retrieve actors who reposted a post*

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/3kaa2gxjhzr2a')
```

`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 `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/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/3lb3g5veo2z2r'
)
```

`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/3lb3g5veo2z2r'
)
bs_get_starter_packs(
  c(
    'at://did:plc:wpe35pganb6d4pg4ekmfy6u5/app.bsky.graph.starterpack/3lb3g5veo2z2r',
    'at://did:plc:bmc56x6ksb7o7sdkq2fgm7se/app.bsky.graph.starterpack/3laywns2q2v27'
  )
```

```
)
```

`bs_get_timeline` *Retrieve the user's home timeline*

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 `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 posts

Lexicon references

feed/getTimeline.json (2023-10-01)

Function introduced

`v0.0.1` (2023-10-02)

Examples

```
bs_get_timeline()
```

`bs_get_video_upload_limits` *Get Video Upload Limits*

Description

Get Video Upload Limits

Usage

```
bs_get_video_upload_limits(
  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 `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 video upload allowances, or a list if `clean = FALSE`

Lexicon references

video/getUploadLimits.json (2024-11-23)

Function introduced

`v0.5.0` (2026-06-01)

Examples

```
bs_get_video_upload_limits()
```

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

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 `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.leaveConvo.json (2025-05-16)

Function introduced

`v0.4.0` (2025-05-16)

Examples

```
bs_leave_convo(convo_id = '3lpidxucy2g27')
```

`bs_like` *Like an existing post*

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

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 *List records in a repo*

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 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

```
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 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.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 `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

a list representation of an external embed

Lexicon references

embed/external.json (2026-03-24)

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

name	Character. Display name for list.
purpose	Purpose of the list. One of 'modlist', 'curatelist', or 'referencelist'
description	Optional character. Description of the list.
avatar	Optional character. Path to image to use as avatar. PNG or JPEG recommended.
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 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` *Add a subject to a list*

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 `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 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 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 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 *Make a post on Bluesky Social*

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 :emoji: 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 bs_created_at().
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 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 match an earlier condidtion, 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 = '3lphbnrx7l32l', value = '\U0001F44D')
```

bs_repost *Repost an existing post*

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 `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 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` *Find profiles matching search criteria*

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` *Find posts matching search criteria*

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 or 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 *Send a message in a conversation (DM thread)*

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 `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.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 `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.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` *Unmute a previously muted conversation*

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

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 `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.updateRead.json (2025-05-16)

Function introduced

`v0.4.0` (2025-05-16)

Examples

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

`bs_upload_blob` *Upload a blob to a repo*

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 `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 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_upload_video` *Upload a video to be processed and stored*

Description

Upload a video to be processed and stored

Usage

```
bs_upload_video(
  video,
  user = get_bluesky_user(),
  pass = get_bluesky_pass(),
  auth = bs_auth(user, pass)
)
```

Arguments

`video`	Character, videos to upload to a repo.
`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

a blob reference list, compatible with `bs_post()`

Lexicon references

video/uploadVideo.json (2024-11-23) video/getJobStatus.json (2024-11-23)

Function introduced

`v0.5.0` (2026-06-01)

Examples

```
vid <- fs::path_package('bskyr', 'man/figures/pkgs.mp4')
bs_upload_video(vid)
```

bs_uri_to_url *Convert Universal Resource Identifiers to Hypertext Transfer Protocol Secure URLs*

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 *Convert Hypertext Transfer Protocol Secure URLs to Universal Resource Identifiers*

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 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

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()
```

pds *Check or Get Bluesky PDS*

Description

Check or Get Bluesky PDS

Usage

```
has_bluesky_pds()

get_bluesky_pds()

bs_get_pds()

bs_has_pds()
```

Value

logical if has, PDS URL if get

Examples

```
has_bluesky_pds()
```

set_bluesky_appview *Adds Bluesky AppView to .Renviron.*

Description

Adds Bluesky AppView to .Renviron.

Usage

```
set_bluesky_appview(appview, overwrite = FALSE, install = FALSE, r_env = NULL)

bs_set_appview(appview, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

appview	Character. Base URL of the AppView server to add.
overwrite	Defaults to FALSE. Boolean. Should existing BLUESKY_APP_APPVIEW 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

appview, invisibly

Examples

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

`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_pds *Adds Bluesky PDS to .Renviron.*

Description

Adds Bluesky PDS to .Renviron.

Usage

```
set_bluesky_pds(pds, overwrite = FALSE, install = FALSE, r_env = NULL)

bs_set_pds(pds, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

pds	Character. Base URL of the Personal Data Server to add.
overwrite	Defaults to FALSE. Boolean. Should existing BLUESKY_APP_PDS 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

pds, invisibly

Examples

```
example_env <- tempfile(fileext = '.Renviron')
set_bluesky_pds('https://bsky.social', 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, 76
∗ **auth**
appview, 4
bs_auth, 6
pass, 87
pds, 88
set_bluesky_appview, 88
set_bluesky_pass, 89
set_bluesky_pds, 90
set_bluesky_user, 90
user, 91
∗ **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, 61
bs_list_convos, 63
bs_mute_convo, 65
bs_remove_reaction, 72
bs_send_message, 78
bs_send_message_batch, 79
bs_unmute_convo, 81
bs_update_all_read, 82
bs_update_read, 83
∗ **embed**
bs_new_embed_external, 66
∗ **feed**
bs_get_author_feed, 25
bs_get_feed, 32
bs_get_feed_generator, 33
bs_get_feed_generators, 34
bs_get_feed_suggestions, 35
bs_get_feeds, 36
bs_get_likes, 40
bs_get_list_feed, 42
bs_get_post_likes, 48
bs_get_post_thread, 49
bs_get_posts, 50
bs_get_quotes, 53
bs_get_reposts, 56
bs_get_timeline, 59
bs_search_posts, 77
∗ **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, 37
bs_get_followers, 38
bs_get_follows, 39
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, 9
∗ **identity**
bs_resolve_handle, 75
∗ **notification**
bs_get_notification_count, 46
bs_get_notifications, 47
∗ **record**
bs_block, 7
bs_follow, 21
bs_like, 62
bs_new_list, 67
bs_new_list_item, 68

bs_new_starter_pack, 69
bs_post, 70
bs_repost, 74
∗ **repo**
bs_create_record, 8
bs_delete_block, 10
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, 64
bs_upload_blob, 84
bs_upload_video, 85
∗ **video**
bs_get_video_upload_limits, 60

appview, 4

bs_accept_convo, 4
bs_add_reaction, 5
bs_auth, 6
bs_block, 7
bs_create_record, 8
bs_created_at, 9
bs_delete_block, 10
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_appview *(*appview*)*, 4
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, 33
bs_get_feed_generators, 34
bs_get_feed_generators(), *33, 34*
bs_get_feed_suggestions, 35
bs_get_feeds, 36
bs_get_follow_suggestions, 37
bs_get_followers, 38
bs_get_follows, 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_notification_count, 46
bs_get_notifications, 47
bs_get_pass *(*pass*)*, 87
bs_get_pds *(*pds*)*, 88
bs_get_post_likes, 48
bs_get_post_thread, 49
bs_get_posts, 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_get_user *(*user*)*, 91
bs_get_video_upload_limits, 60
bs_has_appview *(*appview*)*, 4
bs_has_pass *(*pass*)*, 87
bs_has_pds *(*pds*)*, 88
bs_has_user *(*user*)*, 91
bs_leave_convo, 61
bs_like, 62
bs_list_convos, 63
bs_list_records, 64
bs_mute_convo, 65
bs_new_embed_external, 66

bs_new_list, 67
bs_new_list_item, 68
bs_new_starter_pack, 69
bs_post, 70
bs_remove_reaction, 72
bs_repost, 74
bs_resolve_handle, 75
bs_search_actors, 76
bs_search_posts, 77
bs_send_message, 78
bs_send_message_batch, 79
bs_set_appview *(*set_bluesky_appview*)*, 88
bs_set_pass *(*set_bluesky_pass*)*, 89
bs_set_pds *(*set_bluesky_pds*)*, 90
bs_set_user *(*set_bluesky_user*)*, 90
bs_unblock *(*bs_delete_block*)*, 10
bs_unfollow *(*bs_delete_follow*)*, 11
bs_unlike *(*bs_delete_like*)*, 12
bs_unmute_convo, 81
bs_update_all_read, 82
bs_update_read, 83
bs_upload_blob, 84
bs_upload_video, 85
bs_uri_to_url, 86
bs_url_to_uri, 86

get_bluesky_appview *(*appview*)*, 4
get_bluesky_pass *(*pass*)*, 87
get_bluesky_pds *(*pds*)*, 88
get_bluesky_user *(*user*)*, 91

has_bluesky_appview *(*appview*)*, 4
has_bluesky_pass *(*pass*)*, 87
has_bluesky_pds *(*pds*)*, 88
has_bluesky_user *(*user*)*, 91

pass, 87
pds, 88

set_bluesky_appview, 88
set_bluesky_pass, 89
set_bluesky_pds, 90
set_bluesky_user, 90

tibble::tibble, *5–8*, *15*, *20*, *22–45*, *48–51*, *53–65*, *67*, *69–71*, *73–76*, *78–84*

user, 91