

Package: googleComputeEngineR (via r-universe)

October 31, 2024

Type Package

Version 0.3.0.9000

Title R Interface with Google Compute Engine

Description Interact with the 'Google Compute Engine' API in R. Lets you create, start and stop instances in the 'Google Cloud'. Support for preconfigured instances, with templates for common R needs.

URL <https://cloudyr.github.io/googleComputeEngineR/>

BugReports <https://github.com/cloudyr/googleComputeEngineR/issues>

Depends R (>= 3.3.0)

Imports assertthat, cli, gargle, future (>= 1.2.0), googleAuthR (>= 0.7.0), httr (>= 1.3.1), jsonlite (>= 1.1), utils

Suggests covr, googleCloudStorageR, knitr, rmarkdown, testthat

License MIT + file LICENSE

LazyData true

VignetteBuilder knitr

RoxygenNote 7.1.1

Repository <https://cloudyr.r-universe.dev>

RemoteUrl <https://github.com/cloudyr/googlecomputeengine>

RemoteRef HEAD

RemoteSha bf4e5e7333ae331332b6df71799b6bbcf0acee5c

Contents

as.cluster.gce_instance	4
containers	5
container_logs	5
container_rm	6

container_running	6
container_update_info	7
docker_build	7
docker_cmd	9
docker_cmd.gce_instance	10
docker_inspect	11
docker_pull	11
docker_run	12
gce_attach_disk	13
gce_auth	15
gce_check_gpu	15
gce_check_ssh	16
gce_container_logs	16
gce_delete_access_config	17
gce_delete_disk	18
gce_delete_firewall_rule	18
gce_delete_op	19
gce_delete_op.gce_global_operation	20
gce_delete_op.gce_zone_operation	20
gce_extract_projectzone	21
gce_get_disk	21
gce_get_external_ip	22
gce_get_firewall_rule	23
gce_get_global_project	23
gce_get_global_zone	24
gce_get_image	24
gce_get_image_family	25
gce_get_instance	25
gce_get_machinetype	26
gce_get_metadata	27
gce_get_metadata_project	27
gce_get_network	28
gce_get_op	28
gce_get_op.gce_global_operation	29
gce_get_op.gce_zone_operation	29
gce_get_project	30
gce_get_zone	31
gce_global_project	31
gce_global_zone	32
gce_list_disks	32
gce_list_disks_all	33
gce_list_firewall_rules	34
gce_list_gpus	35
gce_list_images	36
gce_list_instances	37
gce_list_machinetype	38
gce_list_machinetype_all	39
gce_list_networks	40

<code>gce_list_zones</code>	41
<code>gce_list_zone_op</code>	41
<code>gce_make_boot_disk</code>	42
<code>gce_make_disk</code>	43
<code>gce_make_firewall_rule</code>	44
<code>gce_make_firewall_webports</code>	46
<code>gce_make_image_source_url</code>	46
<code>gce_make_machinetype_url</code>	47
<code>gce_make_network</code>	48
<code>gce_metadata_env</code>	48
<code>gce_pull_registry</code>	49
<code>gce_push_registry</code>	50
<code>gce_rstudio_adduser</code>	51
<code>gce_rstudio_password</code>	51
<code>gce_schedule_docker</code>	52
<code>gce_set_machinetype</code>	53
<code>gce_set_metadata</code>	54
<code>gce_set_mincupplatform</code>	56
<code>gce_shiny_addapp</code>	56
<code>gce_shiny_listapps</code>	57
<code>gce_shiny_logs</code>	58
<code>gce_ssh</code>	58
<code>gce_ssh_addkeys</code>	61
<code>gce_ssh_browser</code>	62
<code>gce_ssh_setup</code>	63
<code>gce_startup_logs</code>	64
<code>gce_tag_container</code>	65
<code>gce_vm</code>	66
<code>gce_vm_cluster</code>	68
<code>gce_vm_container</code>	69
<code>gce_vm_create</code>	70
<code>gce_vm_delete</code>	73
<code>gce_vm_deletion_protection</code>	73
<code>gce_vm_gpu</code>	74
<code>gce_vm_logs</code>	75
<code>gce_vm_reset</code>	76
<code>gce_vm_scheduler</code>	76
<code>gce_vm_start</code>	77
<code>gce_vm_stop</code>	78
<code>gce_vm_template</code>	79
<code>gce_wait</code>	81
<code>get_dockerfolder</code>	81
<code>googleComputeEngineR</code>	82
<code>localhost</code>	82
<code>makeDockerClusterPSOCK</code>	82

 as.cluster.gce_instance

Create a future cluster for GCE objects

Description

S3 method for `as.cluster()` in the **future** package.

Usage

```
## S3 method for class 'gce_instance'
as.cluster(
  x,
  project = gce_get_global_project(),
  zone = gce_get_global_zone(),
  rshopts = ssh_options(x),
  ...,
  recursive = FALSE
)
```

Arguments

x	The instance to make a future cluster
project	The GCE project
zone	The GCE zone
rshopts	Options for the SSH
...	Other arguments passed to makeDockerClusterPSOCK
recursive	Not used.

Details

Only works for r-base containers created via `gce_vm_template("r-base")` or for docker containers created using the `--net=host` argument flag

Value

A cluster object.

Examples

```
## Not run:
vm <- gce_vm("r-base", name = "future", predefined_type = "f1-micro")
plan(cluster, workers = vm) ## equivalent to workers = as.cluster(vm)
x %<-% { Sys.getinfo() }
print(x)

## End(Not run)
```

containers	<i>Get list of all containers on a host.</i>
------------	--

Description

Get list of all containers on a host.

Usage

```
containers(host = localhost, ...)
```

Arguments

host	A host object.
...	Other arguments passed to the SSH command for the host

Author(s)

Winston Change <winston@stdout.org>

container_logs	<i>Retrieve logs for a container.</i>
----------------	---------------------------------------

Description

Retrieve logs for a container.

Usage

```
container_logs(container, timestamps = FALSE, follow = FALSE)
```

Arguments

container	A container object
timestamps	Show timestamps.
follow	Follow log output as it is happening.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
container_rm(con)  
  
## End(Not run)
```

container_rm *Delete a container.*

Description

Delete a container.

Usage

```
container_rm(container, force = FALSE)
```

Arguments

container	A container object
force	Force removal of a running container.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
container_rm(con)  
  
## End(Not run)
```

container_running *Report whether a container is currently running.*

Description

Report whether a container is currently running.

Usage

```
container_running(container)
```

Arguments

container	A container object
-----------	--------------------

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
container_running(con)  
  
## End(Not run)
```

container_update_info *Update the information about a container.*

Description

This queries docker (on the host) for information about the container, and saves the returned information into a container object, which is returned. This does not use reference semantics, so if you want to store the updated information, you need to save the result.

Usage

```
container_update_info(container)
```

Arguments

container A container object

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
con <- container_update_info(con)  
  
## End(Not run)
```

docker_build *Build image on an instance from a local Dockerfile*

Description

Uploads a folder with a Dockerfile and supporting files to an instance and builds it

Usage

```
docker_build(  
  host = localhost,  
  dockerfolder,  
  new_image,  
  folder = "buildimage",  
  wait = FALSE,  
  ...  
)
```

Arguments

host	A host object.
dockerfolder	Local location of build directory including valid Dockerfile
new_image	Name of the new image
folder	Where on host to build dockerfile
wait	Whether to block R console until finished build
...	Other arguments passed to the SSH command for the host

Details

Dockerfiles are best practice when creating your own docker images, rather than logging into a Docker container, making changes and committing.

Value

A table of active images on the instance

See Also

[Best practices for writing Dockerfiles](#)

An example Dockerfile for [rOpenSci](#)

General R Docker images found at [rocker-org](#)

Examples

```
## Not run:  
docker_build(localhost, "/home/stuff/dockerfolder" , "new_image", wait = TRUE)  
docker_run(localhost, "new_image")  
  
## End(Not run)
```

docker_cmd	<i>Run a docker command on a host.</i>
------------	--

Description

Run a docker command on a host.

Usage

```
docker_cmd(  
  host,  
  cmd = NULL,  
  args = NULL,  
  docker_opts = NULL,  
  capture_text = FALSE,  
  ...  
)
```

Arguments

host	A host object.
cmd	A docker command, such as "run" or "ps"
args	Arguments to pass to the docker command
docker_opts	Options to docker. These are things that come before the docker command, when run on the command line.
capture_text	If FALSE (the default), return the host object. This is useful for chaining functions. If TRUE, capture the text output from both stdout and stderr, and return that. Note that TRUE may not be available on all types of hosts.
...	Other arguments passed to the SSH command for the host

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
docker_cmd(localhost, "ps", "-a")  
  
## End(Not run)
```

`docker_cmd.gce_instance`*Docker S3 method for use with harbor package*

Description

Docker S3 method for use with harbor package

Usage

```
## S3 method for class 'gce_instance'
docker_cmd(
  host,
  cmd = NULL,
  args = NULL,
  docker_opts = NULL,
  capture_text = FALSE,
  nvidia = FALSE,
  ...
)
```

Arguments

<code>host</code>	The GCE instance
<code>cmd</code>	The command to pass to docker
<code>args</code>	arguments to the command
<code>docker_opts</code>	options for docker
<code>capture_text</code>	whether to return the output
<code>nvidia</code>	If true will use nvidia-docker instead of docker
<code>...</code>	other arguments passed to gce_ssh

Details

Instances launched in the `google-containers` image family automatically add your user to the `docker` group, but for others you will need to run `sudo usermod -a -G docker ${USER}` and log out and back in.

docker_inspect	<i>Inspect one or more containers, given name(s) or ID(s).</i>
----------------	--

Description

Inspect one or more containers, given name(s) or ID(s).

Usage

```
docker_inspect(host = localhost, names = NULL, ...)
```

Arguments

host	A host object.
names	Names of the containers
...	Other arguments passed to the SSH command for the host

Value

A list of lists, where each sublist represents one container. This is the output of ‘docker inspect’ translated directly from raw JSON to an R object.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:  
docker_run(localhost, "debian:testing", "echo foo", name = "harbor-test")  
docker_inspect(localhost, "harbor-test")  
  
## End(Not run)
```

docker_pull	<i>Pull a docker image onto a host.</i>
-------------	---

Description

Pull a docker image onto a host.

Usage

```
docker_pull(host = localhost, image, ...)
```

Arguments

host	A host object.
image	The docker image to pull e.g. rocker/rstudio
...	Other arguments passed to the SSH command for the host

Value

The host object.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_pull(localhost, "debian:testing")

## End(Not run)
```

docker_run

Run a command in a new container on a host.

Description

Run a command in a new container on a host.

Usage

```
docker_run(
  host = localhost,
  image = NULL,
  cmd = NULL,
  name = NULL,
  rm = FALSE,
  detach = FALSE,
  docker_opts = NULL,
  ...
)
```

Arguments

host	An object representing the host where the container will be run.
image	The name or ID of a docker image.
cmd	A command to run in the container.
name	A name for the container. If none is provided, a random name will be used.

rm	If TRUE, remove the container after it finishes. This is incompatible with detach=TRUE.
detach	If TRUE, run the container in the background.
docker_opts	Options to docker. These are things that come before the docker command, when run on the command line.
...	Other arguments passed to the SSH command for the host

Value

A container object. When rm=TRUE, this function returns NULL instead of a container object, because the container no longer exists.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_run(localhost, "debian:testing", "echo foo")
#> foo

# Arguments will be concatenated
docker_run(localhost, "debian:testing", c("echo foo", "bar"))
#> foo bar

docker_run(localhost, "rocker/r-base", c("Rscript", "-e", "1+1"))
#> [1] 2

## End(Not run)
```

gce_attach_disk	<i>Attaches a Disk resource to an instance.</i>
-----------------	---

Description

Attaches a Disk resource to an instance.

Usage

```
gce_attach_disk(
  instance,
  source = NULL,
  autoDelete = NULL,
  boot = NULL,
  deviceName = NULL,
  diskEncryptionKey = NULL,
  index = NULL,
  initializeParams = NULL,
```

```

interface = NULL,
licenses = NULL,
mode = NULL,
type = NULL,
project = gce_get_global_project(),
zone = gce_get_global_zone()
)

```

Arguments

instance	The instance name for this request
source	Specifies a valid partial or full URL to an existing Persistent Disk resource
autoDelete	Specifies whether the disk will be auto-deleted when the instance is deleted (but not when the disk is detached from the instance)
boot	Indicates that this is a boot disk
deviceName	Specifies a unique device name of your choice that is reflected into the /dev/disk/by-id/google-* tree of a Linux operating system running within the instance
diskEncryptionKey	Encrypts or decrypts a disk using a customer-supplied encryption key
index	Assigns a zero-based index to this disk, where 0 is reserved for the boot disk
initializeParams	A gce_make_boot_disk object for creating boot disks. Cannot be used with source also defined.
interface	Specifies the disk interface to use for attaching this disk, which is either SCSI or NVME
licenses	[Output Only] Any valid publicly visible licenses
mode	The mode in which to attach this disk, either READ_WRITE or READ_ONLY
type	Specifies the type of the disk, either SCRATCH or PERSISTENT
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

See Also

[Google Documentation](#)

Other AttachedDisk functions: [AttachedDisk\(\)](#)

gce_auth	<i>Defunct - Authenticate this session</i>
----------	--

Description

No longer used. Authenticate via downloading a JSON file and setting in your environment arguments instead.

Usage

```
gce_auth(new_user = FALSE, no_auto = FALSE)
```

Arguments

new_user	If TRUE, reauthenticate via Google login screen
no_auto	Will ignore auto-authentication settings if TRUE

Value

Invisibly, the token that has been saved to the session

gce_check_gpu	<i>Check GPU installed ok</i>
---------------	-------------------------------

Description

Check GPU installed ok

Usage

```
gce_check_gpu(vm)
```

Arguments

vm	The instance to check
----	-----------------------

Value

The NVIDIA-SMI output via ssh

See Also

<https://cloud.google.com/compute/docs/gpus/add-gpus#verify-driver-install>

Other GPU instances: [gce_list_gpus\(\)](#), [gce_vm_gpu\(\)](#)

gce_check_ssh	<i>Calls API for the current SSH settings for an instance</i>
---------------	---

Description

Calls API for the current SSH settings for an instance

Usage

```
gce_check_ssh(instance)
```

Arguments

instance	An instance to check
----------	----------------------

Value

A data.frame of SSH users and public keys

gce_container_logs	<i>Check the docker logs of a container</i>
--------------------	---

Description

Check the docker logs of a container

Usage

```
gce_container_logs(instance, container)
```

```
gce_check_container(...)
```

Arguments

instance	The instance running docker
container	A running container to get logs of
...	Arguments passed to gce_container_logs

Value

logs

`gce_delete_access_config`*Deletes an Access Config, Typically for an External IP Address.*

Description

Deletes an Access Config, Typically for an External IP Address.

Usage

```
gce_delete_access_config(  
    instance,  
    access_config = "external-nat",  
    network_interface = "nic0",  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

<code>instance</code>	Name of the instance resource, or an instance object e.g. from gce_get_instance
<code>access_config</code>	The name of the access config to delete.
<code>network_interface</code>	The name of the network interface.
<code>project</code>	Project ID for this request, default as set by gce_get_global_project
<code>zone</code>	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

Value

A list of operation objects with pending status

See Also

[Google Documentation](#)

gce_delete_disk *Deletes the specified persistent disk.*

Description

Deleting a disk removes its data permanently and is irreversible.

Usage

```
gce_delete_disk(  
    disk,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

disk	Name of the persistent disk to delete
project	Project ID for this request
zone	The name of the zone for this request

Details

However, deleting a disk does not delete any snapshots previously made from the disk. You must separately delete snapshots.

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

See Also

[Google Documentation](#)

gce_delete_firewall_rule
Delete a firewall rule

Description

Deletes a firewall rule of name specified

Usage

```
gce_delete_firewall_rule(name, project = gce_get_global_project())
```

Arguments

<code>name</code>	Name of the firewall rule
<code>project</code>	The Google Cloud project

See Also

API Documentation <https://cloud.google.com/compute/docs/reference/latest/firewalls/delete>

Other firewall functions: [gce_get_firewall_rule\(\)](#), [gce_list_firewall_rules\(\)](#), [gce_make_firewall_rule\(\)](#), [gce_make_firewall_webports\(\)](#)

<code>gce_delete_op</code>	<i>Deletes the specified Operations resource.</i>
----------------------------	---

Description

Deletes the specified Operations resource.

Usage

```
gce_delete_op(operation)
```

Arguments

<code>operation</code>	Name of the Operations resource to delete
------------------------	---

Value

TRUE if successful

See Also

[Google Documentation](#)

`gce_delete_op.gce_global_operation`*Deletes the specified global Operations resource.*

Description

Deletes the specified global Operations resource.

Usage

```
## S3 method for class 'gce_global_operation'  
gce_delete_op(operation)
```

Arguments

operation Name of the Operations resource to delete

Value

The deleted operation

See Also

[Google Documentation](#)

`gce_delete_op.gce_zone_operation`*Deletes the specified zone-specific Operations resource.*

Description

Deletes the specified zone-specific Operations resource.

Usage

```
## S3 method for class 'gce_zone_operation'  
gce_delete_op(operation)
```

Arguments

operation Name of the Operations resource to delete

Value

The deleted operation

See Also

[Google Documentation](#)

gce_extract_projectzone

Extract zone and project from an instance object

Description

Extract zone and project from an instance object

Usage

```
gce_extract_projectzone(instance)
```

Arguments

instance	The instance
----------	--------------

Value

A list of \$project and \$zone

gce_get_disk

Returns a specified persistent disk.

Description

Returns a specified persistent disk.

Usage

```
gce_get_disk(  
    disk,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

disk	Name of the persistent disk to return
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

`gce_get_external_ip` *Get the external IP of an instance*

Description

Get the external IP of an instance

Usage

```
gce_get_external_ip(instance, verbose = TRUE, ...)
```

Arguments

<code>instance</code>	Name or instance object to find the external IP for
<code>verbose</code>	Give a user message about the IP
<code>...</code>	passed to gce_get_instance

This is a helper to extract the external IP of an instance

Value

The external IP

gce_get_firewall_rule *Get a firewall rule*

Description

Get a firewall rule of name specified

Usage

```
gce_get_firewall_rule(name, project = gce_get_global_project())
```

Arguments

name	Name of the firewall rule
project	The Google Cloud project

See Also

API Documentation <https://cloud.google.com/compute/docs/reference/latest/firewalls/get>

Other firewall functions: [gce_delete_firewall_rule\(\)](#), [gce_list_firewall_rules\(\)](#), [gce_make_firewall_rule\(\)](#), [gce_make_firewall_webports\(\)](#)

gce_get_global_project
Get global project name

Description

Project name set this session to use by default

Usage

```
gce_get_global_project()
```

Details

Set the project name via [gce_global_project](#)

Value

Project name

`gce_get_global_zone` *Get global zone name*

Description

zone name set this session to use by default

Usage

```
gce_get_global_zone()
```

Details

Set the zone name via [gce_global_zone](#)

Value

zone name

`gce_get_image` *Returns the specified image.*

Description

Returns the specified image.

Usage

```
gce_get_image(image_project, image)
```

Arguments

`image_project` Project ID of where the image lies
`image` Name of the image resource to return

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

You may want to use [gce_get_image_family](#) instead to ensure the most up to date image is used.

See Also

[Google Documentation](#)

`gce_get_image_family` *Returns the latest image that is part of an image family and is not deprecated.*

Description

Returns the latest image that is part of an image family and is not deprecated.

Usage

```
gce_get_image_family(image_project, family)
```

Arguments

`image_project` Project ID for this request
`family` Name of the image family to search for

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

`gce_get_instance` *Returns the specified Instance resource.*

Description

Returns the specified Instance resource.

Usage

```
gce_get_instance(  
    instance,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

instance	Name of the instance resource
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

`gce_get_machinetype` *Returns the specified machine type.*

Description

Returns the specified machine type.

Usage

```
gce_get_machinetype(  
    machineType,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

machineType	Name of the machine type to return
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_get_metadata *Extract metadata from an instance object*

Description

Extract metadata from an instance object

Usage

```
gce_get_metadata(instance, key = NULL)
```

Arguments

instance	instance to get metadata from
key	optional metadata key to filter metadata result

Value

data.frame \$key and \$value of metadata or NULL

gce_get_metadata_project
Get project wide metadata

Description

Get project wide metadata

Usage

```
gce_get_metadata_project(project = gce_global_project())
```

Arguments

project	The project to get the project-wide metadata from
---------	---

gce_get_network	<i>Returns the specified network.</i>
-----------------	---------------------------------------

Description

Returns the specified network.

Usage

```
gce_get_network(network, project = gce_get_global_project())
```

Arguments

network	Name of the network to return
project	Project ID for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_get_op	<i>Retrieves the specified Operations resource.</i>
------------	---

Description

s3 method dispatcher

Usage

```
gce_get_op(operation = .Last.value)
```

Arguments

operation	Name of the Operations resource to return
-----------	---

Details

S3 Methods for classes

- `gce_get_op.gce_zone_operation`
- `gce_get_op.gce_global_operation`
- `gce_get_op.gce_region_operation`

See Also

[Google Documentation](#)

`gce_get_op.gce_global_operation`

Retrieves the specified global Operations resource.

Description

Retrieves the specified global Operations resource.

Usage

```
## S3 method for class 'gce_global_operation'  
gce_get_op(operation)
```

Arguments

`operation` Name of the Operations resource to return

See Also

[Google Documentation](#)

`gce_get_op.gce_zone_operation`

Retrieves the specified zone-specific Operations resource.

Description

Retrieves the specified zone-specific Operations resource.

Usage

```
## S3 method for class 'gce_zone_operation'  
gce_get_op(operation)
```

Arguments

operation Name of the Operations resource to return

See Also

[Google Documentation](#)

gce_get_project *Returns the specified Project resource.*

Description

Returns the specified Project resource.

Usage

```
gce_get_project(project = gce_get_global_project())
```

Arguments

project Project ID for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_get_zone	Returns the specified Zone resource. Get a list of available zones by making a list() request.
--------------	--

Description

Returns the specified Zone resource. Get a list of available zones by making a list() request.

Usage

```
gce_get_zone(project, zone)
```

Arguments

project	Project ID for this request
zone	Name of the zone resource to return

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_global_project	Set global project name
--------------------	-------------------------

Description

Set a project name used for this R session

Usage

```
gce_global_project(project = gce_get_global_project())
```

Arguments

project	project name you want this session to use by default, or a project object
---------	---

Details

This sets a project to a global environment value so you don't need to supply the project argument to other API calls.

Value

The project name (invisibly)

gce_global_zone	<i>Set global zone name</i>
-----------------	-----------------------------

Description

Set a zone name used for this R session

Usage

```
gce_global_zone(zone)
```

Arguments

zone zone name you want this session to use by default, or a zone object

Details

This sets a zone to a global environment value so you don't need to supply the zone argument to other API calls.

Value

The zone name (invisibly)

gce_list_disks	<i>Retrieves a list of persistent disks contained within the specified zone.</i>
----------------	--

Description

Retrieves a list of persistent disks contained within the specified zone.

Usage

```
gce_list_disks(
  filter = NULL,
  maxResults = NULL,
  pageToken = NULL,
  project = gce_get_global_project(),
  zone = gce_get_global_zone()
)
```


Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

`gce_list_disks_all` *Retrieves an aggregated list of persistent disks across all zones.*

Description

Retrieves an aggregated list of persistent disks across all zones.

Usage

```
gce_list_disks_all(
  filter = NULL,
  maxResults = NULL,
  pageToken = NULL,
  project = gce_get_global_project()
)
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_list_firewall_rules
List firewall rules

Description

Get a firewall rule of name specified

Usage

```
gce_list_firewall_rules(  
    filter = NULL,  
    maxResults = NULL,  
    pageToken = NULL,  
    project = gce_get_global_project()  
)
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	The Google Cloud project

See Also

API Documentation <https://cloud.google.com/compute/docs/reference/latest/firewalls/list>

Other firewall functions: [gce_delete_firewall_rule\(\)](#), [gce_get_firewall_rule\(\)](#), [gce_make_firewall_rule\(\)](#), [gce_make_firewall_webports\(\)](#)

gce_list_gpus	<i>Retrieves a list GPUs you can attach to an instance</i>
---------------	--

Description

Retrieves a list GPUs you can attach to an instance

Usage

```
gce_list_gpus(  
    filter = NULL,  
    maxResults = NULL,  
    pageToken = NULL,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	The name of the zone for this request

Details

To filter you need a single string in the form field_name eq|ne string e.g. gce_list_instances("status eq RUNNING") where eq is 'equals' and ne is 'not-equals'.

See Also

[GPUs on Compute Engine](#)

Other GPU instances: [gce_check_gpu\(\)](#), [gce_vm_gpu\(\)](#)

gce_list_images	<i>Retrieves the list of private images available to the specified project.</i>
-----------------	---

Description

Retrieves the list of private images available to the specified project.

Usage

```
gce_list_images(  
    image_project,  
    filter = NULL,  
    maxResults = NULL,  
    pageToken = NULL  
)
```

Arguments

image_project	Project ID for this request
filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

If you want to get a list of publicly-available images, use this method to make a request to the respective image project, such as debian-cloud, windows-cloud or google-containers.

See Also

[Google Documentation](#)

`gce_list_instances` *Retrieves the list of instances contained within the specified zone.*

Description

Retrieves the list of instances contained within the specified zone.

Usage

```
gce_list_instances(  
  filter = NULL,  
  maxResults = NULL,  
  pageToken = NULL,  
  project = gce_get_global_project(),  
  zone = gce_get_global_zone()  
)
```

Arguments

<code>filter</code>	Sets a filter expression for filtering listed resources, in the form <code>filter=expression</code>
<code>maxResults</code>	The maximum number of results per page that should be returned
<code>pageToken</code>	Specifies a page token to use
<code>project</code>	Project ID for this request
<code>zone</code>	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

To filter you need a single string in the form `field_name eq|ne string` e.g. `gce_list_instances("status eq RUNNING")` where `eq` is 'equals' and `ne` is 'not-equals'.

See Also

[Google Documentation](#)

`gce_list_machinetype` *Retrieves a list of machine types available to the specified project.*

Description

Retrieves a list of machine types available to the specified project.

Usage

```
gce_list_machinetype(  
    filter = NULL,  
    maxResults = NULL,  
    pageToken = NULL,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

<code>filter</code>	Sets a filter expression for filtering listed resources, in the form filter=expression
<code>maxResults</code>	The maximum number of results per page that should be returned
<code>pageToken</code>	Specifies a page token to use
<code>project</code>	Project ID for this request
<code>zone</code>	The name of the zone for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

`gce_list_machinetype_all`*Retrieves an aggregated list of machine types from all zones.*

Description

Retrieves an aggregated list of machine types from all zones.

Usage

```
gce_list_machinetype_all(  
    filter = NULL,  
    maxResults = NULL,  
    pageToken = NULL,  
    project = gce_get_global_project()  
)
```

Arguments

<code>filter</code>	Sets a filter expression for filtering listed resources, in the form filter=expression
<code>maxResults</code>	The maximum number of results per page that should be returned
<code>pageToken</code>	Specifies a page token to use
<code>project</code>	Project ID for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_list_networks	<i>Retrieves the list of networks available to the specified project.</i>
-------------------	---

Description

Retrieves the list of networks available to the specified project.

Usage

```
gce_list_networks(  
  filter = NULL,  
  maxResults = NULL,  
  pageToken = NULL,  
  project = gce_get_global_project()  
)
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_list_zones	<i>Retrieves the list of Zone resources available to the specified project.</i>
----------------	---

Description

Retrieves the list of Zone resources available to the specified project.

Usage

```
gce_list_zones(project, filter = NULL, maxResults = NULL, pageToken = NULL)
```

Arguments

project	Project ID for this request
filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_list_zone_op	<i>Retrieves a list of Operation resources contained within the specified zone.</i>
------------------	---

Description

Retrieves a list of Operation resources contained within the specified zone.

Usage

```
gce_list_zone_op(  
  filter = NULL,  
  maxResults = NULL,  
  pageToken = NULL,  
  project = gce_get_global_project(),  
  zone = gce_get_global_zone()  
)
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	Name of the zone for request

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>
- <https://www.googleapis.com/auth/compute.readonly>

See Also

[Google Documentation](#)

gce_make_boot_disk *Make a boot disk for attachment to an instance*

Description

Make a boot disk for attachment to an instance

Usage

```
gce_make_boot_disk(
  diskName = NULL,
  diskSizeGb = NULL,
  diskType = NULL,
  sourceImage = NULL,
  sourceImageEncryptionKey = NULL
)
```

Arguments

diskName	Specifies the disk name
diskSizeGb	Specifies the size of the disk in base-2 GB
diskType	Specifies the disk type to use to create the instance
sourceImage	The source image used to create this disk
sourceImageEncryptionKey	The customer-supplied encryption key of the source image

Details

Specifies the parameters for a new disk that will be created alongside the new instance.

Use initialization parameters to create boot disks or local SSDs attached to the new instance.

This property is mutually exclusive with the source property; you can only define one or the other, but not both.

Value

AttachedDiskInitializeParams object

gce_make_disk	<i>Creates a persistent disk in the specified project using the data in the request.</i>
---------------	--

Description

You can create a disk with a sourceImage, a sourceSnapshot, or create an empty 500 GB data disk by omitting all properties.

Usage

```
gce_make_disk(
  name,
  sourceImage = NULL,
  sizeGb = NULL,
  description = NULL,
  diskEncryptionKey = NULL,
  licenses = NULL,
  sourceSnapshot = NULL,
  sourceImageEncryptionKey = NULL,
  sourceSnapshotEncryptionKey = NULL,
  type = NULL,
  project = gce_get_global_project(),
  zone = gce_get_global_zone()
)
```

Arguments

name	Name of the resource
sourceImage	The source image used to create this disk
sizeGb	Size of the persistent disk, specified in GB
description	An optional description of this resource
diskEncryptionKey	Encrypts the disk using a customer-supplied encryption key
licenses	Any applicable publicly visible licenses

sourceSnapshot	The source snapshot used to create this disk
sourceImageEncryptionKey	The customer-supplied encryption key of the source image
sourceSnapshotEncryptionKey	The customer-supplied encryption key of the source snapshot
type	URL of the disk type resource describing which disk type to use to create the disk
project	Project ID for this request
zone	The name of the zone for this request

Details

You can also create a disk that is larger than the default size by specifying the sizeGb property. Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

Value

a zone operation

See Also

[Google Documentation](#)

gce_make_firewall_rule

Add one firewall rule to the network

Description

Use this to create firewall rules to apply to the network settings. Most commonly this is to setup web access (port 80 and 443)

Usage

```
gce_make_firewall_rule(
  name,
  protocol,
  ports,
  sourceRanges = NULL,
  sourceTags = NULL,
  project = gce_get_global_project()
)
```

Arguments

name	Name of the firewall rule
protocol	Protocol such as tcp, udp, icmp, esp, ah, sctp or IP protocol number.
ports	Port numbers to open
sourceRanges	From where to accept connections. If NULL then will default to 0.0.0.0/0 (everywhere)
sourceTags	A list of instance tags this rule applies to. One or both of sourceRanges and sourceTags may be set.
project	The Google Cloud project

Value

A global operation object

sourceRanges and/or sourceTags

If both properties are set, an inbound connection is allowed if the range or the tag of the source matches the sourceRanges OR matches the sourceTags property; the connection does not need to match both properties.

See Also

API Documentation <https://cloud.google.com/compute/docs/reference/latest/firewalls/insert>

Other firewall functions: [gce_delete_firewall_rule\(\)](#), [gce_get_firewall_rule\(\)](#), [gce_list_firewall_rules\(\)](#), [gce_make_firewall_webports\(\)](#)

Examples

```
## Not run:  
  
gce_make_firewall_rule("allow-http", protocol = "tcp", ports = 80)  
gce_make_firewall_rule("allow-https", protocol = "tcp", ports = 443)  
gce_make_firewall_rule("shiny", protocol = "tcp", ports = 3838)  
gce_make_firewall_rule("rstudio", protocol = "tcp", ports = 8787)  
  
## End(Not run)
```

gce_make_firewall_webports

Make HTTP and HTTPS firewall rules

Description

Do the common use case of opening HTTP and HTTPS ports

Usage

```
gce_make_firewall_webports(project = gce_get_global_project())
```

Arguments

project The project the firewall will open for

Details

This will invoke [gce_make_firewall_rule](#) and look for the rules named allow-http and allow-https. If not present, it will create them.

Value

Vector of the firewall objects

See Also

Other firewall functions: [gce_delete_firewall_rule\(\)](#), [gce_get_firewall_rule\(\)](#), [gce_list_firewall_rules\(\)](#), [gce_make_firewall_rule\(\)](#)

gce_make_image_source_url

Make initial disk image object

Description

Make initial disk image object

Usage

```
gce_make_image_source_url(image_project, image = NULL, family = NULL)
```

Arguments

image_project Project ID of where the image lies
image Name of the image resource to return
family Name of the image family to search for

Value

The selfLink of the image object

```
gce_make_machinetype_url
```

Construct a machineType URL

Description

Construct a machineType URL

Usage

```
gce_make_machinetype_url(  
  predefined_type = NULL,  
  cpus = NULL,  
  memory = NULL,  
  zone = gce_get_global_zone()  
)
```

Arguments

predefined_type	A predefined machine type from gce_list_machinetype
cpus	If not defining predefined_type, the number of CPUs
memory	If not defining predefined_type, amount of memory
zone	zone for URL

Details

cpus must be in multiples of 2 up to 32 memory must be in multiples of 256

Value

A url for use in instance creation

`gce_make_network` *Make a network interface for instance creation*

Description

Make a network interface for instance creation

Usage

```
gce_make_network(
  network = "default",
  name = NULL,
  subnetwork = NULL,
  externalIP = NULL,
  project = gce_get_global_project()
)
```

Arguments

<code>network</code>	Name of network resource
<code>name</code>	Name of the access config
<code>subnetwork</code>	A subnetwork name if its exists You need to provide accessConfig explicitly if you want an ephemeral IP assigned, see https://cloud.google.com/compute/docs/vm-ip-addresses
<code>externalIP</code>	An external IP you have created previously, leave NULL to have one assigned or "none" for none
<code>project</code>	Project ID for this request

Value

A Network object

`gce_metadata_env` *Turn metadata into an environment argument*

Description

This turns instance metadata into an environment argument R (and other software) can see. Only works on a running instance.

Usage

```
gce_metadata_env(key)
```


Arguments

key The metadata key. Pass "" to list the keys

Value

The metadata key value, if successful

`gce_pull_registry` *Load a previously saved private Google Container*

Description

Load a previously saved private Google Container

Usage

```
gce_pull_registry(
  instance,
  container_name,
  container_url = "gcr.io",
  pull_only = FALSE,
  project = gce_get_global_project(),
  ...
)
```

Arguments

instance The VM to run within

container_name The name of the saved container

container_url The URL of where the container was saved

pull_only If TRUE, will not run the container, only pull to the VM

project Project ID for this request, default as set by [gce_get_global_project](#)

... Other arguments passed to [docker_run](#) or [docker_pull](#)

After starting a VM, you can load the container again using this command.

- For Shiny based containers, pass "-p 80:3838" to run it at the IP URL
- For RStudio based containers, pass "-p 80:8787" to run it at the IP URL

Value

The instance

See Also

Other container registry functions: [gce_push_registry\(\)](#), [gce_tag_container\(\)](#)

gce_push_registry *Push to Google Container Registry*

Description

Commit and save a running container or docker image to the Google Container Registry

Usage

```
gce_push_registry(
    instance,
    save_name,
    container_name = NULL,
    image_name = NULL,
    container_url = "gcr.io",
    project = gce_get_global_project(),
    wait = FALSE
)
```

Arguments

instance	The VM to run within
save_name	The new name for the saved image
container_name	A running docker container. Can't be set if image_name is too.
image_name	A docker image on the instance. Can't be set if container_name is too.
container_url	The URL of where to save container
project	Project ID for this request, default as set by gce_get_global_project This will only work on the Google Container optimised containers of image_family google_containers. Otherwise you will need to get a container authentication yourself (for now) It will start the push but it may take a long time to finish, especially the first time, this function will return whilst waiting but don't turn off the VM until its finished.
wait	Will wait for operation to finish on the instance if TRUE

Value

The tag the image was tagged with on GCE

See Also

Other container registry functions: [gce_pull_registry\(\)](#), [gce_tag_container\(\)](#)

`gce_rstudio_adduser` *Creates a user on an RStudio templated instance*

Description

RStudio has users based on unix user accounts

Usage

```
gce_rstudio_adduser(
  instance,
  username,
  password,
  admin = TRUE,
  container = "rstudio"
)
```

Arguments

<code>instance</code>	An instance with RStudio installed via gce_vm_template
<code>username</code>	The user to create
<code>password</code>	The user password
<code>admin</code>	Default TRUE - Will the user be able to install packages and other sudo tasks?
<code>container</code>	The rstudio container to add the user to

Value

The instance

`gce_rstudio_password` *Changes password for a user on RStudio container*

Description

RStudio has users based on unix user accounts

Usage

```
gce_rstudio_password(instance, username, password, container = "rstudio")
```

Arguments

<code>instance</code>	An instance with RStudio installed via gce_vm_template
<code>username</code>	The user to change the password for
<code>password</code>	The user password
<code>container</code>	The rstudio container to add the user to

Value

The instance

gce_schedule_docker *Schedule running a docker image upon a VM*

Description

Utility function to start a VM to run a docker container on a schedule. You will need to create and build the Dockerfile first.

Usage

```
gce_schedule_docker(
  docker_image,
  schedule = "53 4 * * *",
  vm = gce_vm_scheduler()
)
```

Arguments

docker_image the hosted docker image to run on a schedule
 schedule The schedule you want to run via cron
 vm A VM object to schedule the script upon that you can SSH into

Details

You may need to run [gce_vm_scheduler](#) yourself first and then set up SSH details if not defaults, to pass to argument vm

You can create a Dockerfile with your R script installed by running it through `containerit::dockerfile`. It also takes care of any dependencies.

It is recommended to create a script that is self contained in output and input, e.g. don't save files to the VM, instead upload or download any files from Google Cloud Storage via authentication via `googleAuthR::gar_gce_auth()` then downloading and uploading data using `library(googleCloudStorageR)` or similar.

Once the script is working locally, build it and upload to a repository so it can be reached via argument `docker_image`

You can build via Google cloud repository build triggers, in which case the name can be created via [gce_tag_container](#) or build via [docker_build](#) to build on another VM or locally, then push to a registry via [gce_push_registry](#)

Any Docker image can be run, it does not have to be an R one.

Value

The crontab schedule of the VM including your script

See Also

Other scheduler functions: [gce_vm_scheduler\(\)](#)

Examples

```
## Not run:
# create a Dockerfile of your script
if(!require(containeRit)){
  remotes::install_github("o2r-project/containerit")
  library(containeRit)
}

## create your scheduled script, example below named schedule.R
## it will run the script whilst making the dockerfile
container <- dockerfile("schedule.R",
                        copy = "script_dir",
                        cmd = CMD_Rscript("schedule.R"),
                        soft = TRUE)
write(container, file = "Dockerfile")

## upload created Dockerfile to GitHub,
## then use a Build Trigger to create Docker image "demoDockerScheduler"
## built trigger uses "demo-docker-scheduler" as must be lowercase

## After image is built:
## Create a VM to run the schedule
vm <- gce_vm_scheduler("my_scheduler")

## setup any SSH not on defaults
vm <- gce_vm_setup(vm, username = "mark")

## get the name of the just built Docker image that runs your script
docker_tag <- gce_tag_container("demo-docker-scheduler", project = "gcer-public")

## Schedule the docker_tag to run every day at 0453AM
gce_schedule_docker(docker_tag, schedule = "53 4 * * *", vm = vm)

## End(Not run)
```

gce_set_machinetype	<i>Changes the machine type for a stopped instance to the machine type specified in the request.</i>
---------------------	--

Description

Changes the machine type for a stopped instance to the machine type specified in the request.

Usage

```
gce_set_machinetype(  
    predefined_type,  
    cpus,  
    memory,  
    instance,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

predefined_type	A predefined machine type from gce_list_machinetype
cpus	If not defining predefined_type, the number of CPUs
memory	If not defining predefined_type, amount of memory
instance	Name of the instance resource to change
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

Value

A zone operation job

See Also

[Google Documentation](#)

gce_set_metadata	<i>Sets metadata for the specified instance or projectwise to the data included in the request.</i>
------------------	---

Description

Set, change and append metadata for an instance.

Usage

```
gce_set_metadata(
    metadata,
    instance,
    project = gce_get_global_project(),
    zone = gce_get_global_zone()
)
```

Arguments

metadata	A named list of metadata key/value pairs to assign to this instance
instance	Name of the instance scoping this request. If "project-wide" will set the metadata project wide, available to all instances
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

To append to existing metadata passed a named list.

To change existing metadata pass a named list with the same key and modified value you will change.

To delete metadata pass an empty string "" with the same key

See Also

[Google Documentation](#)

Other Metadata functions: [Metadata\(\)](#)

Examples

```
## Not run:
# Use "project-wide" to set "enable-oslogin" = "TRUE" to take advantage of OS Login.
# But you won't be able to login via SSH if you do
gce_set_metadata(list("enable-oslogin" = "TRUE"), instance = "project-wide")

# enable google logging
gce_set_metadata(list("google-logging-enabled"="True"), instance = "project-wide")

## End(Not run)
```

gce_set_mincpuplatform

Set a minCPU platform on a stopped instance

Description

Set a minCPU platform on a stopped instance

Usage

```
gce_set_mincpuplatform(instance, minCpuPlatform)
```

Arguments

instance	The (stopped) instance to set a minimum CPU platform upon
minCpuPlatform	The platform to set

gce_shiny_addapp

Add Shiny app to a Shiny template instance

Description

Add a local shiny app to a running Shiny VM installed via [gce_vm_template](#) via [docker_build](#) and [gce_push_registry](#) / [gce_pull_registry](#).

Usage

```
gce_shiny_addapp(instance, app_image, dockerfolder = NULL)
```

Arguments

instance	The instance running Shiny
app_image	The name of the Docker image to create or use existing from Google Container Registry. Must be numbers, dashes or lowercase letters only.
dockerfolder	The folder location containing the Dockerfile and app dependencies

Details

To deploy a Shiny app, you first need to construct a Dockerfile which load the R packages and dependencies, as well as copying over the Shiny app in the same folder.

This function will take the Dockerfile, build it into a Docker image and upload it to Google Container Registry for use later.

If already created, then the function will download the app_image from Google Container Registry and start it on the instance provided.

Any existing Shiny Docker containers are stopped and removed, so if you want multiple apps put them in the same Dockerfile.

Value

The instance

Dockerfile

Example Dockerfile's are found in `system.file("dockerfiles", package = "googleComputeEngineR")`

The Dockerfile is in the same folder as your shiny app, which consists of a `ui.R` and `server.R` in a shiny subfolder. This is copied into the Dockerfile in the last line. Change the name of the subfolder to have that name appear in the final URL of the Shinyapp.

This is then run using the R commands below:

See Also

The vignette entry called Shiny App has examples and a walk through.

Examples

```
## Not run:

vm <- gce_vm("shiny-test",
            template = "shiny",
            predefined_type = "n1-standard-1")

vm <- vm_ssh_setup(vm)

app_dir <- system.file("dockerfiles", "shiny-googleAuthRdemo",
                      package = "googleComputeEngineR")

gce_shiny_addapp(vm, app_image = "gceshinydemo", dockerfolder = app_dir)

# a new VM, it loads the Shiny docker image from before
gce_shiny_addapp(vm2, app_image = "gceshinydemo")

## End(Not run)
```

`gce_shiny_listapps` *List shiny apps on the instance*

Description

List shiny apps on the instance

Usage

```
gce_shiny_listapps(instance)
```

Arguments

instance Instance with Shiny apps installed

Value

character vector

gce_shiny_logs *Get the latest shiny logs for a shinyapp*

Description

Get the latest shiny logs for a shinyapp

Usage

```
gce_shiny_logs(instance, shinyapp = NULL)
```

Arguments

instance Instance with Shiny app installed
 shinyapp Name of shinyapp to see logs for. If NULL will return general shiny logs

Value

log printout

gce_ssh *Remotely execute ssh code, upload & download files.*

Description

Assumes that you have ssh & scp installed. If on Windows see website and examples for workarounds.

Usage

```
gce_ssh(  
  instance,  
  ...,  
  key.pub = NULL,  
  key.private = NULL,  
  wait = TRUE,  
  capture_text = "",  
  username = Sys.info()[["user"]]  
)
```

```

gce_ssh_upload(
  instance,
  local,
  remote,
  username = Sys.info()[["user"]],
  key.pub = NULL,
  key.private = NULL,
  verbose = FALSE,
  wait = TRUE
)

gce_ssh_download(
  instance,
  remote,
  local,
  username = Sys.info()[["user"]],
  key.pub = NULL,
  key.private = NULL,
  verbose = FALSE,
  overwrite = FALSE,
  wait = TRUE
)

```

Arguments

instance	Name of the instance of run ssh command upon
...	Shell commands to run. Multiple commands are combined with && so that execution will halt after the first failure.
key.pub	The filepath location of the public key
key.private	The filepath location of the private key
wait	Whether then SSH output should be waited for or run it asynchronously.
capture_text	Possible values are "", to the R console (the default), NULL or FALSE (discard output), TRUE (capture the output in a character vector) or a character string naming a file.
username	The username you used to generate the key-pair
local, remote	Local and remote paths.
verbose	If TRUE, will print command before executing it.
overwrite	If TRUE, will overwrite the local file if exists.

Details

Only works connecting to linux based instances.

On Windows you will need to install an ssh command line client - see examples for an example using RStudio's built in client.

You will need to generate a new SSH key-pair if you have not connected to the instance before via say the gcloud SDK.

To customise SSH connection see [gce_ssh_setup](#)

capture_text is passed to stdout and stderr of [system2](#)

Otherwise, instructions for generating SSH keys can be found here: <https://cloud.google.com/compute/docs/instances/connecting-to-instance>.

Uploads and downloads are recursive, so if you specify a directory, everything inside the directory will also be downloaded.

See Also

<https://cloud.google.com/compute/docs/instances/connecting-to-instance>

Other ssh functions: [gce_ssh_addkeys\(\)](#), [gce_ssh_browser\(\)](#), [gce_ssh_setup\(\)](#)

Examples

Not run:

```
vm <- gce_vm("my-instance")

## if you have already logged in via gcloud, the default keys will be used
## no need to run gce_ssh_addkeys
## run command on instance
gce_ssh(vm, "echo foo")
#> foo

## if running on Windows, use the RStudio default SSH client
## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
## then run:
vm2 <- gce_vm("my-instance2")

## add SSH info to the VM object
## custom info
vm2 <- gce_ssh_setup(vm2,
  username = "mark",
  key.pub = "C://.ssh/id_rsa.pub",
  key.private = "C://.ssh/id_rsa")

## run command on instance
gce_ssh(vm2, "echo foo")
#> foo

## End(Not run)
```

gce_ssh_addkeys	<i>Add SSH details to a gce_instance</i>
-----------------	--

Description

Add SSH details to a `gce_instance`

Usage

```
gce_ssh_addkeys(  
  instance,  
  key.pub = NULL,  
  key.private = NULL,  
  username = Sys.info()[["user"]],  
  overwrite = FALSE  
)
```

Arguments

<code>instance</code>	The <code>gce_instance</code>
<code>key.pub</code>	filepath to public SSH key
<code>key.private</code>	filepath to the private SSK key
<code>username</code>	SSH username to login with
<code>overwrite</code>	Overwrite existing SSH details if they exist

Details

You will only need to run this yourself if you save your SSH keys somewhere other than `$HOME/.ssh/google_compute_engine` or use a different username than your local username as found in `Sys.info[["user"]]`, otherwise it will configure itself automatically the first time you use [gce_ssh](#) in an R session.

If `key.pub` is `NULL` then will look for default Google credentials at `file.path(Sys.getenv("HOME"), ".ssh", "google_compute_engine.pub")`

Value

The instance with SSH details included in `$ssh`

See Also

Other ssh functions: [gce_ssh_browser\(\)](#), [gce_ssh_setup\(\)](#), [gce_ssh\(\)](#)

Examples

```
## Not run:

library(googleComputeEngineR)

vm <- gce_vm("my-instance")

## if you have already logged in via gcloud, the default keys will be used
## no need to run gce_ssh_addkeys
## run command on instance
gce_ssh(vm, "echo foo")

## if running on Windows, use the RStudio default SSH client
## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
## then run:
vm2 <- gce_vm("my-instance2")

## add SSH info to the VM object
## custom info
vm <- gce_ssh_setup(vm,
                    username = "mark",
                    key.pub = "C://.ssh/id_rsa.pub",
                    key.private = "C://.ssh/id_rsa")

## run command on instance
gce_ssh(vm, "echo foo")
#> foo

## example to check logs of rstudio docker container
gce_ssh(vm, "sudo journalctl -u rstudio")

## End(Not run)
```

gce_ssh_browser	<i>Open a cloud SSH browser for an instance</i>
-----------------	---

Description

This will open an SSH from the browser session if `getOption("browser")` is not NULL

Usage

```
gce_ssh_browser(instance)
```

Arguments

instance the instance resource

Details

You will need to login the first time with an email that has access to the instance.

Value

Opens a browser window to the SSH session, returns the SSH URL.

See Also

<https://cloud.google.com/compute/docs/ssh-in-browser>

Other ssh functions: [gce_ssh_addkeys\(\)](#), [gce_ssh_setup\(\)](#), [gce_ssh\(\)](#)

gce_ssh_setup	<i>Setup a SSH connection with GCE from a new SSH key-pair</i>
---------------	--

Description

Uploads ssh-keys to an instance

Usage

```
gce_ssh_setup(
    instance,
    key.pub = NULL,
    key.private = NULL,
    ssh_overwrite = FALSE,
    username = Sys.info()[["user"]]
)
```

Arguments

instance	Name of the instance of run ssh command upon
key.pub	The filepath location of the public key
key.private	The filepath location of the private key
ssh_overwrite	Will check if SSH settings already set and overwrite them if TRUE
username	The username you used to generate the key-pair

Details

This loads a public ssh-key to an instance's metadata. It does not use the project SSH-Keys, that may be set separately.

You will need to generate a new SSH key-pair if you have not connected to an instance before.

Instructions for this can be found here: <https://cloud.google.com/compute/docs/instances/connecting-to-instance>. Once you have generated run this function once to initiate setup.

If you have historically connected via gcloud or some other means, ssh keys may have been generated automatically.

These will be looked for and used if found, at `file.path(Sys.getenv("HOME"), ".ssh", "google_compute_engine.pub`

Value

TRUE if successful

See Also

<https://cloud.google.com/compute/docs/instances/adding-removing-ssh-keys>

Other ssh functions: [gce_ssh_addkeys\(\)](#), [gce_ssh_browser\(\)](#), [gce_ssh\(\)](#)

Examples

```
## Not run:

library(googleComputeEngineR)

vm <- gce_vm("my-instance")

## if you have already logged in via gcloud, the default keys will be used
## no need to run gce_ssh_addkeys
## run command on instance
gce_ssh(vm, "echo foo")

## if running on Windows, use the RStudio default SSH client
## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
## then run:
vm2 <- gce_vm("my-instance2")

## add SSH info to the VM object
## custom info
vm <- gce_ssh_setup(vm,
  username = "mark",
  key.pub = "C://.ssh/id_rsa.pub",
  key.private = "C://.ssh/id_rsa")

## run command on instance
gce_ssh(vm, "echo foo")
#> foo

## example to check logs of rstudio docker container
gce_ssh(vm, "sudo journalctl -u rstudio")

## End(Not run)
```

gce_startup_logs

Get startup script logs

Description

Get startup script logs

Usage

```
gce_startup_logs(instance, type = c("shell", "cloud-config", "nginx"))
```

Arguments

instance	The instance to get startup script logs from
type	The type of log to run Will use SSH so that needs to be setup

gce_tag_container	<i>Return a container tag for Google Container Registry</i>
-------------------	---

Description

Return a container tag for Google Container Registry

Usage

```
gce_tag_container(
  container_name,
  project = gce_get_global_project(),
  container_url = "gcr.io"
)
```

Arguments

container_name	A running docker container. Can't be set if image_name is too.
project	Project ID for this request, default as set by gce_get_global_project This will only work on the Google Container optimised containers of image_family google_containers. Otherwise you will need to get a container authentication yourself (for now) It will start the push but it may take a long time to finish, especially the first time, this function will return whilst waiting but don't turn off the VM until its finished.
container_url	The URL of where to save container

Value

A tag for use in Google Container Registry

See Also

Other container registry functions: [gce_pull_registry\(\)](#), [gce_push_registry\(\)](#)

gce_vm

*Create or fetch a virtual machine***Description**

Pass in the instance name to fetch its object, or create the instance via [gce_vm_create](#).

Usage

```
gce_vm(
  name,
  ...,
  project = gce_get_global_project(),
  zone = gce_get_global_zone(),
  open_webports = TRUE
)
```

Arguments

name	The name of the instance
...	Arguments passed on to gce_vm_create
image_project	Project ID of where the image lies
image	Name of the image resource to return
image_family	Name of the image family to search for
disk_source	Specifies a valid URL to an existing Persistent Disk resource.
network	A network object created by gce_make_network
externalIP	An external IP you have previously reserved, leave NULL to have one assigned or "none" for no external access.
minCpuPlatform	Specify a minimum CPU platform as per https://cloud.google.com/compute/docs/instance-templates#min-cpu-platform
dry_run	whether to just create the request JSON
disk_size_gb	If not NULL, override default size of the boot disk (size in GB)
use_beta	If set to TRUE will use the beta version of the API. Should not be used for production purposes.
acceleratorCount	Number of GPUs to add to instance. If using this, you may want to instead use gce_vm_gpu which sets some defaults for GPU instances.
acceleratorType	Name of GPU to add, see gce_list_gpus
canIpForward	Allows this instance to send and receive packets with non-matching destination or source IPs
description	An optional description of this resource
metadata	A named list of metadata key/value pairs assigned to this instance
scheduling	Scheduling options for this instance, such as preemptible instances

	serviceAccounts	A list of service accounts, with their specified scopes, authorized for this instance
	tags	A list of tags to apply to this instance
	predefined_type	A predefined machine type from gce_list_machinetype
	cpus	If not defining predefined_type, the number of CPUs
	memory	If not defining predefined_type, amount of memory
project		Project ID for this request
zone		The name of the zone for this request
open_webports		If TRUE, will open firewall ports 80 and 443 if not open already

Details

Will get or create the instance as specified. Will wait for instance to be created if necessary.

Make sure the instance is big enough to handle what you need, for instance the default f1-micro will hang the instance when trying to install large R libraries.

Value

A gce_instance object

Creation logic

You need these parameters defined to call the right function for creation. Check the function definitions for more details.

If the VM name exists but is not running, it start the VM and return the VM object

If the VM is running, it will return the VM object

If you specify the argument template it will call [gce_vm_template](#)

If you specify one of file or cloud_init it will call [gce_vm_container](#)

Otherwise it will call [gce_vm_create](#)

Examples

```
## Not run:

library(googleComputeEngineR)
## auto auth, project and zone pre-set
## list your VMs in the project/zone

the_list <- gce_list_instances()

## start an existing instance
vm <- gce_vm("markdev")

## for rstudio, you also need to specify a username and password to login
vm <- gce_vm(template = "rstudio",
             name = "rstudio-server",
             username = "mark", password = "mark1234")
```

```

## specify your own cloud-init file and pass it into gce_vm_container()
vm <- gce_vm(cloud_init = "example.yml",
             name = "test-container",
             predefined_type = "f1-micro")

## specify disk size at creation
vm <- gce_vm('my-image3', disk_size_gb = 20)

## End(Not run)

```

gce_vm_cluster

Make a VM cluster suitable for running parallel workloads

Description

This wraps the commands for creating a cluster suitable for [future](#) workloads.

Usage

```

gce_vm_cluster(
  vm_prefix = "r-cluster-",
  cluster_size = 3,
  docker_image = NULL,
  ...,
  ssh_args = NULL,
  project = gce_get_global_project(),
  zone = gce_get_global_zone()
)

```

Arguments

vm_prefix	The prefix of the VMs you want to make. Will be appended the cluster number
cluster_size	The number of VMs in your cluster
docker_image	The docker image the jobs on the cluster will run on. Default NULL will use rocker/r-parallel
...	Passed to gce_vm_template
ssh_args	A list of optional arguments that will be passed to gce_ssh_setup
project	The project to launch the cluster in
zone	The zone to launch the cluster in

Examples

```
## Not run:
library(future)
library(googleComputeEngineR)

vms <- gce_vm_cluster()

## make a future cluster
plan(cluster, workers = as.cluster(vms))

## End(Not run)
```

gce_vm_container	<i>Launch a container-VM image</i>
------------------	------------------------------------

Description

This lets you specify docker images when creating the VM. These are a special class of Google instances that are setup for running Docker containers.

Usage

```
gce_vm_container(
  file = NULL,
  cloud_init = NULL,
  shell_script = NULL,
  image_family = "cos-stable",
  image_project = "cos-cloud",
  ...
)
```

Arguments

file	file location of a valid cloud-init or shell_script file. One of file or cloud_init or shell_script must be supplied
cloud_init	contents of a cloud-init file, for example read via readChar(file, nchars = 32768)
shell_script	contents of a shell_script file, for example read via readChar(file, nchars = 32768)
image_family	An image-family. It must come from the image_project family.
image_project	An image-project, where the image-family resides.
...	Other arguments passed to gce_vm_create

Details

file expects a filepath to a <https://cloudinit.readthedocs.io/en/latest/topics/format.html> configuration file or a valid bash script e.g. has `#!/bin/` or `#cloud-config` at top of file.

image_project will be ignored if set, overridden to cos-cloud. If you want to set it then use the [gce_vm_create](#) function directly that this function wraps with some defaults.

Value

A zone operation

See Also

<https://cloud.google.com/container-optimized-os/docs/how-to/create-configure-instance-help-using-cloud-init-files>

gce_vm_create	<i>Creates an instance resource in the specified project using the data included in the request.</i>
---------------	--

Description

Creates an instance resource in the specified project using the data included in the request.

Usage

```
gce_vm_create(
  name,
  predefined_type = "f1-micro",
  image_project = "debian-cloud",
  image_family = "debian-9",
  cpus = NULL,
  memory = NULL,
  image = "",
  disk_source = NULL,
  network = gce_make_network("default", project = project),
  externalIP = NULL,
  canIpForward = NULL,
  description = NULL,
  metadata = NULL,
  scheduling = NULL,
  serviceAccounts = NULL,
  tags = NULL,
  minCpuPlatform = NULL,
  project = gce_get_global_project(),
  zone = gce_get_global_zone(),
  dry_run = FALSE,
```

```

    disk_size_gb = NULL,
    use_beta = FALSE,
    acceleratorCount = NULL,
    acceleratorType = "nvidia-tesla-p4"
)

```

Arguments

name	The name of the resource, provided by the client when initially creating the resource
predefined_type	A predefined machine type from gce_list_machinetype
image_project	Project ID of where the image lies
image_family	Name of the image family to search for
cpus	If not defining predefined_type, the number of CPUs
memory	If not defining predefined_type, amount of memory
image	Name of the image resource to return
disk_source	Specifies a valid URL to an existing Persistent Disk resource.
network	A network object created by gce_make_network
externalIP	An external IP you have previously reserved, leave NULL to have one assigned or "none" for no external access.
canIpForward	Allows this instance to send and receive packets with non-matching destination or source IPs
description	An optional description of this resource
metadata	A named list of metadata key/value pairs assigned to this instance
scheduling	Scheduling options for this instance, such as preemptible instances
serviceAccounts	A list of service accounts, with their specified scopes, authorized for this instance
tags	A list of tags to apply to this instance
minCpuPlatform	Specify a minimum CPU platform as per https://cloud.google.com/compute/docs/instances/specify-min-cpu-platform
project	Project ID for this request
zone	The name of the zone for this request
dry_run	whether to just create the request JSON
disk_size_gb	If not NULL, override default size of the boot disk (size in GB)
use_beta	If set to TRUE will use the beta version of the API. Should not be used for production purposes.
acceleratorCount	Number of GPUs to add to instance. If using this, you may want to instead use gce_vm_gpu which sets some defaults for GPU instances.
acceleratorType	Name of GPU to add, see gce_list_gpus

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

cpus must be in multiples of 2 up to 32 memory must be in multiples of 256

One of image or image_family must be supplied

To create an instance you need to specify:

- Name
- Project [if not default]
- Zone [if not default]
- Machine type - either a predefined type or custom CPU and memory
- Network - usually default, specifies open ports etc.
- Image - a source image containing the operating system

You can add metadata to the server such as startup-script and shutdown-script. Details available here: <https://cloud.google.com/compute/docs/storing-retrieving-metadata>

If you want to not have an external IP then modify the instance afterwards

Value

A zone operation, or if the name already exists the VM object from [gce_get_instance](#)

Preemptible VMS

You can set **preemptible** VMs by passing this in the scheduling arguments `scheduling = list(preemptible = TRUE)`

This creates a VM that may be shut down prematurely by Google - you will need to sort out how to save state if that happens in a shutdown script etc. However, these are much cheaper.

GPUs

Some defaults for launching GPU enabled VMs are available at [gce_vm_gpu](#)

You can add GPUs to your instance, but they must be present in the zone you have specified - use [gce_list_gpus](#) to see which are available. Refer to [this](#) link for a list of current GPUs per zone.

See Also

[Google Documentation](#)

gce_vm_delete	<i>Deletes the specified Instance resource.</i>
---------------	---

Description

Deletes the specified Instance resource.

Usage

```
gce_vm_delete(  
    instances,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

instances	Name of the instance resource, or an instance object e.g. from gce_get_instance , or a list of instances to delete
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

See Also

[Google Documentation](#)

gce_vm_deletion_protection	<i>Toggle deletion protection for existing instances</i>
----------------------------	--

Description

Toggle deletion protection for existing instances

Usage

```
gce_vm_deletion_protection(
  instance,
  cmd = c("status", "true", "false"),
  project = gce_get_global_project(),
  zone = gce_get_global_zone()
)
```

Arguments

instance	The vm to work with its deletion protection
cmd	Whether to get the status, or toggle "true" or "false" on deletion protection for this VM
project	The projectId
zone	The zone

Examples

```
## Not run:

# a workflow for deleting lots of VMs across zones that have deletion protection
zones <- gce_list_zones()
instances <- lapply(zones$name, function(x) gce_list_instances(zone = x))

instances_e <- lapply(instances, function(x) x$items$name)
names(instances_e) <- zones$name

status <- lapply(zones$name, function(x){
  lapply(instances_e[[x]], function(y) {
    gce_vm_deletion_protection(y, cmd = "false", zone = x))
  })
})

deletes <- lapply(zones$name, function(x){
  lapply(instances_e[[x]], function(y) {
    gce_vm_delete(y, zone = x))
  })
})

## End(Not run)
```

gce_vm_gpu

Launch a GPU enabled instance

Description

Helper function that fills in some defaults passed to [gce_vm](#)

Usage

```
gce_vm_gpu(..., return_dots = FALSE)
```

Arguments

... arguments passed to [gce_vm](#)
 return_dots Only return the settings, do not call [gce_vm](#)

Details

If not specified, this function will enter defaults to get a GPU instance up and running.

- acceleratorCount: 1
- acceleratorType: "nvidia-tesla-p4"
- scheduling: list(onHostMaintenance = "TERMINATE", automaticRestart = TRUE)
- image_project: "deeplearning-platform-release"
- image_family: "tf2-ent-latest-gpu"
- predefined_type: "n1-standard-8"
- metadata: "install-nvidia-driver" = "True"

Value

A VM object

See Also

<https://cloud.google.com/deep-learning-vm/docs/quickstart-cli>
 Other GPU instances: [gce_check_gpu\(\)](#), [gce_list_gpus\(\)](#)

gce_vm_logs

Open browser to the serial console output for a VM

Description

Saves a few clicks

Usage

```
gce_vm_logs(instance, open_browser = TRUE)
```

Arguments

instance The VM to see serial console output for
 open_browser Whether to return a URL or open the browser

Value

a URL

gce_vm_reset *Performs a hard reset on the instance.*

Description

Performs a hard reset on the instance.

Usage

```
gce_vm_reset(  
    instances,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

instances	Name of the instance resource, or an instance object e.g. from gce_get_instance
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

See Also

[Google Documentation](#)

gce_vm_scheduler *Create or start a scheduler VM*

Description

This starts up a VM with cron and docker installed that can be used to schedule scripts

Usage

```
gce_vm_scheduler(vm_name = "scheduler", ...)
```

Arguments

vm_name	The name of the VM scheduler to create or return
...	Arguments passed on to gce_vm
name	The name of the instance
open_webports	If TRUE, will open firewall ports 80 and 443 if not open already
project	Project ID for this request
zone	The name of the zone for this request

Value

A VM object

See Also

Other scheduler functions: [gce_schedule_docker\(\)](#)

gce_vm_start	<i>Starts an instance that was stopped using the using the stop method.</i>
--------------	---

Description

Starts an instance that was stopped using the using the stop method.

Usage

```
gce_vm_start(
  instances,
  project = gce_get_global_project(),
  zone = gce_get_global_zone()
)
```

Arguments

instances	Name of the instance resource, or an instance object e.g. from gce_get_instance
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

Value

A list of operation objects with pending status

See Also

[Google Documentation](#)

gce_vm_stop	<i>Stops a running instance, shutting it down cleanly, and allows you to restart the instance at a later time.</i>
-------------	--

Description

Stops a running instance, shutting it down cleanly, and allows you to restart the instance at a later time.

Usage

```
gce_vm_stop(  
    instances,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

```
gce_vm_suspend(  
    instances,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

```
gce_vm_resume(  
    instances,  
    project = gce_get_global_project(),  
    zone = gce_get_global_zone()  
)
```

Arguments

instances	Names of the instance resource, or an instance object e.g. from gce_get_instance
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- <https://www.googleapis.com/auth/cloud-platform>
- <https://www.googleapis.com/auth/compute>

Stopped instances do not incur per-minute, virtual machine usage charges while they are stopped, but any resources that the virtual machine is using, such as persistent disks and static IP addresses, will continue to be charged until they are deleted.

See Also

[Google Documentation](#)

gce_vm_template	<i>Create a template container VM</i>
-----------------	---------------------------------------

Description

This lets you specify templates for the VM you want to launch. It passes the template on to [gce_vm_container](#).

Usage

```
gce_vm_template(
  template = c("rstudio", "shiny", "opencpu", "r-base", "r-parallel", "dynamic",
    "rstudio-gpu", "rstudio-shiny", "rstudio-noauth"),
  username = NULL,
  password = NULL,
  dynamic_image = NULL,
  image_family = "cos-stable",
  wait = TRUE,
  ...
)
```

Arguments

template	The template available
username	username if needed (RStudio)
password	password if needed (RStudio)
dynamic_image	Supply an alternative to the default Docker image for the template
image_family	An image-family. It must come from the cos-cloud family.
wait	Whether to wait for the VM to launch before returning. Default TRUE.
...	Arguments passed on to gce_vm_container
	file file location of a valid cloud-init or shell_script file. One of file or cloud_init or shell_script must be supplied

cloud_init contents of a cloud-init file, for example read via readChar(file, nchars = 32768)
 shell_script contents of a shell_script file, for example read via readChar(file, nchars = 32768)
 image_project An image-project, where the image-family resides.

Details

Templates available are:

- rstudio An RStudio server docker image with tidyverse and devtools
- rstudio-gpu An RStudio server with popular R machine learning libraries and GPU driver. Will launch a GPU enabled VM.
- rstudio-shiny An RStudio server with Shiny also installed, proxied to /shiny
- shiny A Shiny docker image
- opencpu An OpenCPU docker image
- r-base Latest version of R stable
- r-parallel Image with future enabled for parallel workloads
- dynamic Supply your own docker image within dynamic_image

For dynamic templates you will need to launch the docker image with any ports you want opened, other settings etc. via [docker_run](#).

Use dynamic_image to override the default rocker images e.g. rocker/shiny for shiny, etc.

Value

The VM object, or the VM startup operation if wait=FALSE

Examples

```
## Not run:

library(googleComputeEngineR)

## make instance using R-base
vm <- gce_vm_template("r-base", predefined_type = "f1-micro", name = "rbase")

## run an R function on the instance within the R-base docker image
docker_run(vm, "rocker/r-base", c("Rscript", "-e", "1+1"), user = "mark")
#> [1] 2

## End(Not run)
```

gce_wait	<i>Wait for an operation to finish</i>
----------	--

Description

Will periodically check an operation until its status is DONE

Usage

```
gce_wait(operation, wait = 3, verbose = TRUE, timeout_tries = 50)
```

Arguments

operation	The operation object
wait	Time in seconds between checks, default 3 seconds.
verbose	Whether to give user feedback
timeout_tries	Number of times to wait

Value

The completed job object, invisibly

get_dockerfolder	<i>Get Dockerfolder of templates</i>
------------------	--------------------------------------

Description

This gets the folder location of available Dockerfile examples

Usage

```
get_dockerfolder(dockerfile_folder)
```

Arguments

dockerfile_folder	The folder containing Dockerfile
-------------------	----------------------------------

Value

file location

googleComputeEngineR *Working with Google Compute Engine from R*

Description

See demos and examples at the <https://cloudyr.github.io/googleComputeEngineR/>.

localhost *An object representing the current computer that R is running on.*

Description

An object representing the current computer that R is running on.

Usage

```
localhost
```

Format

An object of class localhost (inherits from host) of length 0.

makeDockerClusterPSOCK
Make the Docker cluster on Google Compute Engine

Description

Called by [as.cluster](#)

Usage

```
makeDockerClusterPSOCK(  
  workers,  
  docker_image = "rocker/r-parallel",  
  rscript = c("docker", "run", "--net=host", docker_image, "Rscript"),  
  rscript_args = NULL,  
  install_future = FALSE,  
  ...,  
  verbose = FALSE  
)
```

Arguments

workers	The VMs being called upon
docker_image	The docker image to use on the cluster
rscript	The Rscript command to run on the cluster
rscript_args	Arguments to the RScript
install_future	Whether to check if future is installed first (not needed if using docker derived from rocker/r-parallel which is recommended)
...	Other arguments passed to makeClusterPSOCK
verbose	How much feedback to show

Author(s)

Henrik Bengtsson <henrikb@braju.com>

Index

- * **AttachedDisk functions**
 - [gce_attach_disk](#), 13
- * **AttachedDiskInitializeParams functions**
 - [gce_make_boot_disk](#), 42
- * **GPU instances**
 - [gce_check_gpu](#), 15
 - [gce_list_gpus](#), 35
 - [gce_vm_gpu](#), 74
- * **Metadata functions**
 - [gce_set_metadata](#), 54
- * **container registry functions**
 - [gce_pull_registry](#), 49
 - [gce_push_registry](#), 50
 - [gce_tag_container](#), 65
- * **datasets**
 - [localhost](#), 82
- * **firewall functions**
 - [gce_delete_firewall_rule](#), 18
 - [gce_get_firewall_rule](#), 23
 - [gce_list_firewall_rules](#), 34
 - [gce_make_firewall_rule](#), 44
 - [gce_make_firewall_webports](#), 46
- * **project functions**
 - [gce_get_global_project](#), 23
- * **scheduler functions**
 - [gce_schedule_docker](#), 52
 - [gce_vm_scheduler](#), 76
- * **ssh functions**
 - [gce_ssh](#), 58
 - [gce_ssh_addkeys](#), 61
 - [gce_ssh_browser](#), 62
 - [gce_ssh_setup](#), 63
- * **zone functions**
 - [gce_get_global_zone](#), 24
- [as.cluster](#), 4, 82
- [as.cluster.gce_instance](#), 4
- [AttachedDisk](#), 14
- [container_logs](#), 5
- [container_rm](#), 6
- [container_running](#), 6
- [container_update_info](#), 7
- [containers](#), 5
- [docker_build](#), 7, 52, 56
- [docker_cmd](#), 9
- [docker_cmd.gce_instance](#), 10
- [docker_inspect](#), 11
- [docker_pull](#), 11, 49
- [docker_run](#), 12, 49, 80
- [future](#), 68
- [gce_attach_disk](#), 13
- [gce_auth](#), 15
- [gce_check_container](#)
 - [\(gce_container_logs\)](#), 16
- [gce_check_gpu](#), 15, 35, 75
- [gce_check_ssh](#), 16
- [gce_container_logs](#), 16, 16
- [gce_delete_access_config](#), 17
- [gce_delete_disk](#), 18
- [gce_delete_firewall_rule](#), 18, 23, 34, 45, 46
- [gce_delete_op](#), 19
- [gce_delete_op.gce_global_operation](#), 20
- [gce_delete_op.gce_zone_operation](#), 20
- [gce_extract_projectzone](#), 21
- [gce_get_disk](#), 21
- [gce_get_external_ip](#), 22
- [gce_get_firewall_rule](#), 19, 23, 34, 45, 46
- [gce_get_global_project](#), 17, 23, 26, 49, 50, 54, 55, 65, 73, 76–78
- [gce_get_global_zone](#), 17, 24, 26, 54, 55, 73, 76–78
- [gce_get_image](#), 24
- [gce_get_image_family](#), 24, 25
- [gce_get_instance](#), 17, 22, 25, 72, 73, 76–78
- [gce_get_machinetype](#), 26

`gce_get_metadata`, 27
`gce_get_metadata_project`, 27
`gce_get_network`, 28
`gce_get_op`, 28
`gce_get_op.gce_global_operation`, 29
`gce_get_op.gce_zone_operation`, 29
`gce_get_project`, 30
`gce_get_zone`, 31
`gce_global_project`, 23, 31
`gce_global_zone`, 24, 32
`gce_list_disks`, 32
`gce_list_disks_all`, 33
`gce_list_firewall_rules`, 19, 23, 34, 45, 46
`gce_list_gpus`, 15, 35, 66, 71, 72, 75
`gce_list_images`, 36
`gce_list_instances`, 37
`gce_list_machinetype`, 38, 47, 54, 67, 71
`gce_list_machinetype_all`, 39
`gce_list_networks`, 40
`gce_list_zone_op`, 41
`gce_list_zones`, 41
`gce_make_boot_disk`, 14, 42
`gce_make_disk`, 43
`gce_make_firewall_rule`, 19, 23, 34, 44, 46
`gce_make_firewall_webports`, 19, 23, 34, 45, 46
`gce_make_image_source_url`, 46
`gce_make_machinetype_url`, 47
`gce_make_network`, 48, 66, 71
`gce_metadata_env`, 48
`gce_pull_registry`, 49, 50, 56, 65
`gce_push_registry`, 49, 50, 52, 56, 65
`gce_rstudio_adduser`, 51
`gce_rstudio_password`, 51
`gce_schedule_docker`, 52, 77
`gce_set_machinetype`, 53
`gce_set_metadata`, 54
`gce_set_mincpuplatform`, 56
`gce_shiny_addapp`, 56
`gce_shiny_listapps`, 57
`gce_shiny_logs`, 58
`gce_ssh`, 10, 58, 61, 63, 64
`gce_ssh_addkeys`, 60, 61, 63, 64
`gce_ssh_browser`, 60, 61, 62, 64
`gce_ssh_download` (`gce_ssh`), 58
`gce_ssh_setup`, 60, 61, 63, 63, 68
`gce_ssh_upload` (`gce_ssh`), 58
`gce_startup_logs`, 64
`gce_tag_container`, 49, 50, 52, 65
`gce_vm`, 66, 74, 75, 77
`gce_vm_cluster`, 68
`gce_vm_container`, 67, 69, 79
`gce_vm_create`, 66, 67, 69, 70, 70
`gce_vm_delete`, 73
`gce_vm_deletion_protection`, 73
`gce_vm_gpu`, 15, 35, 66, 71, 72, 74
`gce_vm_logs`, 75
`gce_vm_reset`, 76
`gce_vm_resume` (`gce_vm_stop`), 78
`gce_vm_scheduler`, 52, 53, 76
`gce_vm_start`, 77
`gce_vm_stop`, 78
`gce_vm_suspend` (`gce_vm_stop`), 78
`gce_vm_template`, 51, 56, 67, 68, 79
`gce_wait`, 81
`get_dockerfolder`, 81
`googleComputeEngineR`, 82

`localhost`, 82

`makeClusterPSOCK`, 83
`makeDockerClusterPSOCK`, 82
`Metadata`, 55

`system2`, 60