

---

# **narwal Documentation**

***Release 0.3.2b***

**Larry Ng**

June 04, 2012



# CONTENTS

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Examples</b>	<b>5</b>
<b>3</b>	<b>Documentation</b>	<b>7</b>
3.1	Usage . . . . .	7
3.2	Changelog . . . . .	23
	<b>Python Module Index</b>	<b>25</b>



Version 0.3 (beta).

`narwal` (sic) is a Python wrapper for reddit's API made to be simple, intuitive, and concise, i.e. *pythonic*.

```
>>> import narwal
>>> session = narwal.connect('narwal_bot', 'hunter2', user_agent='i'm a narw(h)al!')
>>> frontpage = session.hot()
>>> for link in frontpage[:3]:
...     print link
...
(3088) Words can not describe how much I love this pic of Obama and Clinton
(1697) Rough day for a mom at the airport.
(1370) I felt awful when this happened.
>>> frontpage[1].upvote()
<Response [200]>
>>> frontpage[1].comment('cool story bro')
<Comment [narwal_bot: cool story bro]>
```

Works with Python 2.7. See the [source on github](#). Written by Larry Ng and released under the ISC License.



# INSTALLATION

To install, just do the usual:

```
$ pip install narwal
```





# EXAMPLES

Start a session:

```
>>> import narwal
>>> session = narwal.connect(user_agent='narwal demo')
```

Start a logged in session:

```
>>> session = narwal.connect('narwal_bot', 'password', user_agent='narwal demo')
```

Get the front page:

```
>>> page1 = session.hot()
```

Get the next page:

```
>>> page2 = page1.more()
```

Get the fourth link's comments:

```
>>> comments = page1[3].comments()
```

Get the second link of r/test/top:

```
>>> link = session.top('test')[1]
```

Submit a comment to it:

```
>>> comment = link.comment('the narwhal ba--')
```

Downvote the comment we just submitted:

```
>>> comment.downvote()
<Response [200]>
```

And reply to it:

```
>>> comment.reply('NO! *slap*')
<Comment [(1) narwal_bot: NO! *slap*]>
```

Check our inbox:

```
>>> inbox = session.inbox()
```

Read the first message:

```
>>> print inbox[0]
larryng: hi there
```

Get the sender's account info and submissions:

```
>>> user = session.user(message.author)
>>> user.submitted()
[<Link [(1) test post please ...]>]
```

Plus a whole lot more, since most of the reddit API has been implemented. See the API docs (or the source) for more features.

# DOCUMENTATION

## 3.1 Usage

After importing narwal,

```
>>> import narwal
```

you'll always be starting with `Reddit`, by invoking either:

```
>>> session = narwal.Reddit(user_agent='hellonarwal')
>>> session = narwal.Reddit('username', 'password', user_agent='nice2meetu')
```

or

```
>>> session = narwal.connect(user_agent='hellonarwal')
>>> session = narwal.connect('username', 'password', user_agent='nice2meetu')
```

But really, `connect()` is the same as instantiating `Reddit`. I just think `connect()` makes more sense intuitively. `narwal` defaults to respecting reddit's rules of making at most only 1 request every 2 seconds and always supplying a useful User-Agent. That's why if you try this:

```
>>> session = narwal.connect()
```

a `ValueError` will be raised complaining that you need to supply a `user_agent`. You can make `narwal` be disrespectful by setting `respect=False` when instantiating a new session:

```
>>> session = narwal.connect(respect=False)
```

But c'mon – be respectful.

Upon receiving a response from GET (and some POST) requests, `narwal` attempts to “thingify” the response content. Thingifying is simply wrapping the useful data as a `things.Thing` subclass, as defined by reddit [here](#). Thingifying works recursively through the object, such that some data in a `Thing` may be another `Thing`. For example:

```
>>> page1 = session.hot()           # Listing
>>> link = page1[0]                 # Link

>>> comments = link.comments()      # Listing
>>> comment = comments[0]           # Comment
>>> replies = comment.replies       # Listing
>>> replies[0]                     # Comment
```

You can access all of `narwal`'s implemented reddit API calls through `narwal.Reddit` methods, but, as you can see in the examples, many of them are accessible through `things`' methods for convenience.

### 3.1.1 narwal

`narwal.connect(*args, **kwargs)`

Just an alias to instantiate `Reddit`, really.

**class** `narwal.Reddit` (*username=None, password=None, user\_agent=None, respect=True*)

A Reddit session.

#### Parameters

- **username** – (optional) reddit username
- **password** – (optional) reddit password
- **user\_agent** – User-Agent
- **respect** (*True or False*) – If True, requires `user_agent` to be specified and limits request rate to 1 every 2 seconds, as per reddit's API rules.

#### `logged_in`

Property. True if logged in.

**get** (*\*args, \*\*kwargs*)

Sends a GET request to a reddit path determined by `args`. Basically `.get('foo', 'bar', 'baz')` will GET <http://www.reddit.com/foo/bar/baz.json>. `kwargs` supplied will be passed to `requests.get()` after having `user_agent` and `cookies` injected. Injection only occurs if they don't already exist.

Returns `things.Blob` object or a subclass of `things.Blob`, or raises `exceptions.BadResponse` if not a 200 Response.

#### Parameters

- **\*args** – strings that will form the path to GET
- **\*\*kwargs** – extra keyword arguments to be passed to `requests.get()`

**post** (*\*args, \*\*kwargs*)

Sends a POST request to a reddit path determined by `args`. Basically `.post('foo', 'bar', 'baz')` will POST <http://www.reddit.com/foo/bar/baz.json>. `kwargs` supplied will be passed to `requests.post` after having `modhash` and `cookies` injected, and after having `modhash` injected into `kwargs['data']` if logged in. Injection only occurs if they don't already exist.

Returns received response JSON content as a dict.

Raises `exceptions.BadResponse` if not a 200 response or no JSON content received or raises `exceptions.PostError` if a reddit error was returned.

#### Parameters

- **\*args** – strings that will form the path to POST
- **\*\*kwargs** – extra keyword arguments to be passed to `requests.POST`

**login** (*username, password*)

Logs into reddit with supplied credentials using SSL. Returns `requests.Response` object, or raises `exceptions.LoginFail` or `exceptions.BadResponse` if not a 200 response.

URL: <https://ssl.reddit.com/api/login>

#### Parameters

- **username** – reddit username
- **password** – corresponding reddit password

**by\_id** (*id\_*)

GETs a link by ID. Returns `things.Link` object.

URL: `http://www.reddit.com/by_id/<id_>`

**Parameters** *id\_* – full name of link

**hot** (*sr=None, limit=None*)

GETs hot links. If *sr* is None, gets from main. Returns `things.Listing` object.

URL: `http://www.reddit.com/[r/<sr>]/?limit=<limit>`

**Parameters**

- **sr** – subreddit name
- **limit** – max number of submissions to get

**new** (*sr=None, limit=None*)

GETs new links. If *sr* is None, gets from main. Returns `things.Listing` object.

URL: `http://www.reddit.com/[r/<sr>/]new/?limit=<limit>`

**Parameters**

- **sr** – subreddit name
- **limit** – max number of submissions to get

**top** (*sr=None, limit=None*)

GETs top links. If *sr* is None, gets from main. Returns `things.Listing` object.

URL: `http://www.reddit.com/[r/<sr>/]top/?limit=<limit>`

**Parameters**

- **sr** – subreddit name
- **limit** – max number of submissions to get

**controversial** (*sr=None, limit=None*)

GETs controversial links. If *sr* is None, gets from main. Returns `things.Listing` object.

URL: `http://www.reddit.com/[r/<sr>/]controversial/?limit=<limit>`

**Parameters**

- **sr** – subreddit name
- **limit** – max number of submissions to get

**comments** (*sr=None, limit=None*)

GETs newest comments. If *sr* is None, gets all. Returns `things.Listing` object.

URL: `http://www.reddit.com/[r/<sr>/]comments/?limit=<limit>`

**Parameters**

- **sr** – subreddit name
- **limit** – max number of comments to get

**user** (*username*)

GETs user info. Returns `things.Account` object.

URL: `http://www.reddit.com/user/<username>/about/`

**Parameters** *username* – username of user

**subreddit** (*sr*)

GETs subreddit info. Returns `things.Subreddit` object.

URL: `http://www.reddit.com/r/<sr>/about/`

**Parameters** *sr* – subreddit name

**info** (*url, limit=None*)

GETs “info” about url. See <https://github.com/reddit/reddit/wiki/API%3A-info.json>.

URL: `http://www.reddit.com/api/info/?url=<url>`

**Parameters**

- **url** – url
- **limit** – max number of links to get

**search** (*query, limit=None*)

Use reddit’s search function. Returns `things.Listing` object.

URL: `http://www.reddit.com/search/?q=<query>&limit=<limit>`

**Parameters**

- **query** – query string
- **limit** – max number of results to get

**domain** (*domain\_, limit=None*)

GETs links from domain\_. Returns `things.Listing` object.

URL: `http://www.reddit.com/domain/?domain=<domain_>&limit=<limit>`

**Parameters**

- **domain** – the domain, e.g. `google.com`
- **limit** – max number of links to get

**user\_overview** (*user, limit=None*)

GETs a user’s posted comments. Returns `things.Listing` object.

**Parameters**

- **user** – reddit username
- **limit** – max number of comments to return

**user\_comments** (*user, limit=None*)

GETs a user’s posted comments. Returns `things.Listing` object.

**Parameters**

- **user** – reddit username
- **limit** – max number of comments to return

**user\_submitted** (*user, limit=None*)

GETs a user’s submissions. Returns `things.Listing` object.

**Parameters**

- **user** – reddit username
- **limit** – max number of submissions to return

**moderators** (*sr*, *limit=None*)

GETs moderators of subreddit *sr*. Returns `things.ListBlob` object.

**NOTE:** The `things.Account` objects in the returned ListBlob *only* have `id` and `name` set. This is because that's all reddit returns. If you need full info on each moderator, you must individually GET them using `user()` or `things.Account.about()`.

URL: `http://www.reddit.com/r/<sr>/about/moderators/`

**Parameters** *sr* – name of subreddit

**me** (*\*args*, *\*\*kwargs*)

Login required. GETs info about logged in user. Returns `:class'things.Account'` object.

See <https://github.com/reddit/reddit/wiki/API%3A-mine.json>.

URL: `http://www.reddit.com/api/me/`

**mine** (*\*args*, *\*\*kwargs*)

Login required. GETs logged in user's subreddits. Returns `things.Listing` object.

See <https://github.com/reddit/reddit/wiki/API%3A-mine.json>.

URL: `http://www.reddit.com/reddits/mine[/ (subscriber|contributor|moderator) ]?limit=`

**Parameters**

- **which** – 'subscriber', 'contributor', or 'moderator'
- **limit** – max number of subreddits to get

**saved** (*\*args*, *\*\*kwargs*)

Login required. GETs logged in user's saved submissions. Returns `things.Listing` object.

URL: `http://www.reddit.com/saved/`

**Parameters** *limit* – max number of submissions to get

**vote** (*\*args*, *\*\*kwargs*)

Login required. POSTs a vote. Returns True or raises `exceptions.UnexpectedResponse` if non-"truthy" value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-vote>.

URL: `http://www.reddit.com/api/vote/`

**Parameters**

- **id\_** – full id of object voting on
- **dir\_** – direction of vote (1, 0, or -1)

**upvote** (*\*args*, *\*\*kwargs*)

Login required. POSTs an upvote (1). Returns True or raises `exceptions.UnexpectedResponse` if non-"truthy" value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-vote>.

URL: `http://www.reddit.com/api/vote/`

**Parameters** *id\_* – full id of object voting on

**downvote** (*\*args*, *\*\*kwargs*)

Login required. POSTs a downvote (-1). Returns True or raises `exceptions.UnexpectedResponse` if non-"truthy" value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-vote>.

URL: `http://www.reddit.com/api/vote/`

**Parameters** `id_` – full id of object voting on

**unvote** (*\*args*, *\*\*kwargs*)

Login required. POSTs a null vote (0). Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-vote>.

URL: `http://www.reddit.com/api/vote/`

**Parameters** `id_` – full id of object voting on

**comment** (*\*args*, *\*\*kwargs*)

Login required. POSTs a comment in response to parent. Returns `things.Comment` object.

See <https://github.com/reddit/reddit/wiki/API%3A-comment>.

URL: `http://www.reddit.com/api/comment/`

**Parameters**

- **parent** – full id of thing commenting on
- **text** – comment text

**edit** (*\*args*, *\*\*kwargs*)

Login required. Sends POST to change selftext or comment text to `text`. Returns `things.Comment` or `things.Link` object depending on what’s being edited. Raises `UnexpectedResponse` if neither is returned.

URL: `http://www.reddit.com/api/editusertext/`

**Parameters**

- **id\_** – full id of link or comment to edit
- **text** – new self or comment text

**submit\_link** (*\*args*, *\*\*kwargs*)

Login required. POSTs a link submission. Returns `things.Link` object if `follow=True` (default), or the string permalink of the new submission otherwise.

Argument `follow` exists because reddit only returns the permalink after POSTing a submission. In order to get detailed info on the new submission, we need to make another request. If you don’t want to make that additional request, just set `follow=False`.

See <https://github.com/reddit/reddit/wiki/API%3A-submit>.

URL: `http://www.reddit.com/api/submit/`

**Parameters**

- **sr** – name of subreddit to submit to
- **title** – title of submission
- **url** – submission link
- **follow** (*bool*) – set to True to follow retrieved permalink to return detailed `things.Link` object. False to just return permalink.

**submit\_text** (*\*args*, *\*\*kwargs*)

Login required. POSTs a text submission. Returns `things.Link` object if `follow=True` (default), or the string permalink of the new submission otherwise.



Argument `follow` exists because reddit only returns the permalink after POSTing a submission. In order to get detailed info on the new submission, we need to make another request. If you don't want to make that additional request, set `follow=False`.

See <https://github.com/reddit/reddit/wiki/API%3A-submit>.

URL: `http://www.reddit.com/api/submit/`

#### Parameters

- **sr** – name of subreddit to submit to
- **title** – title of submission
- **text** – submission self text
- **follow** (*bool*) – set to `True` to follow retrieved permalink to return detailed `things.Link` object. `False` to just return permalink.

**delete** (*\*args, \*\*kwargs*)

Login required. Send POST to delete an object. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/del/`

**Parameters** **id\_** – full id of object to delete

**save** (*\*args, \*\*kwargs*)

Login required. Sends POST to save a link. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-save>.

URL: `http://www.reddit.com/api/save/`

**Parameters** **id\_** – full id of link to save

**unsave** (*\*args, \*\*kwargs*)

Login required. Sends POST to unsave a link. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-unsave>.

URL: `http://www.reddit.com/api/unsave/`

**Parameters** **id\_** – full id of link to unsave

**hide** (*\*args, \*\*kwargs*)

Login required. Sends POST to hide a link. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-hide>.

URL: `http://www.reddit.com/api/hide/`

**Parameters** **id\_** – full id of link to hide

**unhide** (*\*args, \*\*kwargs*)

Login required. Sends POST to unhide a link. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

See <https://github.com/reddit/reddit/wiki/API%3A-unhide>.

URL: `http://www.reddit.com/api/unhide/`

**Parameters** **id\_** – full id of link to unhide

**marknsfw** (\*args, \*\*kwargs)

Login required. Sends POST to mark link as NSFW. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/marknsfw/`

**Parameters** `id_` – full id of link to mark

**unmarknsfw** (\*args, \*\*kwargs)

Login required. Sends POST to unmark link as NSFW. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/unmarknsfw/`

**Parameters** `id_` – full id of link to unmark

**report** (\*args, \*\*kwargs)

Login required. Sends POST to report a link. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/report/`

**Parameters** `id_` – full id of link to report

**message** (\*args, \*\*kwargs)

Alias for `compose()`.

**compose** (\*args, \*\*kwargs)

Login required. Sends POST to send a message to a user. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/compose/`

**Parameters**

- **to** – username or `:class‘things.Account‘` of user to send to
- **subject** – subject of message
- **text** – message body text

**read\_message** (\*args, \*\*kwargs)

Login required. Send POST to mark a message as read. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/read_message/`

**Parameters** `id_` – full id of message to mark

**unread\_message** (\*args, \*\*kwargs)

Login required. Send POST to unmark a message as read. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/unread_message/`

**Parameters** `id_` – full id of message to unmark

**subscribe** (\*args, \*\*kwargs)

Login required. Send POST to subscribe to a subreddit. If `sr` is the name of the subreddit, a GET request is sent to retrieve the full id of the subreddit, which is necessary for this API call. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/subscribe/`

**Parameters** `sr` – full id of subreddit or name of subreddit (full id is preferred)

**unsubscribe** (\*args, \*\*kwargs)

Login required. Send POST to unsubscribe to a subreddit. If `sr` is the name of the subreddit, a GET request is sent to retrieve the full id of the subreddit, which is necessary for this API call. Returns `True` or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/unsubscribe/`

**Parameters** `sr` – full id of subreddit or name of subreddit (full id is preferred)

**inbox** (\*args, \*\*kwargs)

Login required. GETs logged in user’s inbox. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/inbox/`

**Parameters** `limit` – max number of objects to get

**unread** (\*args, \*\*kwargs)

Login required. GETs logged in user’s unread. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/unread/`

**Parameters** `limit` – max number of objects to get

**messages** (\*args, \*\*kwargs)

Login required. GETs logged in user’s messages. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/messages/`

**Parameters** `limit` – max number of messages to get

**commentreplies** (\*args, \*\*kwargs)

Login required. GETs logged in user’s comment replies. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/comments/`

**Parameters** `limit` – max number of comment replies to get

**postreplies** (\*args, \*\*kwargs)

Login required. GETs logged in user’s post replies. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/selfreply/`

**Parameters** `limit` – max number of post replies to get

**sent** (\*args, \*\*kwargs)

Login required. GETs logged in user’s sent messages. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/sent/`

**Parameters** `limit` – max number of messages to get

**modmail** (\*args, \*\*kwargs)

Login required. GETs logged in user’s modmail. Returns `things.Listing` object.

URL: `http://www.reddit.com/message/moderator/`

**Parameters** `limit` – max number of messages to get

**liked** (\*args, \*\*kwargs)

GETs logged-in user’s liked submissions. Returns `things.Listing` object.

**Parameters** `limit` – max number of submissions to get

**disliked** (\*args, \*\*kwargs)

GETs logged-in user’s disliked submissions. Returns `things.Listing` object.

**Parameters** `limit` – max number of submissions to get

**hidden** (\*args, \*\*kwargs)

GETs logged-in user's hidden submissions. Returns `things.Listing` object.

**Parameters** **limit** – max number of submissions to get

**approve** (\*args, \*\*kwargs)

Login required. Sends POST to approve a submission. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/approve/`

**Parameters** **id\_** – full id of submission to approve

**remove** (\*args, \*\*kwargs)

Login required. Sends POST to remove a submission or comment. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/remove/`

**Parameters** **id\_** – full id of object to remove

**distinguish** (\*args, \*\*kwargs)

Login required. Sends POST to distinguish a submission or comment. Returns `things.Link` or `things.Comment`, or raises `exceptions.UnexpectedResponse` otherwise.

URL: `http://www.reddit.com/api/distinguish/`

**Parameters**

- **id\_** – full id of object to distinguish
- **how** – either True, False, or ‘admin’

**flairlist** (\*args, \*\*kwargs)

Login required. Gets flairlist for subreddit *r*. See <https://github.com/reddit/reddit/wiki/API%3A-flairlist>.

However, the wiki docs are wrong (as of 2012/5/4). Returns `things.ListBlob` of `things.Blob` objects, each object being a mapping with *user*, *flair\_css\_class*, and *flair\_text* attributes.

URL: `http://www.reddit.com/r/<r>/api/flairlist`

**Parameters**

- **r** – name of subreddit
- **limit** – max number of items to return
- **after** – full id of user to return entries after
- **before** – full id of user to return entries *before*

**flair** (\*args, \*\*kwargs)

Login required. Sets flair for a user. See <https://github.com/reddit/reddit/wiki/API%3A-flair>. Returns True or raises `exceptions.UnexpectedResponse` if non-“truthy” value in response.

URL: `http://www.reddit.com/api/flair`

**Parameters**

- **r** – name of subreddit
- **name** – name of the user
- **text** – flair text to assign
- **css\_class** – CSS class to assign to flair text

**flaircsv** (\*args, \*\*kwargs)

Login required. Bulk sets flair for users. See <https://github.com/reddit/reddit/wiki/API%3A-flaircsv/>. Returns response JSON content as dict.

URL: <http://www.reddit.com/api/flaircsv>

#### Parameters

- **r** – name of subreddit
- **flair\_csv** – csv string

**contributors** (\*args, \*\*kwargs)

Login required. GETs list of contributors to subreddit `sr`. Returns `things.ListBlob` object.

**NOTE:** The `things.Account` objects in the returned ListBlob *only* have `id` and `name` set. This is because that's all reddit returns. If you need full info on each contributor, you must individually GET them using `user()` or `things.Account.about()`.

URL: <http://www.reddit.com/r/<sr>/about/contributors/>

**Parameters** **sr** – name of subreddit

## 3.1.2 narwal.things

**class** `narwal.things.Blob` (`reddit`)

Bases: `object`

A dumb container because `obj.x` is cooler then `obj['x']`.

**Parameters** **reddit** (`Reddit`) – a reddit session

**class** `narwal.things.ListBlob` (`reddit`, `items=None`)

Bases: `narwal.things.Blob`

A `Blob` that almost looks and feels like a `list`, but isn't.

#### Parameters

- **reddit** (`Reddit`) – a reddit session
- **items** – initial list to absorb

**class** `narwal.things.Thing` (\*args, \*\*kwargs)

Bases: `narwal.things.Blob`

A reddit Thing. See <https://github.com/reddit/reddit/wiki/thing> for more details. You will only be seeing instances of subclasses of this.

*kind* is omitted because it will be implied by `Thing`'s subclasses. *data* is omitted because data will be stored as attributes.

**class** `narwal.things.Created` (\*args, \*\*kwargs)

Bases: `narwal.things.Thing`

An implementation. See <https://github.com/reddit/reddit/wiki/thing> for more details.

**class** `narwal.things.Votable` (\*args, \*\*kwargs)

Bases: `narwal.things.Thing`

An implementation. See <https://github.com/reddit/reddit/wiki/thing> for more details.

**vote** (`dir_`)

POST a vote on the thing. Calls `narwal.Reddit.vote()`.

**Parameters** `dir_` – direction (up: 1, down: -1, no vote: 0)

**upvote()**  
Upvote the thing (POST). Calls `Votable.vote()`.

**downvote()**  
Downvote the thing (POST). Calls `Votable.vote()`.

**unvote()**  
Unvote the thing (POST). Calls `Votable.vote()`.

**class** `narwal.things.Commentable(*args, **kwargs)`  
Bases: `narwal.things.Thing`  
Base class for `Thing` objects that are commentable (i.e. `Link` and `Comment`).

**comment(text)**  
POST a comment to this thing. Calls `narwal.Reddit.comment()`.

**Parameters** `text` – comment’s body text

**edit(text)**  
Edits this thing (POST). Calls `narwal.Reddit.edit()`.

**Parameters** `text` – new text

**comments(limit=None)**  
GETs comments to this thing.

**Parameters** `limit` – max number of comments to return

**distinguish(how=True)**  
Distinguishes this thing (POST). Calls `narwal.Reddit.distinguish()`.

**Parameters** `how` – either `True`, `False`, or ‘admin’

**delete()**  
Deletes this thing (POST). Calls `narwal.Reddit.delete()`.

**remove()**  
Removes this thing (POST). Calls `narwal.Reddit.remove()`.

**class** `narwal.things.Hideable(*args, **kwargs)`  
Bases: `narwal.things.Thing`  
Base class for `Thing` objects that are hideable (i.e. `Link` and `Message`).

**hide()**  
Hides this thing (POST). Calls `narwal.Reddit.hide()`.

**unhide()**  
Hides this thing (POST). Calls `narwal.Reddit.unhide()`.

**class** `narwal.things.Reportable(*args, **kwargs)`  
Bases: `narwal.things.Thing`  
Base class for `Thing` objects that are reportable (i.e. `Link`, `Comment`, and `Message`).

**report()**  
Reports this thing (POST). Calls `narwal.Reddit.report()`.

**class** `narwal.things.Listing(*args, **kwargs)`  
Bases: `narwal.things.ListBlob`  
A reddit `Listing`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

data is omitted because data will be stored as attributes.

#### **children**

Property that aliases `self.children` to `self._items`. This is done so that `ListBlob` special methods work automagically.

#### **has\_more**

Returns True if there are more things that can be retrieved.

#### **more** (*limit=None*)

A convenience function. Calls `self.next_listing`.

#### **next\_listing** (*limit=None*)

GETs next `Listing` directed to by this `Listing`. Returns `Listing` object.

**Parameters** `limit` – max number of entries to get

**Raises** `UnsupportedError` raised when trying to load more comments

#### **prev\_listing** (*limit=None*)

GETs previous `Listing` directed to by this `Listing`. Returns `Listing` object.

**Parameters** `limit` – max number of entries to get

#### **class** `narwal.things.Comment` (*\*args, \*\*kwargs*)

Bases: `narwal.things.Votable`, `narwal.things.Created`, `narwal.things.Commentable`, `narwal.things.Reportable`

A reddit `Comment`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

#### **permalink**

Property. Returns permalink as relative path.

#### **reply** (*text*)

POSTs a comment in reply to this one. Calls `Commentable.comment()`.

**Parameters** `text` – comment body text

#### **class** `narwal.things.Link` (*\*args, \*\*kwargs*)

Bases: `narwal.things.Votable`, `narwal.things.Created`, `narwal.things.Commentable`, `narwal.things.Hideable`, `narwal.things.Reportable`

A reddit `Link`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

#### **save** ()

Saves this link (POST). Calls `narwal.Reddit.save()`.

#### **unsave** ()

Unsave this link (POST). Calls `narwal.Reddit.unsave()`.

#### **marknsfw** ()

Marks link as nsfw (POST). Calls `narwal.Reddit.marknsfw()`.

#### **unmarknsfw** ()

Marks link as nsfw (POST). Calls `narwal.Reddit.unmarknsfw()`.

#### **approve** ()

Approves this link (POST). Calls `narwal.Reddit.approve()`.

#### **refresh** ()

Re-GETs this link (does not alter the object). Returns `Link` object.

#### **class** `narwal.things.Subreddit` (*\*args, \*\*kwargs*)

Bases: `narwal.things.Thing`

A reddit Submission. See <https://github.com/reddit/reddit/wiki/thing> for more details.

**hot** (*limit=None*)

GETs hot links from this subreddit. Calls `narwal.Reddit.hot()`.

**Parameters** **limit** – max number of links to return

**new** (*limit=None*)

GETs new links from this subreddit. Calls `narwal.Reddit.new()`.

**Parameters** **limit** – max number of links to return

**top** (*limit=None*)

GETs top links from this subreddit. Calls `narwal.Reddit.top()`.

**Parameters** **limit** – max number of links to return

**controversial** (*limit=None*)

GETs controversial links from this subreddit. Calls `narwal.Reddit.controversial()`.

**Parameters** **limit** – max number of links to return

**comments** (*limit=None*)

GETs newest comments from this subreddit. Calls `narwal.Reddit.comments()`.

**Parameters** **limit** – max number of links to return

**subscribe** ()

Subscribe to this subreddit (POST). Calls `narwal.Reddit.subscribe()`.

**unsubscribe** ()

Unsubscribe to this subreddit (POST). Calls `narwal.Reddit.unsubscribe()`.

**submit\_link** (*title, url*)

Submit link to this subreddit (POST). Calls `narwal.Reddit.submit_link()`.

**Parameters**

- **title** – title of submission
- **url** – url submission links to

**submit\_text** (*title, text*)

Submit self text submission to this subreddit (POST). Calls `narwal.Reddit.submit_text()`.

**Parameters**

- **title** – title of submission
- **text** – self text

**moderators** (*limit=None*)

GETs moderators for this subreddit. Calls `narwal.Reddit.moderators()`.

**Parameters** **limit** – max number of items to return

**flair** (*name, text, css\_class*)

Sets flair for *user* in this subreddit (POST). Calls `narwal.Reddit.flairlist()`.

**Parameters**

- **name** – name of the user
- **text** – flair text to assign
- **css\_class** – CSS class to assign to flair text



**flairlist** (*limit=1000, after=None, before=None*)

GETs flairlist for this subreddit. Calls `narwal.Reddit.flairlist()`.

**Parameters**

- **limit** – max number of items to return
- **after** – full id of user to return entries after
- **before** – full id of user to return entries *before*

**flaircsv** (*flair\_csv*)

Bulk sets flair for users in this subreddit (POST). Calls `narwal.Reddit.flaircsv()`.

**Parameters** **flair\_csv** – CSV string

**contributors** (*limit=None*)

GETs contributors for this subreddit. Calls `narwal.Reddit.contributors()`.

**Parameters** **limit** – max number of items to return

**class** `narwal.things.Message` (*\*args, \*\*kwargs*)

Bases: `narwal.things.Created`, `narwal.things.Hideable`, `narwal.things.Reportable`

A reddit `Message`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

**read** ()

Marks message as read (POST). Calls `narwal.Reddit.read_message()`.

**unread** ()

Marks message as unread (POST). Calls `narwal.Reddit.unread_message()`.

**reply** (*text*)

POSTs reply to message with own message. Returns posted message.

URL: `http://www.reddit.com/api/comment/`

**Parameters** **text** – body text of message

**refresh** ()

Re-GETs this message (does not alter the object). Returns `Message` object.

**class** `narwal.things.Account` (*\*args, \*\*kwargs*)

Bases: `narwal.things.Thing`

A reddit `Account`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

**overview** (*limit=None*)

GETs overview of user's activities. Calls `narwal.Reddit.user_overview()`.

**Parameters** **limit** – max number of items to get

**comments** (*limit=None*)

GETs user's comments. Calls `narwal.Reddit.user_comments()`.

**Parameters** **limit** – max number of comments to get

**submitted** (*limit=None*)

GETs user's submissions. Calls `narwal.Reddit.user_submitted()`.

**Parameters** **limit** – max number of submissions to get

**about** ()

GETs this user (again). Calls `narwal.Reddit.user()`.

**message** (*subject, text*)

Compose a message to this user. Calls `narwal.Reddit.compose()`.

#### Parameters

- **subject** – subject of message
- **text** – body of message

**class** `narwal.things.More(*args, **kwargs)`

Bases: `narwal.things.Thing`

A reddit `More`. See <https://github.com/reddit/reddit/wiki/thing> for more details.

### 3.1.3 narwal.exceptions

**exception** `narwal.exceptions.AlienException`

Bases: `exceptions.Exception`

Generic exception

**exception** `narwal.exceptions.LoginFail`

Bases: `narwal.exceptions.AlienException`

Unable to login for whatever reason (bad user/password combo, no response, etc.)

**exception** `narwal.exceptions.NotLoggedIn`

Bases: `narwal.exceptions.AlienException`

Need to be logged in to call

**exception** `narwal.exceptions.NoMoreError`

Bases: `narwal.exceptions.AlienException`

Can't get next/prev items of a Listing

**exception** `narwal.exceptions.UnsupportedError`

Bases: `narwal.exceptions.AlienException`

Currently unsupported API feature

**exception** `narwal.exceptions.PostError(errors)`

Bases: `narwal.exceptions.AlienException`

Response containing reddit errors in response.

**errors = None**

list of string reddit errors received in response

**exception** `narwal.exceptions.BadResponse(response)`

Bases: `narwal.exceptions.AlienException`

A non-200 response.

**response = None**

the `requests.Response` object returned

**exception** `narwal.exceptions.UnexpectedResponse(jsonobj)`

Bases: `narwal.exceptions.AlienException`

A 200 response with unexpected content.

**jsonobj = None**

the json dict returned

## 3.2 Changelog

### 3.2.1 v0.3.2b (2012-05-21)

- replaced mutable default argument list() in ListBlob with the proper idiom

### 3.2.2 v0.3.1b (2012-05-10)

- fixed ImportError during installation

### 3.2.3 v0.3.0b (2012-05-07)

- added api\_type='json' to all POST data
- Reddit.post() now returns a (json) dict instead of requests.Response object.
- all POST methods that originally returned a requests.Response object now return something different. See the docs for more details.
- Listing.next\_listing() now works with comment listings
- fixed Reddit.submit\_link() and Reddit.submit\_text()
- fixed Reddit.distinguish()
- fixed Reddit.flairlist() and Reddit.flair()
- added Reddit.by\_id()
- added Subreddit.flairlist(), Subreddit.flair(), and Subreddit.flaircsv()
- changed PostError.errors to return error list exactly as returned by reddit
- renamed relative\_url to reddit\_url and made it work nicely with absolute URLs
- added Link.refresh() and Message.refresh()
- removed Reddit.hide\_message() and Reddit.unhide\_message()
- added limit kwarg to Reddit.moderators() and Reddit.contributors()
- added types 6=link, 7=message to work with local reddit instance
- added extensive test coverage for the reddit module
- fixed several doc errors

### 3.2.4 v0.2.6a (2012-04-28)

- added Reddit.user\_overview()
- attempting to “load more comments” via Listing.more() or .next\_listing() now raises UnsupportedError()

### **3.2.5 v0.2.5a (2012-04-24)**

- added limit parameter to `Reddit.info()`
- changed `Reddit._subreddit_get()` parameter order and removed default values
- fixed default value for items in `ListBlob` to be `list()` instead of `[]`
- added list methods to `ListBlob`
- fixed `Thing.__unicode__()` to always return a string (u" instead of None)
- added utf-8 coding declaration to source files
- more tests

### **3.2.6 v0.2.4a (2012-04-23)**

- automatically un-html-escape unicode chars
- fixed `ValueError` due to zero length fields when string formatting in older versions of Python (thanks staticsafe)

### **3.2.7 v0.2.3a (2012-04-21)**

- hotfix: default `Reddit._username` to None

### **3.2.8 v0.2.2a (2012-04-20)**

- hotfix: `/api/delete` -> `/api/del`

### **3.2.9 v0.2.1a (2012-04-20)**

- fixed `Listing.has_more()`
- made util functions more robust
- moved saving of username until successful login
- moved decorators outside of `Reddit` class for easier testing
- added some tests

### **3.2.10 v0.2.0a (2012-04-18)**

- added `Reddit.edit()` and `Commentable.edit()`
- added `Reddit.distinguish()` and `Commentable.distinguish()`
- refactored `.comments()`, `.remove()`, and `.delete()` into `Commentable`
- refactored `.hide()` and `.unhide()` into `Hideable` (subclassed by `Link`, `Message`)
- refactored `.report()` into `Reportable` (subclassed by `Link`, `Comment`, `Message`)
- changed `Comment.permalink` to return relative path
- added useful `__repr__` for `Reddit`
- removed debugging print statement from `Reddit.comment`

- changed urljoin and relative\_url to return unicode strings



# PYTHON MODULE INDEX

## n

`narwal`, [8](#)  
`narwal.exceptions`, [22](#)  
`narwal.things`, [17](#)