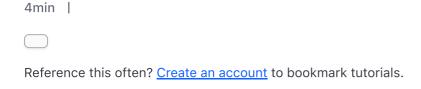
Developer / Terraform / Tutorials / Docker / Variables

# **Define input variables**



You now have enough Terraform knowledge to create useful configurations, but the examples so far have used hard-coded values. Terraform configurations can include variables to make your configuration more dynamic and flexible.

### **Prerequisites**

After following previous tutorials, you will have a directory named learn-terraform-dockercontainer with the following configuration in a file called main.tf.

Mac or Linux

Windows

```
terraform {
  required_providers {
   docker = {
     source = "kreuzwerker/docker"
     version = "~> 3.0.1"
 }
provider "docker" {}
resource "docker_image" "nginx" {
 name = "nginx:latest"
 keep_locally = false
resource "docker_container" "nginx" {
  image = docker_image.nginx.image_id
 name = "tutorial"
 ports {
   internal = 80
   external = 8080
```

Ensure that your configuration matches this, and that you have run terraform init in the learn-terraform-docker-container directory.

### Set the container name with a variable

The current configuration includes a number of hard-coded values. Terraform variables allow you to write configuration that is flexible and easier to re-use.

Add a variable to define the container name.

Create a new file called variables.tf with a block defining a new container\_name variable.

```
variable "container_name" {
  description = "Value of the name for the Docker container"
  type = string
  default = "ExampleNginxContainer"
}
```

#### Note

Terraform loads all files in the current directory ending in .tf, so you can name your configuration files however you choose.

In main.tf, update the docker\_container resource block to use the new variable. The container\_name variable block will default to its default value ("ExampleNginxContainer") unless you declare a different value.

```
resource "docker_container" "nginx" {
   image = docker_image.nginx.image_id
- name = "tutorial"
+ name = var.container_name
   ports {
     internal = 80
     external = 8080
   }
}
```

## **Apply your configuration**

Apply the configuration. Respond to the confirmation prompt with a yes.

```
$ terraform apply

docker_image.nginx: Refreshing state... [id=sha256:d1a364dc548d5357f0da3268c888e1971t
docker_container.nginx: Refreshing state... [id=e1df62aa5ae5cf9e31e59066c9e686fe0980t

Terraform used the selected providers to generate the following execution plan. Resol
following symbols:
    -/+ destroy and then create replacement

Terraform will perform the following actions:

    # docker_container.nginx must be replaced
    -/+ resource "docker_container" "nginx" {
##...

Plan: 1 to add, 0 to change, 1 to destroy.
```

Now apply the configuration again, this time overriding the default container name by passing in a variable using the \_var flag. Terraform will update the container's name attribute with the new name. Respond to the confirmation prompt with yes.

```
$ terraform apply -var "container_name=YetAnotherName"
docker_image.nginx: Refreshing state... [id=sha256:d1a364dc548d5357f0da3268c888e1971k
docker_container.nginx: Refreshing state... [id=ee653c84b8208dc141d0d79e1e4a276e3989a
Terraform used the selected providers to generate the following execution plan. Resolu
following symbols:
-/+ destroy and then create replacement
Terraform will perform the following actions:
  # docker container.nginx must be replaced
-/+ resource "docker_container" "nginx" {
##...
Plan: 1 to add, 0 to change, 1 to destroy.
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.
  Enter a value: yes
docker_container.nginx: Destroying... [id=ee653c84b8208dc141d0d79e1e4a276e3989a335933
docker_container.nginx: Destruction complete after 1s
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 2s [id=a7a6d308b8d4f77cda08ce20729f89
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

Setting variables via the command-line will not save their values. Terraform supports many ways to use and set variables so you can avoid having to enter them repeatedly as you execute commands. To learn more, follow our in-depth tutorial, <a href="Customize Terraform">Customize Terraform</a>
<a href="Configuration with Variables">Configuration with Variables</a>.

Was this tutorial helpful? Yes No

### Destroy

Next

Outputs



Theme

System

Certifications System Status Cookie Manager Terms of Use Security Privacy

Trademark Policy Trade Controls Accessibility Give Feedback