HashiCorp Learn

Sign in

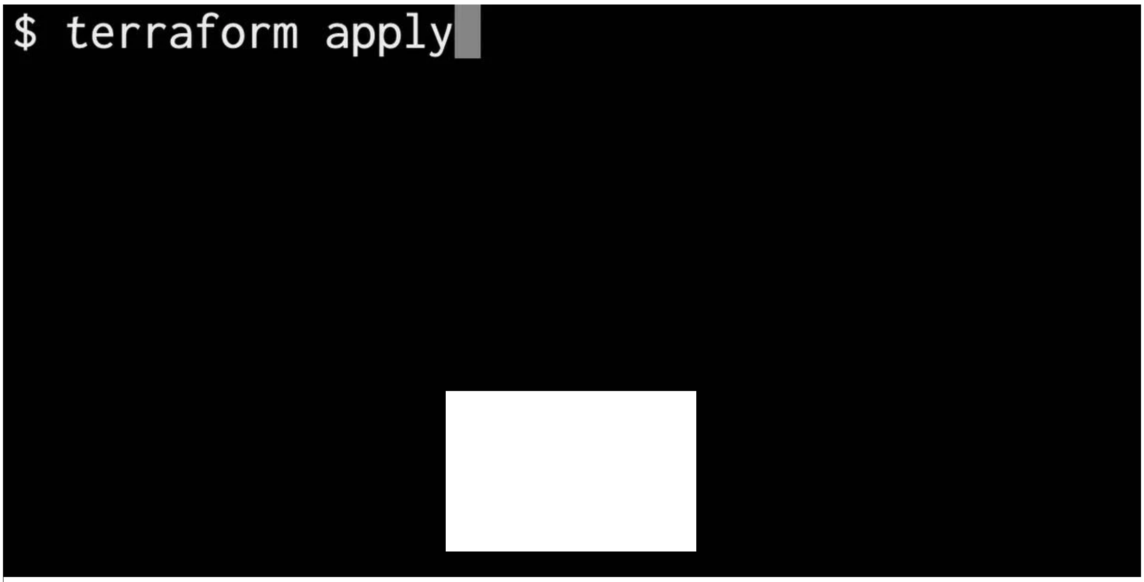
Show sidebar

Jump to section

Bookmark

Query Data with Outputs

3 MIN | PRODUCTS USED Terraform



```
$ terraform apply
```

1:48

In the previous tutorial, you used an input variable to parameterize your Terraform configuration. In this tutorial, you will use output values to organize data to be easily queried and displayed to the Terraform user.

If you have not yet completed the Define Input Variables tutorial, do so before following this one.

Initial configuration

After following the previous tutorials in this collection, you will have a directory named `learn-terraform-aws-instance` with the following configuration.

Copy

```
# main.tf

terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.27"
    }
  }

  required_version = ">= 0.14.9"
}

provider "aws" {
  profile = "default"
  region  = "us-west-2"
}

resource "aws_instance" "app_server" {
  ami           = "ami-08d70e59c07c61a3a"
  instance_type = "t2.micro"

  tags = {
    Name = var.instance_name
  }
}

# variables.tf

variable "instance_name" {
  description = "Value of the Name tag for the EC2 instance"
  type        = string
  default     = "ExampleAppServerInstance"
}
```

Ensure that your configuration matches this, and that you have initialized your configuration in the `learn-terraform-aws-instance` directory.

```
$ terraform init
```

Apply the configuration before continuing this tutorial. Respond to the confirmation prompt with a `yes` .

```
$ terraform apply
```

Output EC2 instance configuration

Create a file called `outputs.tf` in your `learn-terraform-aws-instance` directory.

Add the configuration below to `outputs.tf` to define outputs for your EC2 instance's ID and IP address.

```
output "instance_id" {
  description = "ID of the EC2 instance"
  value       = aws_instance.app_server.id
}

output "instance_public_ip" {
  description = "Public IP address of the EC2 instance"
  value       = aws_instance.app_server.public_ip
}
```

Inspect output values

You must apply this configuration before you can use these output values. Apply your configuration now. Respond to the confirmation prompt with `yes` .

```
$ terraform apply
aws_instance.app_server: Refreshing state... [id=i-0bf954919ed765de
```

```
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:


Terraform will perform the following actions:


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


Changes to Outputs:
  + instance_id        = "i-0bf954919ed765de1"
  + instance_public_ip = "54.186.202.254"


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



Apply complete! Resources: 0 added, 0 changed, 0 destroyed.


Outputs:


instance_id = "i-0bf954919ed765de1"
instance_public_ip = "54.186.202.254"
```

Terraform prints output values to the screen when you apply your configuration. Query the outputs with the `terraform output` command.

Copy

```
$ terraform output
instance_id = "i-0bf954919ed765de1"
instance_public_ip = "54.186.202.254"
```

You can use Terraform outputs to connect your Terraform projects with other parts of your infrastructure, or with other Terraform projects. To learn more, follow our in-depth tutorial, Output Data from Terraform.

Destroy infrastructure

Tip: If you plan to continue to the next tutorial in this collection, skip this destroy step.

Destroy your infrastructure. Respond to the confirmation prompt with `yes` .

Copy

```
$ terraform destroy

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.app_server will be destroyed
  - resource "aws_instance" "app_server" {
      - ami                          = "ami-08d70e59c07c61a3a" -> r
      - arn                          = "arn:aws:ec2:us-west-2:56165
##...

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

Changes to Outputs:
  - instance_id        = "i-0bf954919ed765de1" -> null
  - instance_public_ip = "54.186.202.254" -> null

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.app_server: Destroying... [id=i-0bf954919ed765de1]
aws_instance.app_server: Still destroying... [id=i-0bf954919ed765de
aws_instance.app_server: Still destroying... [id=i-0bf954919ed765de
aws_instance.app_server: Still destroying... [id=i-0bf954919ed765de
aws_instance.app_server: Destruction complete after 31s
```

```
Destroy complete! Resources: 1 destroyed.
```

Was this tutorial helpful?

Yes No

NEXT
Store Remote State ›

‹ PREVIOUS
Define Input Variables

stdin: is not a tty