doapi Documentation

Release 0.2.0

John T. Wodder II

2019 Apr 15

Contents

Contents:

CHAPTER 1

Library

1.1 Introduction

Todo: Organize this better

Todo: Document potential weirdness when accessing `doapi.last_*` while a generator is being evaluated

Under normal circumstances, the `fetch` and `fetch_all_*` methods of a resource will only raise a *DOAPIError* if the resource no longer exists.

Generators producing objects always yield them in whatever order the API endpoint returns them in.

If you want all paginated results to be fetched at once, wrap the generator in `list()`.

Passing objects produced by one *doapi* object to methods of another results in undefined behavior.

1.1.1 Resource Objects

Todo: Document `doapi_manager`, dict/mapping methods, and conversion to a dict

Document that you can copy a Resource object by passing it to the constructor for its class?

Instances of classes representing DigitalOcean API resources — i.e., *Account*, *Action*, *BackupWindow*, *Domain*, *DomainRecord*, *Droplet*, *FloatingIP*, *Image*, *Kernel*, *NetworkInterface*, *Networks*, *Region*, *SSHKey*, *Size*, and *Tag* — make their API fields available in three different ways:

- as regular object attributes: `droplet.id`
- via indexing: `droplet["id"]`

- via indexing the `fields` dictionary attribute: `droplet.fields["id"]`

Modifying a resource object's fields only affects your local copy of the resource's data; to actually modify the resource on DigitalOcean's servers, call one of the object's methods.

Note that calling a mutating method on a resource object simply sends a request to the API endpoint and does not modify the local Python object. To get the most up-to-date information on a resource, you must call the resource object's `fetch` method to acquire a new object.

Note that resource objects have whatever attributes the API returns them with, which may or may not be the same set of attributes as the documentation says they should have. Also note that only documented attributes are ever converted to custom classes; e.g., if the API suddenly returns an SSH key with a `"region"` field, the region data will be left as a `dict` rather than converted to a *`Region`*.

All resource objects have the following method:

`Resource.`**`for_json`**`()`

New in version 0.2.0.

Recursively convert the resource and its attributes to values suitable for direct JSONification. This method is primarily intended for use by `simplejson.dump()`.

Return type dict

1.2 doapi Class

class `doapi.`**`doapi`**`(`*api_token*, *endpoint='https://api.digitalocean.com'*, *timeout=None*, *wait_interval=2*, *wait_time=None*, *per_page=None*`)`

The primary class for interacting with the DigitalOcean API, used for creating and fetching resources. The resource objects returned by these methods have methods of their own for manipulating them individually.

Parameters

- **`api_token`** (*`str`*) – the API token to use for authentication
- **`endpoint`** (*`string`*) – the URL relative to which requests will be made
- **`timeout`** (float, tuple, or `None`) – the `timeout` value to use when making requests
- **`wait_interval`** (*`number`*) – the default number of seconds that "wait" operations will sleep for between requests
- **`wait_time`** – the default number of seconds after which "wait" operations will return, or `None` or a negative number to wait indefinitely
- **`per_page`** (integer or `None`) – the default number of objects that *`paginate()`* will fetch on each request, or `None` to leave unspecified

`DEFAULT_ENDPOINT = 'https://api.digitalocean.com'`

The official DigitalOcean API endpoint

`api_token = None`

The API token used for authentication

`endpoint = None`

The API endpoint URL relative to which requests will be made

`timeout = None`

The `timeout` value to use when making requests; see the requests documentation for more information

wait_interval = None

The default number of seconds that *wait_droplets()*, *wait_actions()*, and the wait methods of *Action*, *Droplet*, *FloatingIP*, and *Image* will sleep for between requests

wait_time = None

The default number of seconds after which "wait" operations will return, or None or a negative number to wait indefinitely

per_page = None

The default number of objects that *paginate()* will fetch on each request, or None to leave unspecified

last_response = None

The requests.Response object returned for the most recent request, or None if no requests have been made yet

last_meta = None

The meta field in the body of the most recent response, or None if there was no such field, no requests have been made yet, or the last response was an error

session = None

The requests.Session object through which all requests are performed

close()

Close the session. All API methods will be unusable after calling this method.

Returns None

request(*url*, *params=None*, *data=None*, *method='GET'*)

Perform an HTTP request and return the response body as a decoded JSON value

Parameters

- **url** (*str*) – the URL to make the request of. If url begins with a forward slash, *endpoint* is prepended to it; otherwise, url is treated as an absolute URL.
- **params** (*dict*) – parameters to add to the URL's query string
- **data** – a value to send in the body of the request. If data is not a string, it will be serialized as JSON before sending; either way, the *Content-Type* header of the request will be set to *application/json*. Note that a data value of None means "Don't send any data"; to send an actual None value, convert it to JSON (i.e., the string "null") first.
- **method** (*str*) – the HTTP method to use: "GET", "POST", "PUT", or "DELETE" (case-insensitive); default: "GET"

Returns a decoded JSON value, or None if no data was returned

Return type list or dict (depending on the request) or None

Raises

- **ValueError** – if method is an invalid value
- ***DOAPIError*** – if the API endpoint replies with an error

last_rate_limit

A dict of the rate limit information returned in the most recent response, or None if no requests have been made yet. The dict consists of all headers whose names begin with "RateLimit" (case insensitive).

The DigitalOcean API specifies the following rate limit headers:

Variables

- **RateLimit-Limit** (*string*) – the number of requests that can be made per hour

- **RateLimit-Remaining** (*string*) – the number of requests remaining until the limit is reached
- **RateLimit-Reset** (*string*) – the Unix timestamp for the time when the oldest request will expire from rate limit consideration

paginate(*url*, *key*, *params=None*)

Fetch a sequence of paginated resources from the API endpoint. The initial request to `url` and all subsequent requests must respond with a JSON object; the field specified by `key` must be a list, whose elements will be yielded, and the next request will be made to the URL in the `.links.pages.next` field until the responses no longer contain that field.

Parameters

- **url** (*str*) – the URL to make the initial request of. If `url` begins with a forward slash, *endpoint* is prepended to it; otherwise, `url` is treated as an absolute URL.
- **key** (*str*) – the field on each page containing a list of values to yield
- **params** (*dict*) – parameters to add to the initial URL's query string. A `"per_page"` parameter may be included to override the default *per_page* setting.

Return type generator of decoded JSON values

Raises

- **ValueError** – if a response body is not an object or `key` is not one of its keys
- ***DOAPIError*** – if the API endpoint replies with an error

fetch_droplet(*obj*)

Fetch a droplet by ID number

Parameters **obj** (integer, `dict`, or *Droplet*) – the ID of the droplet, a `dict` with an `"id"` field, or a *Droplet* object (to re-fetch the same droplet)

Return type *Droplet*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_droplets(*tag_name=None*)

Returns a generator that yields all of the droplets belonging to the account

Changed in version 0.2.0: `tag_name` parameter added

Parameters **tag_name** (string or *Tag*) – if non-`None`, only droplets with the given tag are returned

Return type generator of *Droplet*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_droplet(*name*, *image*, *size*, *region*, *ssh_keys=None*, *backups=None*, *ipv6=None*, *private_networking=None*, *user_data=None*, ***kwargs*)

Create a new droplet. All fields other than `name`, `image`, `size`, and `region` are optional and will be omitted from the API request if not specified.

The returned *Droplet* object will represent the droplet at the moment of creation; the actual droplet may not be active yet and may not have even been assigned an IP address. To wait for the droplet to activate, use the *Droplet*'s *wait()* method.

Parameters

- **name** (*str*) – a name for the droplet

- **image** (integer, string, or *Image*) – the image ID, slug, or *Image* object representing the base image to use for the droplet
- **size** (string or *Size*) – the slug or *Size* object representing the size of the new droplet
- **region** (string or *Region*) – the slug or *Region* object representing the region in which to create the droplet
- **ssh_keys** (*iterable*) – an iterable of SSH key resource IDs, SSH key fingerprints, and/or *SSHKey* objects specifying the public keys to add to the new droplet's `/root/.ssh/authorized_keys` file
- **backups** (*bool*) – whether to enable automatic backups on the new droplet
- **ipv6** (*bool*) – whether to enable IPv6 on the new droplet
- **private_networking** (*bool*) – whether to enable private networking for the new droplet
- **user_data** (*str*) – a string of user data/metadata for the droplet
- **kwargs** – additional fields to include in the API request

Returns the new droplet resource

Return type *Droplet*

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_multiple_droplets(*names, image, size, region, ssh_keys=None, backups=None, ipv6=None, private_networking=None, user_data=None, **kwargs*)

Create multiple new droplets at once with the same image, size, etc., differing only in name. All fields other than `names`, `image`, `size`, and `region` are optional and will be omitted from the API request if not specified.

The returned *Droplet* objects will represent the droplets at the moment of creation; the actual droplets may not be active yet and may not have even been assigned IP addresses. To wait for the droplets to activate, use their *wait()* method or *wait_droplets*.

Parameters

- **names** (*list of strings*) – the names for the new droplets
- **image** (integer, string, or *Image*) – the image ID, slug, or *Image* object representing the base image to use for the droplets
- **size** (string or *Size*) – the slug or *Size* object representing the size of the new droplets
- **region** (string or *Region*) – the slug or *Region* object representing the region in which to create the droplets
- **ssh_keys** (*iterable*) – an iterable of SSH key resource IDs, SSH key fingerprints, and/or *SSHKey* objects specifying the public keys to add to the new droplets' `/root/.ssh/authorized_keys` files
- **backups** (*bool*) – whether to enable automatic backups on the new droplets
- **ipv6** (*bool*) – whether to enable IPv6 on the new droplets
- **private_networking** (*bool*) – whether to enable private networking for the new droplets
- **user_data** (*str*) – a string of user data/metadata for the droplets
- **kwargs** – additional fields to include in the API request

Returns the new droplet resources

Return type list of *Droplet*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_droplet_neighbors()

Returns a generator of all sets of multiple droplets that are running on the same physical hardware

Return type generator of lists of *Droplet*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

wait_droplets(*droplets*, *status=None*, *locked=None*, *wait_interval=None*, *wait_time=None*)

Poll the server periodically until all droplets in `droplets` have reached some final state, yielding each *Droplet*'s final value when it's done. If `status` is non-`None`, `wait_droplets` will wait for each droplet's `status` field to equal the given value. If `locked` is non-`None`, `wait_droplets` will wait for each droplet's `locked` field to equal (the truth value of) the given value. Exactly one of `status` and `locked` must be non-`None`.

If `wait_time` is exceeded, a *WaitTimeoutError* (containing any remaining in-progress droplets) is raised.

If a `KeyboardInterrupt` is caught, any remaining droplets are returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Changed in version 0.2.0: `locked` parameter added

Changed in version 0.2.0: No longer waits for actions to complete

Parameters

- **droplets** (*iterable*) – an iterable of *Droplet*s and/or other values that are acceptable arguments to *fetch_droplet()*
- **status** (string or `None`) – When non-`None`, the desired value for the `status` field of each *Droplet*, which should be one of *Droplet.STATUS_ACTIVE*, *Droplet.STATUS_ARCHIVE*, *Droplet.STATUS_NEW*, and *Droplet.STATUS_OFF*. (For the sake of forwards-compatibility, any other value is accepted as well.)
- **locked** (`bool` or `None`) – When non-`None`, the desired value for the `locked` field of each *Droplet*
- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to *wait_interval* if not specified or `None`
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if any droplets have not yet completed, or a negative number to wait indefinitely; defaults to *wait_time* if not specified or `None`

Return type generator of *Droplet*s

Raises

- **TypeError** – if both or neither of `status` & `locked` are defined
- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if `wait_time` is exceeded

fetch_action(*obj*)

Fetch an action by ID number

Parameters **obj** (integer, dict, or *Action*) – the ID of the action, a dict with an "id" field, or an *Action* object (to re-fetch the same action)

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_last_action()

Fetch the most recent action performed on the account, or None if no actions have been performed yet. If multiple actions were triggered simultaneously, the choice of which to return is undefined.

Return type *Action* or None

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_actions()

Returns a generator that yields all of the actions associated with the account

Return type generator of *Action*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

wait_actions(*actions*, *wait_interval=None*, *wait_time=None*)

Poll the server periodically until all actions in actions have either completed or errored out, yielding each *Action*'s final value as it ends.

If wait_time is exceeded, a *WaitTimeoutError* (containing any remaining in-progress actions) is raised.

If a KeyboardInterrupt is caught, any remaining actions are returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Parameters

- **actions** (*iterable*) – an iterable of *Action*s and/or other values that are acceptable arguments to *fetch_action()*
- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to *wait_interval* if not specified or None
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if any actions have not yet completed, or a negative number to wait indefinitely; defaults to *wait_time* if not specified or None

Return type generator of *Action*s

Raises

- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if wait_time is exceeded

wait_actions_on_objects(*objects*, *wait_interval=None*, *wait_time=None*)

New in version 0.2.0.

Poll the server periodically until the most recent action on each resource in objects has finished, yielding each resource's final state when the corresponding action is done.

If wait_time is exceeded, a *WaitTimeoutError* (containing any remaining in-progress actions) is raised.

If a KeyboardInterrupt is caught, any remaining actions are returned immediately without waiting for completion.

Parameters

- **objects** (*iterable*) – an iterable of resource objects that have fetch_last_action methods
- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to *wait_interval* if not specified or None
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if any actions have not yet completed, or a negative number to wait indefinitely; defaults to *wait_time* if not specified or None

Return type generator of objects

Raises

- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if wait_time is exceeded

fetch_ssh_key(*obj*)

Fetch an SSH public key by ID number or fingerprint

Parameters **obj** (integer, string, dict, or *SSHKey*) – the ID or fingerprint of the SSH key, a dict with an "id" or "fingerprint" field, or an *SSHKey* object (to re-fetch the same SSH key)

Return type *SSHKey*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_ssh_keys()

Returns a generator that yields all of the SSH public keys belonging to the account

Return type generator of *SSHKey*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_ssh_key(*name*, *public_key*, ***kwargs*)

Add a new SSH public key resource to the account

Parameters

- **name** (*str*) – the name to give the new SSH key resource
- **public_key** (*str*) – the text of the public key to register, in the form used by authorized_keys files
- **kwargs** – additional fields to include in the API request

Returns the new SSH key resource

Return type *SSHKey*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_image(*obj*)

Fetch an image by ID number

Parameters **obj** (integer, dict, or *Image*) – the ID of the image, a dict with an "id" field, or an *Image* object (to re-fetch the same image)

Return type *Image*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_image_by_slug(*slug*)
Fetch an image by its slug

Parameters **slug** (*str*) – the slug of the image to fetch

Return type *Image*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_images(*type=None*, *private=None*)
Returns a generator that yields all of the images available to the account

Parameters

- **type** (*string or None*) – the type of images to fetch: `"distribution"`, `"application"`, or all (`None`); default: `None`
- **private** (*bool*) – whether to only return the user's private images; default: return all images

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_distribution_images()
Returns a generator that yields all of the distribution images available to the account

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_application_images()
Returns a generator that yields all of the application images available to the account

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_private_images()
Returns a generator that yields all of the user's private images

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_regions()
Returns a generator that yields all of the regions available to the account

Return type generator of *Region*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_sizes()
Returns a generator that yields all of the sizes available to the account

Return type generator of *Size*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_account()
Returns an *Account* object representing the user's account

Return type *Account*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_domain(*obj*)
Fetch a domain by FQDN

Parameters **obj** (string, dict, or *Domain*) – the domain name, a dict with a "name" field, or a *Domain* object (to re-fetch the same domain)

Return type *Domain*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_domains()

Returns a generator that yields all of the domains belonging to the account

Return type generator of *Domain*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_domain(*name*, *ip_address*, ***kwargs*)

Add a new domain name resource to the account.

Note that this method does not actually register a new domain name; it merely configures DigitalOcean's nameservers to provide DNS resolution for the domain. See How To Set Up a Host Name with DigitalOcean for more information.

Parameters

- **name** (*str*) – the domain name to add
- **ip_address** (string or *FloatingIP*) – the IP address to which the domain should point
- **kwargs** – additional fields to include in the API request

Returns the new domain resource

Return type *Domain*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_floating_ip(*obj*)

Fetch a floating IP

Parameters **obj** (string, integer, dict, or *FloatingIP*) – an IP address (as a string or 32-bit integer), a dict with an "ip" field, or a *FloatingIP* object (to re-fetch the same floating IP)

Return type *FloatingIP*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_floating_ips()

Returns a generator that yields all of the floating IPs belonging to the account

Return type generator of *FloatingIP*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_floating_ip(*droplet_id=None*, *region=None*, ***kwargs*)

Create a new floating IP assigned to a droplet or reserved to a region. Either droplet_id or region must be specified, but not both.

The returned *FloatingIP* object will represent the IP at the moment of creation; if the IP address is supposed to be assigned to a droplet, the assignment may not have been completed at the time the object is returned. To wait for the assignment to complete, use the *FloatingIP*'s *wait_for_action()* method.

Parameters

- **droplet_id** (integer or *Droplet*) – the droplet to assign the floating IP to as either an ID or a *Droplet* object
- **region** (string or *Region*) – the region to reserve the floating IP to as either a slug or a *Region* object
- **kwargs** – additional fields to include in the API request

Returns the new floating IP

Return type *FloatingIP*

Raises

- **TypeError** – if both droplet_id & region or neither of them are defined
- ***DOAPIError*** – if the API endpoint replies with an error

fetch_tag(*obj*)

New in version 0.2.0.

Fetch a tag by name

Parameters **obj** (string, dict, or *Tag*) – the name of the tag, a dict with a "name" field, or a *Tag* object (to re-fetch the same tag)

Return type *Tag*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_tags()

New in version 0.2.0.

Returns a generator that yields all of the tags belonging to the account

Return type generator of *Tag*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_tag(*name*)

New in version 0.2.0.

Add a new tag resource to the account

Parameters **name** (*str*) – the name of the new tag

Return type *Tag*

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.3 Droplets

1.3.1 Droplet

class doapi.**Droplet**

A droplet resource, representing a virtual machine provided by DigitalOcean.

New droplets are created via the *doapi.create_droplet()* and *doapi.create_multiple_droplets()* methods and can be retrieved with the *doapi.fetch_droplet()* and *doapi.fetch_all_droplets()* methods.

The DigitalOcean API specifies the following fields for droplet objects:

Variables

- **id** (*int*) – a unique identifier for the droplet
- **backup_ids** (*list of integers*) – image IDs of backups taken of the droplet
- **created_at** (*datetime.datetime*) – date & time of the droplet's creation
- **disk** (*number*) – size of the droplet's disk in gigabytes
- **features** (*list of strings*) – a list of strings naming the features enabled on the droplet
- **image** (*Image*) – the base image used to create the droplet
- **kernel** (*Kernel* or `None`) – the droplet's current kernel
- **locked** (*bool*) – whether the droplet is currently locked, preventing actions on it
- **memory** (*number*) – RAM of the droplet in megabytes
- **name** (*string*) – a human-readable name for the droplet
- **networks** (*Networks*) – the network interfaces configured for the droplet
- **next_backup_window** (*BackupWindow* or `None`) – the start & end of the next timeframe in which the droplet will be backed up; only defined if backups are enabled on the droplet
- **region** (*Region*) – the region in which the droplet is located
- **size** (*Size*) – the current size of the droplet
- **size_slug** (*string*) – the unique slug identifier for the droplet's size
- **snapshot_ids** (*list of integers*) – image IDs of snapshots taken of the droplet
- **status** (*string*) – the current state of the droplet: `"new"`, `"active"`, `"off"`, or `"archive"`
- **tags** (*list of strings*) – tags that have been applied to the droplet
- **vcpus** (*int*) – number of virtual CPUs

action_url

The endpoint for actions on the individual resource

act(***data*)

Perform an arbitrary action on the resource. `data` will be serialized as JSON and POSTed to the resource's *action_url*. All currently-documented actions require the POST body to be a JSON object containing, at a minimum, a `"type"` field.

Returns an *Action* representing the in-progress operation on the resource

Return type *Action*

Raises **DOAPIError** – if the API endpoint replies with an error

wait_for_action(*wait_interval=None*, *wait_time=None*)

Poll the server periodically until the resource's most recent action has either completed or errored out, and return the resource's final state afterwards. If no actions have ever been performed on the resource, return `self`. If the resource no longer exists by the time the action has completed, return `None`.

If `wait_time` is exceeded, a *WaitTimeoutError* (containing the resource's current state) is raised.

If a `KeyboardInterrupt` is caught, the resource's current state is returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Changed in version 0.2.0: Name changed from `wait` to `wait_for_action`

Changed in version 0.2.0: Return `self` if there were no actions on the resource

Changed in version 0.2.0: Return `None` if the resource no longer exists afterwards

Parameters

- **`wait_interval`** (*number*) – how many seconds to sleep between requests; defaults to the *`doapi`* object's *`wait_interval`* if not specified or `None`
- **`wait_time`** (*number*) – the total number of seconds after which the method will raise an error if the action has not yet completed, or a negative number to wait indefinitely; defaults to the *`doapi`* object's *`wait_time`* if not specified or `None`

Returns the resource's final state

Raises

- ***`DOAPIError`*** – if the API endpoint replies with an error
- ***`WaitTimeoutError`*** – if `wait_time` is exceeded

`fetch_all_actions`()

Returns a generator that yields all of the actions associated with the resource

Return type generator of *`Action`*s

Raises ***`DOAPIError`*** – if the API endpoint replies with an error

`fetch_last_action`()

Fetch the most recent action performed on the resource, or `None` if no actions have been performed on it yet. If multiple actions were triggered simultaneously, the choice of which to return is undefined.

Changed in version 0.2.0: Return `None` if there were no actions on the resource

Return type *Action*

Raises ***`DOAPIError`*** – if the API endpoint replies with an error

`fetch_current_action`()

Fetch the action currently in progress on the resource, or `None` if there is no such action

Return type *`Action`* or `None`

Raises ***`DOAPIError`*** – if the API endpoint replies with an error

`__int__`()

Convert the resource to its unique integer ID

`tag`(*t*)

New in version 0.2.0.

Apply the given tag to the resource

Parameters **`t`** (string or *`Tag`*) – the tag to apply

Returns `None`

Raises ***`DOAPIError`*** – if the API endpoint replies with an error

`untag`(*t*)

New in version 0.2.0.

Remove the given tag from the resource

Parameters **`t`** (string or *`Tag`*) – the tag to remove

Returns `None`

Raises ***DOAPIError*** – if the API endpoint replies with an error

STATUS_ACTIVE = 'active'
The status of droplets that are powered on and operating

STATUS_ARCHIVE = 'archive'
The status of "archived" droplets

STATUS_NEW = 'new'
The status of recently-created droplets that are not yet usable

STATUS_OFF = 'off'
The status of droplets that are powered off

active
`True` iff the droplet's status is `"active"`

new
`True` iff the droplet's status is `"new"`

off
`True` iff the droplet's status is `"off"`

archive
`True` iff the droplet's status is `"archive"`

region_slug
The unique slug identifier for the droplet's region

image_slug
The unique slug identifier for the droplet's image, or `None` if the image doesn't have a slug

ip_address
The IP address of the first interface listed in the droplet's `networks` field (ordering IPv4 before IPv6), or `None` if there are no interfaces

ipv4_address
New in version 0.2.0.

The IP address of the first IPv4 interface listed in the droplet's `networks` field, or `None` if there is no IPv4 interface

ipv6_address
New in version 0.2.0.

The IP address of the first IPv6 interface listed in the droplet's `networks` field, or `None` if there is no IPv4 interface

url
The endpoint for general operations on the individual droplet

fetch()
Fetch & return a new *Droplet* object representing the droplet's current state

Return type *Droplet*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the droplet no longer exists)

fetch_all_neighbors()
Returns a generator that yields all of the droplets running on the same physical server as the droplet

Return type generator of *Droplet*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_snapshots()

Returns a generator that yields all of the snapshot images created from the droplet

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_backups()

Returns a generator that yields all of the backup images created from the droplet

Return type generator of *Image*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_kernels()

Returns a generator that yields all of the kernels available to the droplet

Return type generator of *Kernel*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_backups()

Enable backups on the droplet

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

disable_backups()

Disable backups on the droplet

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

reboot()

Reboot the droplet

A reboot action is an attempt to reboot the Droplet in a graceful way, similar to using the **reboot** command from the console. [APIDocs]

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_cycle()

Power cycle the droplet

A powercycle action is similar to pushing the reset button on a physical machine, it's similar to booting from scratch. [APIDocs]

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

shutdown()

Shut down the droplet

A shutdown action is an attempt to shutdown the Droplet in a graceful way, similar to using the **shutdown** command from the console. Since a `shutdown` command can fail, this action guarantees that the command is issued, not that it succeeds. The preferred way to turn off a Droplet is to attempt a shutdown, with a reasonable timeout, followed by a power off action to ensure the Droplet is off. [APIDocs]

Returns an `Action` representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_off()

Power off the droplet

A `power_off` event is a hard shutdown and should only be used if the `shutdown()` action is not successful. It is similar to cutting the power on a server and could lead to complications. [APIDocs]

Returns an `Action` representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_on()

Power on the droplet

Returns an `Action` representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

restore(*image*)

Restore the droplet to the specified backup image

A Droplet restoration will rebuild an image using a backup image. The image ID that is passed in must be a backup of the current Droplet instance. The operation will leave any embedded SSH keys intact. [APIDocs]

Parameters **image** (integer, string, or `Image`) – an image ID, an image slug, or an `Image` object representing a backup image of the droplet

Returns an `Action` representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

password_reset()

Reset the password for the droplet

Returns an `Action` representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

resize(*size*, *disk=None*)
Resize the droplet

Parameters

- **size** (string or *Size*) – a size slug or a *Size* object representing the size to resize to
- **disk** (*bool*) – Set to `True` for a permanent resize, including disk changes

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

rebuild(*image*)
Rebuild the droplet with the specified image

A rebuild action functions just like a new create. [APIDocs]

Parameters image (integer, string, or *Image*) – an image ID, an image slug, or an *Image* object representing the image the droplet should use as a base

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

rename(*name*)
Rename the droplet

Parameters name (*str*) – the new name for the droplet

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

change_kernel(*kernel*)
Change the droplet's kernel

Parameters kernel (integer or *Kernel*) – a kernel ID or *Kernel* object representing the new kernel

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_ipv6()
Enable IPv6 networking on the droplet

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_private_networking()
Enable private networking on the droplet

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises *DOAPIError* – if the API endpoint replies with an error

snapshot (*name*)

Create a snapshot image of the droplet

Parameters **name** (*str*) – the name for the new snapshot

Returns an *Action* representing the in-progress operation on the droplet

Return type *Action*

Raises *DOAPIError* – if the API endpoint replies with an error

delete ()

Delete the droplet

Returns None

Raises *DOAPIError* – if the API endpoint replies with an error

wait (*status=None*, *locked=None*, *wait_interval=None*, *wait_time=None*)

Poll the server periodically until the droplet has reached some final state. If status is non-None, wait will wait for the droplet's status field to equal the given value. If locked is non-None, *wait* will wait for the droplet's locked field to equal (the truth value of) the given value. Exactly one of status and locked must be non-None.

If wait_time is exceeded, a *WaitTimeoutError* (containing the droplet's most recently fetched state) is raised.

If a KeyboardInterrupt is caught, the droplet's most recently fetched state is returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Changed in version 0.2.0: locked parameter added

Changed in version 0.2.0: No longer waits for latest action to complete

Parameters

- **status** (string or None) – When non-None, the desired value for the status field of the droplet, which should be one of *Droplet.STATUS_ACTIVE*, *Droplet.STATUS_ARCHIVE*, *Droplet.STATUS_NEW*, and *Droplet.STATUS_OFF*. (For the sake of forwards-compatibility, any other value is accepted as well.)
- **locked** (bool or None) – When non-None, the desired value for the locked field of the droplet
- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to the *doapi* object's *wait_interval* if not specified or None
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if the droplet has not yet completed, or a negative number to wait indefinitely; defaults to the *doapi* object's *wait_time* if not specified or None

Returns the droplet's final state

Return type *Droplet*

Raises

- **TypeError** – if both or neither of status & locked are defined
- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if wait_time is exceeded

1.3.2 BackupWindow

class doapi.**BackupWindow**

A backup window resource, representing an upcoming timeframe in which a droplet is scheduled to be backed up.

A *Droplet*'s next backup window is stored in its `next_backup_window` attribute.

The DigitalOcean API implicitly specifies the following fields for backup window objects:

Variables

- **start** (*datetime.datetime*) – beginning of the window
- **end** (*datetime.datetime*) – end of the window

1.3.3 Kernel

class doapi.**Kernel**

A kernel resource, representing a kernel version that can be installed on a given droplet.

A *Droplet*'s current kernel is stored in its `kernel` attribute, and the set of kernels available to a given *Droplet* can be retrieved with the `droplet.fetch_all_kernels()` method.

The DigitalOcean API specifies the following fields for kernel objects:

Variables

- **id** (*int*) – a unique identifier for the kernel
- **name** (*string*) – a human-readable name for the kernel
- **version** (*string*) – the version string for the kernel

__int__()

Convert the resource to its unique integer ID

1.3.4 Networks

class doapi.**Networks**

A networks resource, representing a set of network interfaces configured for a specific droplet.

A *Droplet*'s network information is stored in its `networks` attribute.

The DigitalOcean API implicitly specifies the following fields for networks objects:

Variables

- **v4** (list of *NetworkInterface*s) – a list of IPv4 interfaces allocated for a droplet
- **v6** (list of *NetworkInterface*s) – a list of IPv6 interfaces allocated for a droplet

1.3.5 NetworkInterface

class doapi.**NetworkInterface**

A network interface resource, representing an IP address allocated to a specific droplet.

A *Droplet*'s network interfaces are listed in its `networks` attribute.

The DigitalOcean API implicitly specifies the following fields for network interface objects:

Variables

- **gateway** (*string*) – gateway
- ***ip_address*** (*string*) – IP address
- **netmask** – netmask
- **type** – "public" or "private"

ip_version

The IP version used by the interface: 4 or 6

__str__()

Show just the IP address of the interface

1.4 Domains

1.4.1 Domain

class doapi.**Domain**

A domain resource, representing a domain name whose DNS is managed by DigitalOcean's nameservers.

New domains are created via the *doapi.create_domain()* method and can be retrieved with the *doapi.fetch_domain()* and *doapi.fetch_all_domains()* methods.

The DigitalOcean API specifies the following fields for domain objects:

Variables

- **name** (*string*) – the domain name
- **ttl** (*number*) – the time-to-live for the domain's records, in seconds
- **zone_file** (*string*) – the complete zone file for the domain

__str__()

Convert the domain to just the actual domain name

url

The endpoint for general operations on the individual domain

fetch()

Fetch & return a new *Domain* object representing the domain's current state

Return type *Domain*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the domain no longer exists)

delete()

Delete the domain

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

record_url

The endpoint for operations on the domain's DNS records

fetch_record(*obj*)

Fetch a domain record by ID number

Parameters **obj** (integer, dict, or *DomainRecord*) – the ID of the record, a dict with an "id" field, or a *DomainRecord* object (to re-fetch the same record)

Return type *DomainRecord*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_records()

Returns a generator that yields all of the DNS records for the domain

Return type generator of *DomainRecord*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

create_record(*type*, *name*, *data*, *priority=None*, *port=None*, *weight=None*, ***kwargs*)

Add a new DNS record to the domain

Parameters

- **type** (*str*) – the type of DNS record to add ("A", "CNAME", etc.)
- **name** (*str*) – the name (hostname, alias, etc.) of the new record
- **data** (*str*) – the value of the new record
- **priority** (*int*) – the priority of the new record (SRV and MX records only)
- **port** (*int*) – the port that the service is accessible on (SRV records only)
- **weight** (*int*) – the weight of records with the same priority (SRV records only)
- **kwargs** – additional fields to include in the API request

Returns the new domain record

Return type *DomainRecord*

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.4.2 DomainRecord

class doapi.**DomainRecord**

A domain record resource, representing an individual DNS record that can be set & modified by the user of the DigitalOcean API.

New domain records are created via the *Domain.create_record()* method and can be retrieved with the *Domain.fetch_record()* and *Domain.fetch_all_records()* methods.

The DigitalOcean API specifies the following fields for domain record objects:

Variables

- **id** (*int*) – a unique identifier for the domain record
- **type** (*string*) – the type of the DNS record
- **name** (*string*) – the name of the DNS record
- **data** (*string*) – the value of the DNS record
- **priority** (number or None) – the priority of the record (SRV and MX records only)
- **port** (number or None) – the port of the record (SRV records only)
- **weight** (number or None) – the weight of the record (SRV records only)

domain
The `Domain` to which the record belongs

__int__()
Convert the resource to its unique integer ID

url
The endpoint for general operations on the individual domain record

fetch()
Fetch & return a new `DomainRecord` object representing the domain record's current state

Return type *DomainRecord*

Raises **DOAPIError** – if the API endpoint replies with an error (e.g., if the domain record no longer exists)

fetch_domain()
Fetch & return the domain resource that the record belongs to

Return type *Domain*

Raises **DOAPIError** – if the API endpoint replies with an error

update_record(***attrs*)
Update the record, modifying any number of its attributes (except `id`). `update_record` takes the same keyword arguments as `Domain.create_record()`; pass in only those attributes that you want to update.

Returns an updated `DomainRecord` object

Return type *DomainRecord*

Raises **DOAPIError** – if the API endpoint replies with an error

delete()
Delete the domain record

Returns `None`

Raises **DOAPIError** – if the API endpoint replies with an error

1.5 Floating IPs

1.5.1 FloatingIP

class doapi.**FloatingIP**
A floating IP resource, representing a public IP address that can be (re)assigned at any time to any droplet in a certain region.

New floating IPs are created via the `doapi.create_floating_ip()` method and can be retrieved with the `doapi.fetch_floating_ip()` and `doapi.fetch_all_floating_ips()` methods.

The DigitalOcean API specifies the following fields for floating IP objects:

Variables

- **ip** (*string*) – the IP address
- **droplet** (`Droplet` or `None`) – the droplet the floating IP is currently assigned to, or `None` if the address is currently unassigned

- **region** (*Region*) – the region the floating IP is reserved to

action_url
The endpoint for actions on the individual resource

act(***data*)
Perform an arbitrary action on the resource. `data` will be serialized as JSON and POSTed to the resource's *action_url*. All currently-documented actions require the POST body to be a JSON object containing, at a minimum, a `"type"` field.

Returns an *Action* representing the in-progress operation on the resource

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

wait_for_action(*wait_interval=None*, *wait_time=None*)
Poll the server periodically until the resource's most recent action has either completed or errored out, and return the resource's final state afterwards. If no actions have ever been performed on the resource, return `self`. If the resource no longer exists by the time the action has completed, return `None`.

If `wait_time` is exceeded, a *WaitTimeoutError* (containing the resource's current state) is raised.

If a `KeyboardInterrupt` is caught, the resource's current state is returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Changed in version 0.2.0: Name changed from `wait` to `wait_for_action`

Changed in version 0.2.0: Return `self` if there were no actions on the resource

Changed in version 0.2.0: Return `None` if the resource no longer exists afterwards

Parameters

- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to the *doapi* object's *wait_interval* if not specified or `None`
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if the action has not yet completed, or a negative number to wait indefinitely; defaults to the *doapi* object's *wait_time* if not specified or `None`

Returns the resource's final state

Raises

- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if `wait_time` is exceeded

fetch_all_actions()
Returns a generator that yields all of the actions associated with the resource

Return type generator of *Action*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_last_action()
Fetch the most recent action performed on the resource, or `None` if no actions have been performed on it yet. If multiple actions were triggered simultaneously, the choice of which to return is undefined.

Changed in version 0.2.0: Return `None` if there were no actions on the resource

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_current_action()

Fetch the action currently in progress on the resource, or `None` if there is no such action

Return type *Action* or `None`

Raises ***DOAPIError*** – if the API endpoint replies with an error

__str__()

Convert the floating IP to just the actual IP address

__int__()

New in version 0.2.0.

Convert the floating IP to a 32-bit integer

url

The endpoint for general operations on the individual floating IP

fetch()

Fetch & return a new *FloatingIP* object representing the floating IP's current state

Return type *FloatingIP*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the floating IP no longer exists)

delete()

Delete the floating IP

Returns `None`

Raises ***DOAPIError*** – if the API endpoint replies with an error

assign(*droplet_id*)

Assign the floating IP to a droplet

Parameters **droplet_id** (integer or *Droplet*) – the droplet to assign the floating IP to as either an ID or a *Droplet* object

Returns an *Action* representing the in-progress operation on the floating IP

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

unassign()

Unassign the floating IP

Returns an *Action* representing the in-progress operation on the floating IP

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.6 Images

1.6.1 Image

class doapi.**Image**

An image resource, representing an OS image that can be used to create or reset a droplet.

New images can be created via the `Droplet.snapshot()` method. They are also automatically created regularly for droplets that have backups enabled. Images can be retrieved with the `doapi.fetch_image()` and `doapi.fetch_all_images()` methods, among others.

The DigitalOcean API specifies the following fields for domain objects:

Variables

- **`id`** (*`int`*) – a unique identifier for the image
- **`created_at`** (*`datetime.datetime`*) – date & time of the image's creation
- **`distribution`** (*`string`*) – the base Linux distribution used for the image
- **`min_disk_size`** (*`number`*) – the minimum `disk` size (in gigabytes) required for a droplet to use the image
- **`name`** (*`string`*) – a human-readable name for the image
- **`public`** (*`bool`*) – whether the image is public (i.e., available to all accounts) or not (i.e., only accessible from your account)
- **`regions`** (*`list of strings`*) – the slugs of the regions in which the image is available
- **`size_gigabytes`** (*`number`*) – the size of the image in gigabytes
- **`slug`** (string or `None`) – the unique slug identifier for the image (only defined for public images)
- **`type`** (*`string`*) – the type of the image: `"snapshot"` or `"backup"`

`action_url`

The endpoint for actions on the individual resource

`act`(***data*)

Perform an arbitrary action on the resource. `data` will be serialized as JSON and POSTed to the resource's `action_url`. All currently-documented actions require the POST body to be a JSON object containing, at a minimum, a `"type"` field.

Returns an `Action` representing the in-progress operation on the resource

Return type *Action*

Raises **`DOAPIError`** – if the API endpoint replies with an error

`wait_for_action`(*wait_interval=None*, *wait_time=None*)

Poll the server periodically until the resource's most recent action has either completed or errored out, and return the resource's final state afterwards. If no actions have ever been performed on the resource, return `self`. If the resource no longer exists by the time the action has completed, return `None`.

If `wait_time` is exceeded, a `WaitTimeoutError` (containing the resource's current state) is raised.

If a `KeyboardInterrupt` is caught, the resource's current state is returned immediately without waiting for completion.

Changed in version 0.2.0: Raises `WaitTimeoutError` on timeout

Changed in version 0.2.0: Name changed from `wait` to `wait_for_action`

Changed in version 0.2.0: Return `self` if there were no actions on the resource

Changed in version 0.2.0: Return `None` if the resource no longer exists afterwards

Parameters

- **`wait_interval`** (*`number`*) – how many seconds to sleep between requests; defaults to the `doapi` object's `wait_interval` if not specified or `None`

- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if the action has not yet completed, or a negative number to wait indefinitely; defaults to the *doapi* object's *wait_time* if not specified or None

Returns the resource's final state

Raises

- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if wait_time is exceeded

fetch_all_actions()

Returns a generator that yields all of the actions associated with the resource

Return type generator of *Action*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_last_action()

Fetch the most recent action performed on the resource, or None if no actions have been performed on it yet. If multiple actions were triggered simultaneously, the choice of which to return is undefined.

Changed in version 0.2.0: Return None if there were no actions on the resource

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_current_action()

Fetch the action currently in progress on the resource, or None if there is no such action

Return type *Action* or None

Raises ***DOAPIError*** – if the API endpoint replies with an error

__int__()

Convert the resource to its unique integer ID

__str__()

Convert the image to its slug representation. If the image does not have a slug, an AttributeError is raised.

url

The endpoint for general operations on the individual image

fetch()

Fetch & return a new *Image* object representing the image's current state

Return type *Image*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the image no longer exists)

update_image(*name*)

Update (i.e., rename) the image

Parameters **name** (*str*) – the new name for the image

Returns an updated *Image* object

Return type *Image*

Raises ***DOAPIError*** – if the API endpoint replies with an error

delete()

Delete the image

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

transfer(*region*)

Transfer the image to another region

Parameters **region** (string or *Region*) – the slug or *Region* object representing the region to which to transfer the image

Returns an *Action* representing the in-progress operation on the image

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

convert()

Convert the image to a snapshot

Returns an *Action* representing the in-progress operation on the image

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.7 SSH Keys

1.7.1 SSHKey

class doapi.**SSHKey**

An SSH key resource, representing an SSH public key that can be automatically added to the /root/.ssh/authorized_keys files of new droplets.

New SSH keys are created via the *doapi.create_ssh_key()* method and can be retrieved with the *doapi.fetch_ssh_key()* and *doapi.fetch_all_ssh_keys()* methods.

The DigitalOcean API specifies the following fields for SSH key objects:

Variables

- **id** (*int*) – a unique identifier for the SSH key
- **fingerprint** (*string*) – the unique fingerprint of the SSH key
- **name** (*string*) – a human-readable name for the SSH key
- **public_key** (*string*) – the entire SSH public key as it was uploaded to DigitalOcean

__int__()

Convert the resource to its unique integer ID

__str__()

Convert the SSH key to its fingerprint

url

The endpoint for general operations on the individual SSH key

fetch()

Fetch & return a new *SSHKey* object representing the SSH key's current state

Return type *SSHKey*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the SSH key no longer exists)

update_ssh_key(*name*)
Update (i.e., rename) the SSH key

Parameters **name** (*str*) – the new name for the SSH key

Returns an updated *SSHKey* object

Return type *SSHKey*

Raises ***DOAPIError*** – if the API endpoint replies with an error

delete()
Delete the SSH key

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.8 Tags

1.8.1 Tag

class doapi.**Tag**
New in version 0.2.0.

A tag resource, representing a label that can be applied to other resources.

New tags are created via the *doapi.create_tag()* method and can be retrieved with the *doapi.fetch_tag()* and *doapi.fetch_all_tags()* methods.

The DigitalOcean API specifies the following fields for tag objects:

Variables

- **name** (*string*) – the name of the tag
- **resources** – a dict mapping resource types (e.g., "droplets") to sub-dicts containing fields "count" (the number of resources of the given type with the given tag) and "last_tagged" (the resource of the given type to which the tag was most recently applied)

url
The endpoint for general operations on the individual tag

fetch()
Fetch & return a new *Tag* object representing the tag's current state

Return type *Tag*

Raises ***DOAPIError*** – if the API endpoint replies with an error (e.g., if the tag no longer exists)

__str__()
Convert the tag object to its name

update_tag(*name*)
Update (i.e., rename) the tag

Parameters **name** (*str*) – the new name for the tag

Returns an updated *Tag* object

Return type *Tag*

Raises ***DOAPIError*** – if the API endpoint replies with an error

delete()

Delete the tag

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

add(**resources*)

Apply the tag to one or more resources

Parameters **resources** – one or more `Resource` objects to which tags can be applied

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

remove(**resources*)

Remove the tag from one or more resources

Parameters **resources** – one or more `Resource` objects to which tags can be applied

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_all_droplets()

Returns a generator that yields all of the droplets to which the tag is currently applied

Return type generator of *Droplet*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

delete_all_droplets()

Delete all of the droplets to which the tag is applied

Returns None

Raises ***DOAPIError*** – if the API endpoint replies with an error

act_on_droplets(***data*)

Perform an arbitrary action on all of the droplets to which the tag is applied. `data` will be serialized as JSON and POSTed to the proper API endpoint. All currently-documented actions require the POST body to be a JSON object containing, at a minimum, a `"type"` field.

Returns a generator of *Action*s representing the in-progress operations on the droplets

Return type generator of *Action*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_cycle()

Power cycle all of the droplets to which the tag is applied

Returns a generator of *Action*s representing the in-progress operations on the droplets

Return type generator of *Action*s

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_on()

Power on all of the droplets to which the tag is applied

Returns a generator of *Action*s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

power_off()

Power off all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

shutdown()

Shut down all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_private_networking()

Enable private networking on all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_ipv6()

Enable IPv6 networking on all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

enable_backups()

Enable backups on all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

disable_backups()

Disable backups on all of the droplets to which the tag is applied

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

snapshot(*name*)

Create snapshot images of all of the droplets to which the tag is applied

Parameters **name** (`str`) – the name for the new snapshots

Returns a generator of `Action`s representing the in-progress operations on the droplets

Return type generator of `Action`s

Raises ***DOAPIError*** – if the API endpoint replies with an error

1.9 Immutable Resources

1.9.1 Account

class doapi.**Account**

An account resource describing the user's DigitalOcean account.

Current details on the user's account can be retrieved with the *doapi.fetch_account()* method.

The DigitalOcean API specifies the following fields for account objects:

Variables

- **droplet_limit** (*int*) – the maximum number of droplets the account may have at any one time
- **email** (*string*) – the e-mail address the account used to register for DigitalOcean
- **email_verified** (*bool*) – whether the user's account has been verified via e-mail
- **floating_ip_limit** (*int*) – the maximum number of floating IPs the account may have at any one time
- **status** (*string*) – the status of the account: `"active"`, `"warning"`, or `"locked"`
- **status_message** – a human-readable string describing the status of the account
- **uuid** (*alphanumeric string*) – a UUID for the user

STATUS_ACTIVE = 'active'

The status of an account that is currently active and warning-free

STATUS_WARNING = 'warning'

The status of an account that is currently in a "warning" state, e.g., from having reached the droplet limit

STATUS_LOCKED = 'locked'

The status of a locked account

fetch()

Fetch & return a new *Account* object representing the account's current state

Return type *Account*

Raises ***DOAPIError*** – if the API endpoint replies with an error

url

The endpoint for operations on the user's account

1.9.2 Action

class doapi.**Action**

An action resource, representing a change made to another resource.

Actions are created in response to almost all mutating requests on droplets, images, and floating IPs, and they can be retrieved with the *doapi.fetch_action()*, *doapi.fetch_last_action()*, *doapi.fetch_all_actions()* methods as well as the `fetch_all_actions`, `fetch_last_action`, and `fetch_current_action` methods of *Droplet*, *Image*, and *FloatingIP*.

The DigitalOcean API specifies the following fields for action objects:

Variables

- **id** (*int*) – a unique identifier for the action

- **completed_at** (datetime.datetime or None) – date & time of the action's completion, or None if the action has not completed yet
- **region** (*Region*) – the region in which the action occurred
- ***region_slug*** (*string*) – the unique slug identifier for the region in which the action occurred
- **resource_id** (*int*) – the unique ID of the resource that the action operated on. If the resource was a droplet or image, this will be its id field. If the resource was a floating IP, this will be the IP address as a 32-bit integer.
- **resource_type** (*string*) – the type of resource that the action operated on: "droplet", "image", or "floating_ip"
- **started_at** (*datetime.datetime*) – date & time of the action's initiation
- **status** (*string*) – the current status of the action: "in-progress", "completed", or "errored"
- **type** (*string*) – the type of action performed

__int__()
Convert the resource to its unique integer ID

STATUS_IN_PROGRESS = 'in-progress'
The status of actions that are currently still in progress

STATUS_COMPLETED = 'completed'
The status of actions that have completed successfully

STATUS_ERRORED = 'errored'
The status of actions that failed to complete successfully

completed
True iff the action's status is "completed"

in_progress
True iff the action's status is "in-progress"

done
True iff the action's status is *not* "in-progress"

errored
True iff the action's status is "errored"

url
The endpoint for general operations on the individual action

fetch()
Fetch & return a new *Action* object representing the action's current state

Return type *Action*

Raises ***DOAPIError*** – if the API endpoint replies with an error

fetch_resource()
Fetch & return the resource that the action operated on, or None if the resource no longer exists (specifically, if the API returns a 404)

Return type *Droplet*, *Image*, *FloatingIP*, or None

Raises

- **ValueError** – if the action has an unknown `resource_type` (This indicates a deficiency in the library; please report it!)
- ***DOAPIError*** – if the API endpoint replies with a non-404 error

wait(*wait_interval=None*, *wait_time=None*)

Poll the server periodically until the action has either completed or errored out and return its final state.

If `wait_time` is exceeded, a *WaitTimeoutError* (containing the action's most recently fetched state) is raised.

If a `KeyboardInterrupt` is caught, the action's most recently fetched state is returned immediately without waiting for completion.

Changed in version 0.2.0: Raises *WaitTimeoutError* on timeout

Parameters

- **wait_interval** (*number*) – how many seconds to sleep between requests; defaults to the *doapi* object's *wait_interval* if not specified or `None`
- **wait_time** (*number*) – the total number of seconds after which the method will raise an error if the action has not yet completed, or a negative number to wait indefinitely; defaults to the *doapi* object's *wait_time* if not specified or `None`

Returns the action's final state

Return type *Action*

Raises

- ***DOAPIError*** – if the API endpoint replies with an error
- ***WaitTimeoutError*** – if `wait_time` is exceeded

raise_for_error()

New in version 0.2.0.

If the action's status is `"errored"`, raise an *ActionError*. Otherwise, do nothing.

Returns `None`

Raises ***ActionError*** – if the action's status is `"errored"`

1.9.3 Region

class doapi.**Region**

A region resource, representing a physical datacenter in which droplets can be located.

Available regions can be retreived with the *doapi.fetch_all_regions()* method.

The DigitalOcean API specifies the following fields for region objects:

Variables

- **available** (*bool*) – whether new droplets can be created in the region
- **features** (*list of strings*) – a list of strings naming the features available in the region
- **name** (*string*) – a human-readable name for the region
- **sizes** (*list of strings*) – the slugs of the sizes available in the region
- **slug** (*string*) – the unique slug identifier for the region

__str__()
Convert the region to its slug representation

1.9.4 Size

class doapi.**Size**
A size resource, representing an option for the amount of RAM, disk space, etc. provisioned for a droplet.

Available sizes can be retreived with the *doapi.fetch_all_sizes()* method.

The DigitalOcean API specifies the following fields for size objects:

Variables

- **available** (*bool*) – whether new droplets can be created with this size
- **disk** (*number*) – disk size of a droplet of this size in gigabytes
- **memory** (*number*) – RAM of a droplet of this size in megabytes
- **price_hourly** (*number*) – the hourly cost for a droplet of this size in USD
- **price_monthly** (*number*) – the monthly cost for a droplet of this size in USD
- **regions** (*list of strings*) – the slugs of the regions in which this size is available
- **slug** (*string*) – the unique slug identifier for the size
- ***transfer*** (*number*) – the amount of transfer bandwidth in terabytes available for a droplet of this size
- **vcpus** (*int*) – the number of virtual CPUs on a droplet of this size

__str__()
Convert the size to its slug representation

1.10 Utility Classes

1.10.1 Exceptions

ActionError

exception doapi.**ActionError**(*action*)
Bases: exceptions.Exception

New in version 0.2.0.

Raised when *raise_for_error()* is called on an *Action* that failed to complete successfully

action = None
The *Action* that failed

DOAPIError

exception doapi.**DOAPIError**(*response*)
Bases: exceptions.Exception

An exception raised in reaction to the API endpoint responding with a 4xx or 5xx error. Any method that performs an API request may raise this error.

If the body of the error response is a JSON object, its fields will be added to the DOAPIError's attributes (except where a pre-existing attribute would be overwritten). DigitalOcean error response bodies have been observed to consist of an object with two string fields, "id" and "message".

Note that this class is only for representing errors reported by the endpoint in response to API requests. Everything else that can go wrong uses the normal Python exceptions.

response = None

The requests.Response object

http_error_msg = None

An error message that should be appropriate for human consumption, containing the type of HTTP error, the URL that was requested, and the body of the response.

WaitTimeoutError

exception doapi.**WaitTimeoutError**(*in_progress*, *attr*, *value*, *wait_interval*, *wait_time*)

Bases: exceptions.Exception

New in version 0.2.0.

Raised when the runtime of a wait method exceeds wait_time

in_progress = None

A list of any waited-on objects that have not yet completed

attr = None

The objects' attribute that was being monitored

value = None

The desired value for the objects' *attr* attribute

wait_interval = None

The wait_interval value for the wait operation

wait_time = None

The wait_time value for the wait operation

1.10.2 DOEncoder

class doapi.**DOEncoder**(*skipkeys=False*, *ensure_ascii=True*, *check_circular=True*, *allow_nan=True*, *sort_keys=False*, *indent=None*, *separators=None*, *encoding='utf-8'*, *default=None*)

Bases: json.encoder.JSONEncoder

A json.JSONEncoder subclass that converts resource objects to dicts for JSONification. It also converts iterators to lists.

default(*obj*)

Implement this method in a subclass such that it returns a serializable object for o, or calls the base implementation (to raise a TypeError).

For example, to support arbitrary iterators, you could implement default like this:

```
def default(self, o):
    try:
        iterable = iter(o)
    except TypeError:
        pass
```

(continues on next page)

(continued from previous page)

```
    else:
        return list(iterable)
    # Let the base class default method raise the TypeError
    return JSONEncoder.default(self, o)
```

1.11 Examples

```
# Shut.
# Down.
# EVERYTHING.

actions = []

for droplet in manager.fetch_all_droplets():
    if droplet.active:  # Only droplets that are already up can be shut down.
        actions.append(droplet.shutdown())

# `list` forces the generator returned by `wait_actions` to be fully
# evaluated, thereby making Python wait until all of the shutdown actions
# have completed.
list(manager.wait_actions(actions))
```

CHAPTER 2

Command-Line Programs

2.1 Common CLI Behavior

Todo: Document the plain **doapi** command

2.1.1 API Authentication

In order to perform API requests, an OAuth token must be supplied to **doapi** so that it can authenticate with the DigitalOcean server. You can generate an OAuth token for your account via the "Apps & API" section of your DigitalOcean account's control panel.

The **doapi** commands will look for an OAuth token in the following locations, in order:

1. Specified on the command line with `--api-token` *token* or `--api-token-file` *file*
2. The value of the `DO_API_TOKEN` environment variable
3. The contents of a `.doapi` file in your home directory

2.1.2 Handling Non-Uniqueness of Identifiers

Most resources in the API are referred to by only a single identifier each, but droplets, images, and SSH keys can be referred to by either a unique ID number, a unique slug (for certain images), a unique fingerprint (for SSH keys), or a (potentially) *non*-unique name.

When the user specifies an object identifier that could be an ID, slug, fingerprint, or name, the default order of resolution is as follows:

1. If the identifier is an integer and there is an object of the relevant type with that integer as its ID number, **doapi** uses that object.
2. For images, if there is an image whose slug equals the identifier, **doapi** uses that image.

3. For SSH keys, if the identifier is in the form of an SSH public key fingerprint (16 colon-separated two-digit hexadecimal integers) and there is an SSH key with that fingerprint registered with the user's DigitalOcean account, **doapi** uses that SSH key.
4. Otherwise, the identifier is assumed to be a name, and if there exists exactly one object of the relevant type with that name, **doapi** uses that object. If there is not exactly one object with that name, **doapi** will exit without taking any action; if the name is shared by multiple objects, the error message will include all of the objects' ID numbers.

For subcommands that operate on lists of objects, this behavior can be changed by passing the `--multiple` option to the subcommand, causing any identifier shared by multiple objects of the relevant type (whether as ID, slug, fingerprint, and/or name) to be interpreted as a list of all of those objects in unspecified order.

In all cases, if the same object is specified more than once on the command line, all occurrences after the first are ignored with a warning.

Changed in version 0.2.0: `--multiple` now also matches by ID, slug, & fingerprint

Todo: Mention `--unique` and the warnings in its absence

2.1.3 Universal Options

All commands take the following options in addition to those listed in their individual documentation:

--api-token `<token>`
: Use `<token>` as an OAuth token for authentication with the API; mutually exclusive with `--api-token-file`

--api-token-file `<file>`
: Use the contents of `<file>` (after stripping leading & trailing whitespace) as an OAuth token for authentication with the API; mutually exclusive with `--api-token`

--endpoint `<URL>`
: Use `<URL>` as the base URL for all API requests; default value: `https://api.digitalocean.com` (the official DigitalOcean API endpoint)

--help
: Show command usage and exit

--timeout `<seconds>`
: The maximum number of seconds to wait when attempting to connect to or read from the remote endpoint; default value: no timeout

--version
: Show doapi version and exit

Note that these options (other than *`--help`*) cannot be attached to subcommands:

```
doapi-droplet --timeout 1000 show  # Good
doapi --timeout 1000 droplet show  # Good
doapi-droplet show --timeout 1000  # Bad!
```

2.1.4 Waiting Options

By default, all subcommands that perform non-atomic actions return immediately after initiating the action, without waiting for it to complete. They can be made to instead wait until completion with the *`--wait`* option, which can be configured further with *--wait-interval* and *--wait-time*, as described below:

-w, --wait

Periodically poll the server for the current status of all actions until they all complete or error out or until the time limit specified by *--wait-time* is exceeded. If this action is not specified, the subcommand will exit immediately after initiating the actions.

--wait-interval <seconds>

How often to poll the server for the actions' current statuses; default value: 2 seconds

Changed in version 0.2.0: Default value changed from 5 seconds to 2 seconds

--wait-time <seconds>

The maximum number of seconds to wait for all actions to complete. After this much time has passed since program invocation, any remaining in-progress actions will be output immediately without waiting for them to finish.

If *--wait* is specified but this option is not, the subcommand will wait indefinitely.

2.2 doapi-account

2.2.1 NAME

doapi-account — fetch DigitalOcean account data

2.2.2 SYNOPSIS

```
doapi-account [<universal options>] [--rate-limit]
```

2.2.3 DESCRIPTION

doapi-account displays the current information about your DigitalOcean account as available via the API. The account information is output as an *Account* object converted to JSON.

2.2.4 OPTIONS

In addition to the *universal options* common to all **doapi** commands, **doapi-account** takes the following:

--rate-limit

Fetch the account data, but instead of displaying it, output the rate limit headers from the response. (The account data is only requested so that there's a response to get the headers from.)

2.3 doapi-action

2.3.1 NAME

doapi-action — manage DigitalOcean API actions

2.3.2 SYNOPSIS

```
doapi-action show [<action> ...]
doapi-action show {--in-progress | --last}
doapi-action wait [--wait-time <seconds>] [--wait-interval <seconds>] [<action> ...]
doapi-action resource <action> ...
doapi-action resource {--in-progress | --last}
```

doapi-action also takes the *universal options* common to all **doapi** commands.

Actions are specified by ID number.

2.3.3 show

```
doapi-action show [<action> ...]
doapi-action show {--in-progress | --last}
```

Show the given actions. If no actions or flags are specified, all actions ever performed on the account are shown. The actions are output as a list of *Action* objects converted to JSON.

Options

--in-progress
Show only the currently in-progress actions

--last
Show only the most recent action performed on the account instead of a list of all actions. If multiple actions were triggered simultaneously, the choice of which to display is undefined.

2.3.4 wait

```
doapi-action wait [--wait-time <seconds>] [--wait-interval <seconds>] [<action> ...]
```

Wait for the given actions to either complete or error out. The finished actions are output as a list of *Action* objects converted to JSON, with each action output (roughly) as soon as it finishes. If no actions are specified, **wait** will wait for all currently in-progress actions to complete.

Options

--wait-interval <seconds>
How often to poll the server for the actions' current statuses; default value: 5 seconds

--wait-time <seconds>
The maximum number of seconds to wait for all actions to complete. After this much time has passed since program invocation, any remaining in-progress actions will be output immediately without waiting for them to finish.

If this option is not specified, **wait** will wait indefinitely.

2.3.5 `resource`

```
doapi-action resource <action> ...
doapi-action resource {--in-progress | --last}
```

Show the resources that the specified actions operated on. The resources are output as a list of *`Droplet`*, *`Image`*, and/or *`FloatingIP`* objects converted to JSON. Resources that no longer exist are output as `null`.

Options

`--in-progress`
Show only the resources currently being acted upon

`--last`
Show only the resource operated on by the account's most recent action. If multiple actions were triggered simultaneously, the choice of which to display is undefined. If no actions have ever been performed on the account, the output is `null`.

2.4 `doapi-domain`

2.4.1 NAME

`doapi-domain` — manage DigitalOcean domains & domain records

2.4.2 SYNOPSIS

```
doapi-domain new <domain> <ip address>
doapi-domain show [<domain> ...]
doapi-domain delete <domain> ...

doapi-domain new-record [--priority <int>] [--port <int>] [--weight <int>]
                        [--delete] <domain> <type> <name> <data>

doapi-domain show-record <domain> [<record id> ...]

doapi-domain update-record [--type <type>]
                           [--name <name>]
                           [--data <data>]
                           [--port     <int> | --no-port]
                           [--priority <int> | --no-priority]
                           [--weight   <int> | --no-weight]
                           <domain> <record id>

doapi-domain delete-record <domain> <record id> [<record id> ...]
```

`doapi-domain` also takes the *universal options* common to all **`doapi`** commands.

Domains are specified as the base domain name without any subdomains (e.g., `example.com`, never `www.example.com`). Records of a given domain are specified by ID number.

2.4.3 `new`

```
doapi-domain new <domain> <ip address>
```

Set up a domain name `<domain>` pointing to `<ip address>`. The new domain is output as a *`Domain`* object converted to JSON.

Note that this command does not actually register a new domain name; it merely configures DigitalOcean's nameservers to provide DNS resolution for the domain. See How To Set Up a Host Name with DigitalOcean for more information.

2.4.4 `show`

```
doapi-domain show [<domain> ...]
```

Show domains. If no domains are specified, all domains registered to the account are shown. The domains are output as a list of *`Domain`* objects converted to JSON.

2.4.5 `delete`

```
doapi-domain delete <domain> ...
```

Delete domains. If any of the given domains do not exist, nothing is deleted. There is no output.

2.4.6 `new-record`

```
doapi-domain new-record [--priority <int>] [--port <int>] [--weight <int>]
                        [--delete] <domain> <type> <name> <data>
```

Add a new domain record with the given type, name, & data to domain `<domain>`. The new record is output as a *`DomainRecord`* object converted to JSON.

Options

`--delete`
After creating the new record, delete any old records with the same type & name.

`--port` `<int>`
Specify the port on which the service is available (SRV records only)

`--priority` `<int>`
Specify the priority for the new record (SRV and MX records only)

`--weight` `<int>`
Specify the weight for the new record (SRV records only)

2.4.7 `show-record`

```
doapi-domain show-record <domain> [<record id> ...]
```

Show records for domain <domain>. If no records are specified, all records for the domain are shown. The records are output as a list of *DomainRecord* objects converted to JSON.

2.4.8 update-record

```
doapi-domain update-record [--type <type>]
                           [--name <name>]
                           [--data <data>]
                           [--port     <int> | --no-port]
                           [--priority <int> | --no-priority]
                           [--weight   <int> | --no-weight]
                           <domain> <record id>
```

Modify one or more fields of a domain record. The updated record is output as a *DomainRecord* object converted to JSON.

Options

--data <data>
Set the record's data to <data>

--name <name>
Set the record's name to <name>

--no-port
Unset the record's port field

--no-priority
Unset the record's priority field

--no-weight
Unset the record's weight field

--port <int>
Set the record's port to <int>

--priority <int>
Set the record's priority to <int>

--type <type>
Set the record's type to <type>

--weight <int>
Set the record's weight to <int>

2.4.9 delete-record

```
doapi-domain delete-record <domain> <record id> [<record id> ...]
```

Delete records of the given domain. If any of the given records do not exist, nothing is deleted. There is no output.

2.5 doapi-droplet

2.5.1 NAME

doapi-droplet — manage DigitalOcean droplets

2.5.2 SYNOPSIS

```
doapi-droplet new
        -I|--image <image>
        -S|--size <size>
        -R|--region <region>
        [-B|--backups]
        [-6|--ipv6]
        [-P|--private-networking]
        [-U|--user-data <string|@file>]
        [-K|--ssh-key <key>] ...
        [--unique]
        [-w|--wait] [--wait-time <seconds>] [--wait-interval <seconds>]
        <name> ...

doapi-droplet show [-M|--multiple] [--tag <tag> | <droplet> ...]

doapi-droplet enable-backups [<wait options>] [-M|--multiple] {--tag <tag> | <droplet>
↪ ...}
doapi-droplet disable-backups [<wait options>] [-M|--multiple] {--tag <tag> |
↪<droplet> ...}
doapi-droplet reboot [<wait options>] [-M|--multiple] <droplet> ...
doapi-droplet power-cycle [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ..
↪.}
doapi-droplet shutdown [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet power-off [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet power-on [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet password-reset [<wait options>] [-M|--multiple] <droplet> ...
doapi-droplet enable-ipv6 [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ..
↪.}
doapi-droplet enable-private-networking [<wait options>] [-M|--multiple] {--tag <tag>␣
↪| <droplet> ...}

doapi-droplet show-snapshots [-M|--multiple] <droplet> ...
doapi-droplet backups [-M|--multiple] <droplet> ...
doapi-droplet kernels [-M|--multiple] <droplet> ...
doapi-droplet delete [-M|--multiple] {--tag <tag> | <droplet> ...}

doapi-droplet neighbors
doapi-droplet neighbors [-M|--multiple] <droplet> ...

doapi-droplet restore [<wait options>] <droplet> <backup>

doapi-droplet resize [<wait options>] [--disk] [-M|--multiple] <size> <droplet> ...

doapi-droplet rebuild [<wait options>] [-I|--image <image>] [-M|--multiple] <droplet>␣
↪...

doapi-droplet rename [<wait options>] [--unique] <droplet> <new name>
```

(continues on next page)

(continued from previous page)

```
doapi-droplet snapshot [<wait options>] [--unique] <name> {--tag <tag> | <droplet> ...
↪}
doapi-droplet change-kernel [<wait options>] [-M|--multiple] <kernel> <droplet> ...

doapi-droplet act [<wait options>] [-p|--params <JSON|@file>] [-M|--multiple] <type>
↪{--tag <tag> | <droplet> ...}
doapi-droplet actions [--last | --in-progress] [-M|--multiple] <droplet> ...
doapi-droplet wait [--wait-time <seconds>] [--wait-interval <seconds>] [-M|--
↪multiple] [-S|--status <status> | --locked | --unlocked] <droplet> ...

doapi-droplet tag [-M|--multiple] <tag> <droplet> ...
doapi-droplet untag [-M|--multiple] <tag> <droplet> ...
```

doapi-droplet also takes the *universal options* common to all **doapi** commands.

Droplets can be specified by ID number or name. A name that is also a valid ID is interpreted as such rather than as a name (and so droplets with such names must be referred to by their actual ID instead).

2.5.3 new

```
doapi-droplet new
        -I|--image <image>
        -S|--size <size>
        -R|--region <region>
        [-B|--backups]
        [-6|--ipv6]
        [-P|--private-networking]
        [-U|--user-data <string|@file>]
        [-K|--ssh-key <key>] ...
        [--unique]
        [-w|--wait] [--wait-time <seconds>] [--wait-interval <seconds>]
        <name> ...
```

2.5.4 show

```
doapi-droplet show [-M|--multiple] [--tag <tag> | <droplet> ...]
```

2.5.5 Simple Actions

```
doapi-droplet enable-backups [<wait options>] [-M|--multiple] {--tag <tag> | <droplet>
↪ ...}
doapi-droplet disable-backups [<wait options>] [-M|--multiple] {--tag <tag> |
↪<droplet> ...}
doapi-droplet reboot [<wait options>] [-M|--multiple] <droplet> ...
doapi-droplet power-cycle [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ..
↪.}
doapi-droplet shutdown [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet power-off [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet power-on [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ...}
doapi-droplet password-reset [<wait options>] [-M|--multiple] <droplet> ...
```

(continues on next page)

(continued from previous page)

```
doapi-droplet enable-ipv6 [<wait options>] [-M|--multiple] {--tag <tag> | <droplet> ..
↪.}
doapi-droplet enable-private-networking [<wait options>] [-M|--multiple] {--tag <tag>␣
↪| <droplet> ...}
```

2.5.6 show-snapshots

```
doapi-droplet show-snapshots [-M|--multiple] <droplet> ...
```

2.5.7 backups

```
doapi-droplet backups [-M|--multiple] <droplet> ...
```

2.5.8 kernels

```
doapi-droplet kernels [-M|--multiple] <droplet> ...
```

2.5.9 neighbors

```
doapi-droplet neighbors
doapi-droplet neighbors [-M|--multiple] <droplet> ...
```

2.5.10 restore

```
doapi-droplet restore [<wait options>] <droplet> <backup>
```

2.5.11 resize

```
doapi-droplet resize [<wait options>] [--disk] [-M|--multiple] <size> <droplet> ...
```

2.5.12 rebuild

```
doapi-droplet rebuild [<wait options>] [-I|--image <image>] [-M|--multiple] <droplet>␣
↪...
```

2.5.13 rename

```
doapi-droplet rename [<wait options>] [--unique] <droplet> <new name>
```

2.5.14 snapshot

```
doapi-droplet snapshot [<wait options>] [--unique] <name> {--tag <tag> | <droplet> ...
↪}
```

2.5.15 change-kernel

```
doapi-droplet change-kernel [<wait options>] [-M|--multiple] <kernel> <droplet> ...
```

2.5.16 act

```
doapi-droplet act [<wait options>] [-p|--params <JSON|@file>] [-M|--multiple] <type>
↪{--tag <tag> | <droplet> ...}
```

2.5.17 actions

```
doapi-droplet actions [--last | --in-progress] [-M|--multiple] <droplet> ...
```

2.5.18 wait

```
doapi-droplet wait [--wait-time <seconds>] [--wait-interval <seconds>] [-M|--
↪multiple] [-S|--status <status> | --locked | --unlocked] <droplet> ...
```

2.5.19 delete

```
doapi-droplet delete [-M|--multiple] {--tag <tag> | <droplet> ...}
```

2.5.20 tag

```
doapi-droplet tag [-M|--multiple] <tag> <droplet> ...
```

2.5.21 untag

```
doapi-droplet untag [-M|--multiple] <tag> <droplet> ...
```

2.6 doapi-floating-ip

2.6.1 NAME

doapi-floating-ip — manage DigitalOcean floating IP addresses

2.6.2 SYNOPSIS

```
doapi-floating-ip new      [<wait options>] -D|--droplet <droplet>
doapi-floating-ip new      [<wait options>] -R|--region <region>

doapi-floating-ip show     [<ip> ...]

doapi-floating-ip assign   [<wait options>] <ip> <droplet>
doapi-floating-ip unassign [<wait options>] <ip> ...

doapi-floating-ip act      [<wait options>] [-p|--params <JSON|@file>] <type> <ip> ...
doapi-floating-ip actions  [--in-progress | --last] <ip> ...
doapi-floating-ip wait     [--wait-interval <seconds>] [--wait-time <seconds>] <ip> ..
↪.

doapi-floating-ip delete   <ip> ...
```

doapi-floating-ip also takes the *universal options* common to all **doapi** commands.

Floating IPs can be specified in either dotted-quad notation (e.g., `127.0.0.1`) or as 32-bit integers (e.g., `2130706433`).

2.6.3 new

```
doapi-floating-ip new [<wait options>] -D|--droplet <droplet>
doapi-floating-ip new [<wait options>] -R|--region <region>
```

Create a new floating IP assigned to a droplet or reserved to a region. The new IP is output as a `FloatingIP` object converted to JSON.

Options

In addition to the *waiting options*, the **new** subcommand takes the following:

-D <droplet>, **--droplet** <droplet>
Assign the new floating IP to the specified droplet (identified by ID or name). The floating IP will also be reserved to the droplet's region.

-R <region>, **--region** <region>
Reserve the floating IP to the specified region (identified by slug).

2.6.4 show

```
doapi-floating-ip show [<ip> ...]
```

Show floating IPs. If no IPs are specified, all IPs allocated to the account are shown. The IPs are output as a list of `FloatingIP` objects converted to JSON.

2.6.5 assign

```
doapi-floating-ip assign [<wait options>] <ip> <droplet>
```

Assign the given floating IP to a given droplet (identified by ID or name). The `Action` object thus produced is output as JSON.

The **assign** subcommand only takes the *waiting options*.

2.6.6 unassign

```
doapi-floating-ip unassign [<wait options>] <ip> ...
```

Unassign the given floating IP(s) from their droplet(s). The `Action` objects thus produced are output as a JSON list.

The **unassign** subcommand only takes the *waiting options*.

2.6.7 act

```
doapi-floating-ip act [<wait options>] [-p|--params <JSON|@file>] <type> <ip> ...
```

Perform an arbitrary action of type `<type>` (`assign`, `unassign`, or something otherwise not implemented here) on one or more floating IPs. The `Action` objects thus produced are output as a JSON list.

Options

In addition to the *waiting options*, the **act** subcommand takes the following:

-p <data>, **--params** <data>
A JSON object/dictionary of parameters to the action. If `<data>` begins with "`@`", the rest of the argument (if there is any) is treated as a file from which to read the JSON; a filename of `-` causes data to be read from standard input.

2.6.8 actions

```
doapi-floating-ip actions [--in-progress | --last] <ip> ...
```

List all of the actions that have ever been performed on the given floating IP(s). The results are output as a JSON list containing a sublist of `Action` objects for each IP specified on the command line, in order.

Options

--in-progress
Show only the currently in-progress action on each floating IP instead of a list of all actions. If there is currently no in-progress action on an IP, show `null`.

--last
Show only the most recent action on each floating IP instead of a list of all actions. If multiple actions on a single IP were triggered simultaneously, the choice of which to return is undefined. If no actions were ever performed on a floating IP, show `null`.

2.6.9 wait

```
doapi-floating-ip wait [--wait-interval <seconds>] [--wait-time <seconds>] <ip> ...
```

Wait for the currently in-progress actions on the given floating IP(s) to either complete or error out. The finished actions are output as a list of *Action* objects converted to JSON, with each action output (roughly) as soon as it finishes. If there are no actions currently in progress on a given floating IP, nothing will be output for it.

Options

--wait-interval <seconds>
How often to poll the server for the actions' current statuses; default value: 2 seconds

--wait-time <seconds>
The maximum number of seconds to wait for all actions to complete. After this much time has passed since program invocation, any remaining in-progress actions will be output immediately without waiting for them to finish.

If this option is not specified, **wait** will wait indefinitely.

2.6.10 delete

```
doapi-floating-ip delete <ip> ...
```

Delete floating IPs. If any of the given floating IPs do not exist, nothing is deleted. There is no output.

2.7 doapi-image

2.7.1 NAME

doapi-image — manage DigitalOcean droplet images

2.7.2 SYNOPSIS

```
doapi-image show     [-M|--multiple] [<image> ...]
doapi-image show     {--application|--distribution|--type=<type>|--private}

doapi-image update   [--unique] <image> <new name>

doapi-image convert  [<wait options>] [-M|--multiple] <image> ...
doapi-image transfer [<wait options>] [-M|--multiple] <region> <image> ...

doapi-image act      [<wait options>] [-p|--params <JSON|@file>] [-M|--multiple]
↪<type> <image> ...
doapi-image actions  [--in-progress | --last] [-M|--multiple] <image> ...
doapi-image wait     [--wait-time <seconds>] [--wait-interval <seconds>] [-M|--
↪multiple] <image> ...

doapi-image delete   [-M|--multiple] <image> ...
```

doapi-image also takes the *universal options* common to all **doapi** commands.

Images can be specified by ID number, slug (if any), or name. A name that is also a valid ID or slug is interpreted as such rather than as a name (and so images with such names must be referred to by their ID instead).

2.7.3 show

```
doapi-image show [-M|--multiple] [<image> ...]
doapi-image show {--application|--distribution|--type=<type>|--private}
```

Show images. If no images or flags are specified, all images available to the account are shown. The images are output as a list of `Image` objects converted to JSON.

Options

--application
Only show application images

--distribution
Only show distribution images

-M, --multiple
Arguments that could refer to multiple images are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

--private
Only show the user's private images

--type=<type>
Only show images of type `<type>` (`application`, `distribution`, or something not otherwise implemented here)

2.7.4 update

```
doapi-image update [--unique] <image> <new name>
```

Update (i.e., rename) an image. The updated image is output as an `Image` object converted to JSON.

Options

--unique
If `<new name>` is already in use by another image, fail with an error. Without this option, a warning will still be generated if `<new name>` is already in use.

2.7.5 convert

```
doapi-image convert [<wait options>] [-M|--multiple] <image> ...
```

Convert one or more images to snapshots. The `Action` objects thus produced are output as a JSON list.

Options

In addition to the *waiting options*, the **convert** subcommand takes the following:

-M, --multiple
Arguments that could refer to multiple images are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

2.7.6 transfer

```
doapi-image transfer [<wait options>] [-M|--multiple] <region> <image> ...
```

Transfer one or more images to another region (identified by its slug). The `Action` objects thus produced are output as a JSON list.

Options

In addition to the *waiting options*, the **transfer** subcommand takes the following:

-M, --multiple
Arguments that could refer to multiple images are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

2.7.7 act

```
doapi-image act [<wait options>] [-p|--params <JSON|@file>] [-M|--multiple] <type>
↪<image> ...
```

Perform an arbitrary action of type `<type>` (`convert`, `transfer`, or something otherwise not implemented here) on one or more images. The `Action` objects thus produced are output as a JSON list.

Options

In addition to the *waiting options*, the **act** subcommand takes the following:

-M, --multiple
Arguments that could refer to multiple images are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

-p `<data>`**, --params** `<data>`
A JSON object/dictionary of parameters to the action. If `<data>` begins with "`@`", the rest of the argument (if there is any) is treated as a file from which to read the JSON; a filename of `-` causes data to be read from standard input.

2.7.8 actions

```
doapi-image actions [--in-progress | --last] <image> ...
```

List all of the actions that have ever been performed on the given image(s). The results are output as a JSON list containing a sublist of `Action` objects for each image specified on the command line, in order.

Options

--in-progress

Show only the currently in-progress action on each image instead of a list of all actions. If there is currently no in-progress action on an image, show `null`.

--last

Show only the most recent action on each image instead of a list of all actions. If multiple actions on a single image were triggered simultaneously, the choice of which to return is undefined. If no actions were ever performed on an image, show `null`.

2.7.9 `wait`

```
doapi-image wait [--wait-interval <seconds>] [--wait-time <seconds>] <image> ...
```

Wait for the currently in-progress actions on the given image(s) to either complete or error out. The finished actions are output as a list of *`Action`* objects converted to JSON, with each action output (roughly) as soon as it finishes. If there are no actions currently in progress on a given image, nothing will be output for it.

Options

--wait-interval `<seconds>`

How often to poll the server for the actions' current statuses; default value: 2 seconds

--wait-time `<seconds>`

The maximum number of seconds to wait for all actions to complete. After this much time has passed since program invocation, any remaining in-progress actions will be output immediately without waiting for them to finish.

If this option is not specified, **`wait`** will wait indefinitely.

2.7.10 `delete`

```
doapi-image delete [-M|--multiple] <image> ...
```

Delete images. If any of the given images do not exist, nothing is deleted. There is no output.

Options

-M, --multiple

Arguments that could refer to multiple images are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

2.8 `doapi-region`

2.8.1 NAME

`doapi-region` — list available DigitalOcean droplet regions

2.8.2 SYNOPSIS

```
doapi-region [<universal options>]
```

2.8.3 DESCRIPTION

doapi-region lists all of the droplet regions available to your DigitalOcean account. The regions are output as a list of *Region* objects converted to JSON.

2.8.4 OPTIONS

doapi-region only takes the *universal options* common to all **doapi** commands.

2.9 doapi-request

2.9.1 NAME

doapi-request — perform a raw DigitalOcean API request

2.9.2 SYNOPSIS

```
doapi-request [<universal options>]
              [-d|--data <string|@file>]
              [-D|--dump-header <file>]
              [--paginate <key>]
              [-X|--request <method>]
              <URL>|<path>
```

2.9.3 DESCRIPTION

doapi-request makes a request to a DigitalOcean API endpoint and outputs the text of the response (if any) as nicely-formatted JSON. It's like a DigitalOcean-centric *curl(1)* that doesn't make you type out your API token every time, with an option for handling pagination.

A request can be made either to an absolute URL or to a path like `/v2/account` that will be appended to the API endpoint in use.

2.9.4 OPTIONS

In addition to the *universal options* common to all **doapi** commands, **doapi-request** takes the following:

-d <data>, **--data** <data>

Send the given data (which must be valid JSON) in the body of the request. If `<data>` begins with "`@`", the rest of the argument (if there is any) is treated as a file from which to read the data; a filename of `-` causes data to be read from standard input.

--data and *--paginate* are mutually exclusive.

-D <file>, **--dump-header** <file>

Dump the headers of the HTTP response as a JSON object to the given file. If *--paginate* is also used, only the headers for the last response will be dumped. <file> may be - to write the headers to standard output.

--paginate <key>

(GET method only) Assume that the API will respond with a paginated list of one or more values, i.e., an object containing a field <key> mapped to a list of values and a field "links" containing a URL for retrieving the next set of values in the result list, if any. Instead of performing a single request and outputting the response body, **doapi-requests** will perform multiple requests to retrieve all of the pages and will output a concatenated list of all of the values in the <key> field of each page.

--data and *--paginate* are mutually exclusive.

-X <method>, **--request** <method>

Specifies the HTTP method to use for the request. Valid options are GET (the default), POST, PUT, and DELETE (case insensitive).

2.10 doapi-size

2.10.1 NAME

doapi-size — list available DigitalOcean droplet sizes

2.10.2 SYNOPSIS

```
doapi-size [<universal options>]
```

2.10.3 DESCRIPTION

doapi-size lists all of the droplet sizes available to your DigitalOcean account. The sizes are output as a list of *Size* objects converted to JSON.

2.10.4 OPTIONS

doapi-size only takes the *universal options* common to all **doapi** commands.

2.11 doapi-ssh-key

2.11.1 NAME

doapi-ssh-key — manage SSH public keys on DigitalOcean

2.11.2 SYNOPSIS

```
doapi-ssh-key new    [--unique]       <name> [<file>]
doapi-ssh-key show   [-M|--multiple] [<ssh key> ...]
doapi-ssh-key update [--unique]       <ssh key> <new name>
doapi-ssh-key delete [-M|--multiple] <ssh key> ...
```

`doapi-ssh-key` also takes the *universal options* common to all **`doapi`** commands.

SSH keys can be specified by ID number, fingerprint, or name. A name that is also a valid ID or fingerprint is interpreted as such rather than as a name (and so SSH keys with such names must be referred to by their ID or fingerprint instead).

2.11.3 `new`

```
doapi-ssh-key new [--unique] <name> [<file>]
```

Register the contents of `<file>` (or standard input if no file is specified) as a new SSH public key with name `<name>`. The new key is output as an *`SSHKey`* object converted to JSON.

Options

`--unique`
If `<name>` is already in use by another key, fail with an error. Without this option, a warning will still be generated if `<name>` is already in use.

2.11.4 `show`

```
doapi-ssh-key show [-M|--multiple] [<ssh key> ...]
```

Show SSH keys. If no keys are specified, all keys registered to the account are shown. The keys are output as a list of *`SSHKey`* objects converted to JSON.

Options

`-M, --multiple`
Arguments that could refer to multiple SSH keys are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

2.11.5 `update`

```
doapi-ssh-key update [--unique] <ssh key> <new name>
```

Update (i.e., rename) an SSH key. The updated key is output as an *`SSHKey`* object converted to JSON.

Options

`--unique`
If `<new name>` is already in use by another key, fail with an error. Without this option, a warning will still be generated if `<new name>` is already in use.

2.11.6 `delete`

```
doapi-ssh-key delete [-M|--multiple] <ssh key> ...
```

Delete SSH keys. If any of the given SSH keys do not exist, nothing is deleted. There is no output.

Options

`-M, --multiple`
Arguments that could refer to multiple SSH keys are interpreted as such rather than using the default resolution rules; see *Handling Non-Uniqueness of Identifiers* for more information.

2.12 `doapi-tag`

2.12.1 NAME

`doapi-tag` — manage DigitalOcean tags

2.12.2 SYNOPSIS

```
doapi-tag new    <tag> ...
doapi-tag show   [<tag> ...]
doapi-tag update <tag> <new name>
doapi-tag delete <tag> ...
```

`doapi-tag` also takes the *universal options* common to all **`doapi`** commands.

2.12.3 `new`

```
doapi-tag new <tag> ...
```

Create one or more new tags with the given names. The new tags are output as *`Tag`* objects converted to JSON.

2.12.4 `show`

```
doapi-tag show [<tag> ...]
```

Show tags. If no tags are specified, all tags registered to the account are shown. The tags are output as a list of *`Tag`* objects converted to JSON.

To list the individual resources with a given tag, use the **`show`** subcommand for the relevant resource type; e.g., to list all of the droplets with a given tag, run `doapi-droplet show --tag=`*`tag`*.

2.12.5 `update`

```
doapi-tag update <tag> <new name>
```

Update (i.e., rename) a tag. The updated tag is output as a *`Tag`* object converted to JSON.

2.12.6 delete

```
doapi-tag delete <tag> ...
```

Delete tags. If any of the given tags do not exist, nothing is deleted. There is no output.

CHAPTER 3

Indices and tables

- genindex
- search

Bibliography

[APIDocs] https://developers.digitalocean.com/documentation/v2/

Python Module Index

Index