» docker_registry_image

Note: The initial (current) version of this data source can reliably read only **public** images **from the official Docker Hub Registry**.

Reads the image metadata from a Docker Registry. Used in conjunction with the docker_image resource to keep an image up to date on the latest available version of the tag.

» Example Usage

```
data "docker_registry_image" "ubuntu" {
  name = "ubuntu:precise"
}

resource "docker_image" "ubuntu" {
  name          = "${data.docker_registry_image.ubuntu.name}"
  pull_triggers = ["${data.docker_registry_image.ubuntu.sha256_digest}"]
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required, string) The name of the Docker image, including any tags. e.g. `alpine:latest`

» Attributes Reference

The following attributes are exported in addition to the above configuration:

- `sha256_digest` (string) - The content digest of the image, as stored on the registry.

» docker_container

Manages the lifecycle of a Docker container.

» Example Usage

```
# Start a container
resource "docker_container" "ubuntu" {
  name  = "foo"
  image = "${docker_image.ubuntu.latest}"
}

# Find the latest Ubuntu precise image.
resource "docker_image" "ubuntu" {
  name = "ubuntu:precise"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required, string) The name of the Docker container.
- `image` - (Required, string) The ID of the image to back this container. The easiest way to get this value is to use the `docker_image` resource as is shown in the example above.
- `command` - (Optional, list of strings) The command to use to start the container. For example, to run `/usr/bin/myprogram -f baz.conf` set the command to be `["/usr/bin/myprogram", "-f", "baz.conf"]`.
- `entrypoint` - (Optional, list of strings) The command to use as the Entrypoint for the container. The Entrypoint allows you to configure a container to run as an executable. For example, to run `/usr/bin/myprogram` when starting a container, set the entrypoint to be `["/usr/bin/myprogram"]`.
- `user` - (Optional, string) User used for run the first process. Format is `user` or `user:group` which user and group can be passed literraly or by name.
- `dns` - (Optional, set of strings) Set of DNS servers.
- `dns_opts` - (Optional, set of strings) Set of DNS options used by the DNS provider(s), see `resolv.conf` documentation for valid list of options.
- `dns_search` - (Optional, set of strings) Set of DNS search domains that are used when bare unqualified hostnames are used inside of the container.
- `env` - (Optional, set of strings) Environment variables to set.
- `labels` - (Optional, map of strings) Key/value pairs to set as labels on the container.
- `links` - (Optional, set of strings) Set of links for link based connectivity between containers that are running on the same host.
- `hostname` - (Optional, string) Hostname of the container.
- `domainname` - (Optional, string) Domain name of the container.

- `restart` - (Optional, string) The restart policy for the container. Must be one of ”no”, ”on-failure”, ”always”, ”unless-stopped”.
- `max_retry_count` - (Optional, int) The maximum amount of times to an attempt a restart when `restart` is set to ”on-failure”
- `must_run` - (Optional, bool) If true, then the Docker container will be kept running. If false, then as long as the container exists, Terraform assumes it is successful.
- `capabilities` - (Optional, block) See Capabilities below for details.
- `ports` - (Optional, block) See Ports below for details.
- `host` - (Optional, block) See Extra Hosts below for details.
- `privileged` - (Optional, bool) Run container in privileged mode.
- `publish_all_ports` - (Optional, bool) Publish all ports of the container.
- `volumes` - (Optional, block) See Volumes below for details.
- `memory` - (Optional, int) The memory limit for the container in MBs.
- `memory_swap` - (Optional, int) The total memory limit (memory + swap) for the container in MBs. This setting may compute to `-1` after `terraform apply` if the target host doesn’t support memory swap, when that is the case docker will use a soft limitation.
- `cpu_shares` - (Optional, int) CPU shares (relative weight) for the container.
- `log_driver` - (Optional, string) The logging driver to use for the container. Defaults to ”json-file”.
- `log_opts` - (Optional, map of strings) Key/value pairs to use as options for the logging driver.
- `network_alias` - (Optional, set of strings) Network aliases of the container for user-defined networks only.
- `network_mode` - (Optional, string) Network mode of the container.
- `networks` - (Optional, set of strings) Id of the networks in which the container is.
- `destroy_grace_seconds` - (Optional, int) If defined will attempt to stop the container before destroying. Container will be destroyed after `n` seconds or on successful stop.
- `upload` - (Optional, block) See File Upload below for details.

» Capabilities

`capabilities` is a block within the configuration that allows you to add or drop linux capabilities. For more information about what capabilities you can add and drop please visit the docker run documentation.

- `add` - (Optional, set of strings) list of linux capabilities to add.
- `drop` - (Optional, set of strings) list of linux capabilities to drop.

Example:

```
resource "docker_container" "ubuntu" {
  name = "foo"
  image = "${docker_image.ubuntu.latest}"
  capabilities {
    add = ["ALL"]
    drop = ["SYS_ADMIN"]
  }
}
```

» Ports

`ports` is a block within the configuration that can be repeated to specify the port mappings of the container. Each `ports` block supports the following:

- `internal` - (Required, int) Port within the container.
- `external` - (Required, int) Port exposed out of the container.
- `ip` - (Optional, string) IP address/mask that can access this port.
- `protocol` - (Optional, string) Protocol that can be used over this port, defaults to TCP.

» Extra Hosts

`host` is a block within the configuration that can be repeated to specify the extra host mappings for the container. Each `host` block supports the following:

- `host` - (Required, string) Hostname to add.
- `ip` - (Required, string) IP address this hostname should resolve to.

This is equivalent to using the `--add-host` option when using the `run` command of the Docker CLI.

» Volumes

`volumes` is a block within the configuration that can be repeated to specify the volumes attached to a container. Each `volumes` block supports the following:

- `from_container` - (Optional, string) The container where the volume is coming from.
- `host_path` - (Optional, string) The path on the host where the volume is coming from.
- `volume_name` - (Optional, string) The name of the docker volume which should be mounted.
- `container_path` - (Optional, string) The path in the container where the volume will be mounted.
- `read_only` - (Optional, bool) If true, this volume will be readonly. Defaults to false.

One of `from_container`, `host_path` or `volume_name` must be set.

» File Upload

`upload` is a block within the configuration that can be repeated to specify files to upload to the container before starting it. Each `upload` supports the following

- `content` - (Required, string) A content of a file to upload.
- `file` - (Required, string) path to a file in the container.

» Attributes Reference

The following attributes are exported:

- `ip_address` - The IP address of the container as read from its NetworkSettings.
- `ip_prefix_length` - The IP prefix length of the container as read from its NetworkSettings.
- `gateway` - The network gateway of the container as read from its NetworkSettings.
- `bridge` - The network bridge of the container as read from its NetworkSettings.

» docker_image

Note: The initial (current) version of this resource can only pull **public** images **from the official Docker Hub Registry**.

Pulls a Docker image to a given Docker host from a Docker Registry.

This resource will *not* pull new layers of the image automatically unless used in conjunction with `docker_registry_image` data source to update the `pull_triggers` field.

» Example Usage

```
# Find the latest Ubuntu precise image.
resource "docker_image" "ubuntu" {
  name = "ubuntu:precise"
}

# Access it somewhere else with ${docker_image.ubuntu.latest}
```

» Dynamic image

```
data "docker_registry_image" "ubuntu" {
  name = "ubuntu:precise"
}

resource "docker_image" "ubuntu" {
  name          = "${data.docker_registry_image.ubuntu.name}"
  pull_triggers = ["${data.docker_registry_image.ubuntu.sha256_digest}"]
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required, string) The name of the Docker image, including any tags.
- `keep_locally` - (Optional, boolean) If true, then the Docker image won't be deleted on destroy operation. If this is false, it will delete the image from the docker local storage on destroy operation.
- `pull_triggers` - (Optional, list of strings) List of values which cause an image pull when changed. This is used to store the image digest from the registry when using the `docker_registry_image` data source to trigger an image update.
- `pull_trigger` - **Deprecated**, use `pull_triggers` instead.

» Attributes Reference

The following attributes are exported in addition to the above configuration:

- `latest` (string) - The ID of the image.

» docker_network

Manages a Docker Network. This can be used alongside docker_container to create virtual networks within the docker environment.

» Example Usage

```
# Create a new docker network
resource "docker_network" "private_network" {
  name = "my_network"
}

# Access it somewhere else with ${docker_network.private_network.name}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required, string) The name of the Docker network.
- `check_duplicate` - (Optional, boolean) Requests daemon to check for networks with same name.
- `driver` - (Optional, string) Name of the network driver to use. Defaults to `bridge` driver.
- `options` - (Optional, map of strings) Network specific options to be used by the drivers.
- `internal` - (Optional, boolean) Restrict external access to the network. Defaults to `false`.
- `ipam_driver` - (Optional, string) Driver used by the custom IP scheme of the network.
- `ipam_config` - (Optional, block) See IPAM config below for details.

» IPAM config

Configuration of the custom IP scheme of the network.

The `ipam_config` block supports:

- `subnet` - (Optional, string)
- `ip_range` - (Optional, string)
- `gateway` - (Optional, string)
- `aux_address` - (Optional, map of string)

» Attributes Reference

The following attributes are exported in addition to the above configuration:

- `id` (string)
- `scope` (string)

» docker_volume

Creates and destroys a volume in Docker. This can be used alongside docker_container to prepare volumes that can be shared across containers.

» Example Usage

```
# Creates a docker volume "shared_volume".
resource "docker_volume" "shared_volume" {
  name = "shared_volume"
}

# Reference the volume with ${docker_volume.shared_volume.name}
```

» Argument Reference

The following arguments are supported:

- `name` - (Optional, string) The name of the Docker volume (generated if not provided).
- `driver` - (Optional, string) Driver type for the volume (defaults to local).
- `driver_opts` - (Optional, map of strings) Options specific to the driver.

» Attributes Reference

The following attributes are exported in addition to the above configuration:

- `mountpoint` (string) - The mountpoint of the volume.