
dockerman Documentation

Release 0.1.2

Matt Bodenhamer

Oct 04, 2017

Contents

1	dockerman package	3
1.1	Subpackages	3
1.2	Submodules	4
1.3	dockerman.base module	4
1.4	dockerman.container module	4
1.5	dockerman.main module	6
1.6	dockerman.utils module	7
1.7	Module contents	7
2	Changelog	9
2.1	0.1.2 (2017-10-03)	9
2.2	0.1.1 (2017-10-03)	9
2.3	0.1 (2017-10-03)	9
3	Indices and tables	11
	Python Module Index	13

Manage docker images and docker containers used for local/desktop applications. Project is currently in pre-alpha.

Contents:

Subpackages

dockerman.tests package

Submodules

dockerman.tests.test_container module

dockerman.tests.test_main module

`dockerman.tests.test_main.test_main()`

dockerman.tests.test_utils module

`dockerman.tests.test_utils.test_dictify_strings()`

`dockerman.tests.test_utils.test_join()`

`dockerman.tests.test_utils.test_split()`

Module contents

Submodules

dockerman.base module

dockerman.container module

Representation of a Docker container.

class `dockerman.container.Container` (*image* [, *command*], ***kwargs*)

Bases: `syn.base.b.base.Base`

Positional Arguments:

image: *basestring* The image to run

command [Optional]: *basestring* The command to be run in the container

Keyword-Only Arguments:

_status: *ContainerStatus* **cpu_shares [Optional]:** *int*

CPU shares (relative weight)

detach (default = False): *bool* Detached mode

dns [Optional]: *list (basestring)* DNS name servers

domainname [Optional]: *basestring* Custom DNS search domains

entrypoint [Optional]: *basestring* Container entrypoint

environment [Optional]: *dict (any => basestring)* Environment variables to set in the container

hostname [Optional]: *basestring* Optional hostname for the container

id [Optional]: *basestring* The id of the running container

labels [Optional]: *dict (any => basestring)* A dictionary of name-value labels

mac_address [Optional]: *basestring* MAC address to assign to the container

mem_limit [Optional]: *float | basestring* Memory limit

memswap_limit [Optional]: *int* name: *basestring*

A name for the container

network_disabled (default = False): *bool* Disable networking

ports [Optional]: *list (int)* A list of port numbers

stdin_open (default = False): *bool* Keep STDIN open even if not attached

stop_signal [Optional]: *basestring* The signal used to stop the container

tty (default = False): *bool* Allocate a pseudo-TTY

user [Optional]: *basestring | int* Username or UID

volume_driver [Optional]: *basestring* The name of a volume driver/plugin

volumes [Optional]: *list (basestring)* Volume names

volumes_from [Optional]: *list (basestring)* List of container names or Ids to get volumes from

working_dir [Optional]: *basestring* Path to working directory

Class Options:

- args: ('image', 'command')
- autodoc: True
- coerce_args: True
- id_equality: False
- init_validate: True
- make_hashable: False
- make_type_object: True
- optional_none: True
- register_subclasses: False
- repr_template:
- coerce_hooks: ()
- create_hooks: ()
- init_hooks: ()
- init_order: ()
- metaclass_lookup: ('coerce_hooks', 'init_hooks', 'create_hooks', 'setstate_hooks')
- setstate_hooks: ()

Groups:

- _all: _status, command, cpu_shares, detach, dns, domainname, entrypoint, environment, hostname, id, image, labels, mac_address, mem_limit, memswap_limit, name, network_disabled, ports, stdin_open, stop_signal, tty, user, volume_driver, volumes, volumes_from, working_dir
- run_args: command, cpu_shares, detach, dns, entrypoint, environment, hostname, image, labels, mac_address, mem_limit, memswap_limit, name, stdin_open, stop_signal, tty, user, volume_driver, volumes, volumes_from, working_dir
- create_container: command, cpu_shares, detach, dns, domainname, entrypoint, environment, hostname, image, labels, mac_address, mem_limit, memswap_limit, name, network_disabled, ports, stdin_open, stop_signal, tty, user, volume_driver, volumes, volumes_from, working_dir
- _internal: id

is_port_live (*port*)

marshal_args (*group*)

pause (***kwargs*)

poll (*port, **kwargs*)

remove (***kwargs*)

run (***kwargs*)

start (***kwargs*)

status

stop (***kwargs*)

unpause (***kwargs*)

class dockerman.container.**ContainerStatus** (***kwargs*)

Bases: `syn.base.b.base.Base`

Keyword-Only Arguments:

dict: *dict* exists (*default* = False); *bool* id [**Optional**]: *basestring* ip_addr (*default* =): *basestring* paused (*default* = False); *bool* running (*default* = False); *bool*

Class Options:

- args: ()
- autodoc: True
- coerce_args: False
- id_equality: False
- init_validate: False
- make_hashable: False
- make_type_object: True
- optional_none: True
- register_subclasses: False
- repr_template:
- coerce_hooks: ()
- create_hooks: ()
- init_hooks: ()
- init_order: ()
- metaclass_lookup: ('coerce_hooks', 'init_hooks', 'create_hooks', 'setstate_hooks')
- setstate_hooks: ()

Groups:

- _all: dict, exists, id, ip_addr, paused, running

reset ()

`dockerman.container.container` (**args*, ***kws*)

dockerman.main module

`dockerman.main.main` (**args*)

dockerman.utils module

`dockerman.utils.join` (*obj=None, sep=' '*)

`dockerman.utils.split` (*obj=None, sep=None*)

`dockerman.utils.dictify_strings` (*obj=None, empty=True, sep=None*)

`dockerman.utils.call` (*s*)

`dockerman.utils.scan_port` (*addr, port*)

`dockerman.utils.container_exists` (*name, client=<docker.api.client.APIClient object>*)

Module contents

0.1.2 (2017-10-03)

Added Container.poll().

0.1.1 (2017-10-03)

Updated package metadata.

0.1 (2017-10-03)

Initial release.

CHAPTER 3

Indices and tables

- `genindex`
- `modindex`
- `search`

d

- `dockerman`, 7
- `dockerman.base`, 4
- `dockerman.container`, 4
- `dockerman.main`, 6
- `dockerman.tests`, 4
- `dockerman.tests.test_main`, 3
- `dockerman.tests.test_utils`, 3
- `dockerman.utils`, 7

C

call() (in module dockerman.utils), 7
Container (class in dockerman.container), 4
container() (in module dockerman.container), 6
container_exists() (in module dockerman.utils), 7
ContainerStatus (class in dockerman.container), 6

D

dictify_strings() (in module dockerman.utils), 7
dockerman (module), 7
dockerman.base (module), 4
dockerman.container (module), 4
dockerman.main (module), 6
dockerman.tests (module), 4
dockerman.tests.test_main (module), 3
dockerman.tests.test_utils (module), 3
dockerman.utils (module), 7

I

is_port_live() (dockerman.container.Container method), 5

J

join() (in module dockerman.utils), 7

M

main() (in module dockerman.main), 6
marshal_args() (dockerman.container.Container method), 5

P

pause() (dockerman.container.Container method), 5
poll() (dockerman.container.Container method), 5

R

remove() (dockerman.container.Container method), 5
reset() (dockerman.container.ContainerStatus method), 6
run() (dockerman.container.Container method), 5

S

scan_port() (in module dockerman.utils), 7
split() (in module dockerman.utils), 7
start() (dockerman.container.Container method), 5
status (dockerman.container.Container attribute), 5
stop() (dockerman.container.Container method), 6

T

test_dictify_strings() (in module dockerman.tests.test_utils), 3
test_join() (in module dockerman.tests.test_utils), 3
test_main() (in module dockerman.tests.test_main), 3
test_split() (in module dockerman.tests.test_utils), 3

U

unpause() (dockerman.container.Container method), 6