RIFERIMENTO RAPIDO TERRAFORM

Provider, risorse, variabili, state, moduli

Nozioni di base

Flusso di lavoro principale

```
terraform init      # install providers & modules
terraform plan      # preview changes
terraform apply     # apply changes
terraform destroy   # tear down all resources
```

Comandi essenziali

`terraform init`	Inizializza la directory di lavoro, scarica i provider
`terraform plan`	Mostra il piano di esecuzione senza applicarlo
`terraform apply`	Applica le modifiche all'infrastruttura
`terraform destroy`	Elimina tutte le risorse gestite
`terraform fmt`	Formatta i file `.tf` secondo lo stile canonico
`terraform validate`	Controlla la sintassi della configurazione
`terraform show`	Mostra lo stato corrente o il piano
`terraform output`	Stampa i valori di output

Provider

Configurazione del provider

```
terraform {
  required_providers {
    aws = { source = "hashicorp/aws", version = "~> 5.0" }
  }
}
provider "aws" {
  region = "us-east-1"
}
```

Note sui provider

`source`	Indirizzo nel registry (`hashicorp/aws`, `hashicorp/google`)
`version`	Vincolo di versione (`~> 5.0`, `>= 3.0, < 4.0`)
`.terraform.lock.hcl`	File di lock — da committare nel controllo versione
`alias`	Usa più configurazioni per lo stesso provider

Risorse

Blocchi di risorse

```
resource "aws_instance" "web" {
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t3.micro"
  tags = { Name = "web-server" }
}
```

Meta-argomenti delle risorse

`depends_on`	Dipendenza esplicita da un'altra risorsa
`count`	Crea più istanze (`count = 3`)
`for_each`	Crea istanze da una mappa o set
`provider`	Seleziona un alias di provider non predefinito
`lifecycle`	Personalizza il comportamento di creazione/aggiornamento/distruzione

Riferimento alle risorse

```
# type.name.attribute
aws_instance.web.id
aws_instance.web.public_ip
aws_vpc.main.cidr_block
```

Variabili

Dichiarazione delle variabili

```
variable "region" {
  type    = string
  default = "us-east-1"
}
variable "instance_count" {
  type        = number
  description = "Number of instances"
}
```

Impostazione dei valori delle variabili

`-var 'region=us-west-2'`	Flag da riga di comando
`-var-file=prod.tfvars`	Carica da un file `.tfvars`
`terraform.tfvars`	Caricato automaticamente se presente
`TF_VAR_region`	Variabile d'ambiente
`Prompt interattivo`	Richiesto durante plan/apply se non c'è default

Tipi di variabile

`string`	`"us-east-1"`
`number`	`42`
`bool`	`true` / `false`
`list(string)`	`["a", "b"]`
`map(string)`	`{ key = "val" }`
`object({...})`	Tipo strutturato con attributi denominati

Output

Definizione degli output

```
output "instance_ip" {
  value       = aws_instance.web.public_ip
  description = "Public IP of the web server"
}
output "db_password" {
  value     = random_password.db.result
  sensitive = true
}
```

Comandi per gli output

`terraform output`	Stampa tutti gli output
`terraform output instance_ip`	Stampa un output specifico
`terraform output -json`	Formato JSON per gli script
`sensitive = true`	Nasconde il valore dall'output CLI
`module.vpc.vpc_id`	Accede agli output del modulo figlio

State

Backend remoto

```
terraform {
  backend "s3" {
    bucket = "my-tf-state"
    key    = "prod/terraform.tfstate"
    region = "us-east-1"
  }
}
```

Comandi per lo state

`terraform state list`	Elenca tutte le risorse nello state
`terraform state show <addr>`	Mostra gli attributi di una risorsa
`terraform state mv <src> <dst>`	Rinomina / sposta una risorsa nello state
`terraform state rm <addr>`	Rimuove la risorsa dallo state (mantiene l'infrastruttura)
`terraform state pull`	Scarica lo state remoto su stdout
`terraform import <addr> <id>`	Importa l'infrastruttura esistente nello state

Moduli

Utilizzo dei moduli

```
module "vpc" {
  source  = "terraform-aws-modules/vpc/aws"
  version = "~> 5.0"
  cidr    = "10.0.0.0/16"
}
```

Sorgenti dei moduli

`"./modules/vpc"`	Percorso locale
`"terraform-aws-modules/vpc/aws"`	Terraform Registry
`"github.com/org/repo//module"`	Repository GitHub
`"s3::https://bucket/module.zip"`	Bucket S3

Struttura del modulo

```
modules/vpc/
  main.tf       # resources
  variables.tf  # input variables
  outputs.tf    # output values
```

Sorgenti dati

Lettura di risorse esistenti

```
data "aws_ami" "ubuntu" {
  most_recent = true
  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/*"]
  }
  owners = ["099720109477"]
}
```

Sorgenti dati comuni

`data.aws_ami`	Cerca un'AMI tramite filtri
`data.aws_vpc`	Cerca un VPC esistente
`data.aws_caller_identity`	ID account AWS corrente
`data.aws_region`	Regione AWS corrente
`data.terraform_remote_state`	Legge gli output da un altro file state
`data.external`	Esegue un programma esterno per i dati

Lifecycle

Regole lifecycle

```
resource "aws_instance" "web" {
  lifecycle {
    create_before_destroy = true
    prevent_destroy       = true
    ignore_changes        = [tags]
  }
}
```

Opzioni lifecycle

`create_before_destroy`	Crea il sostituto prima di distruggere il vecchio
`prevent_destroy`	Errore se `terraform destroy` punta a questa risorsa
`ignore_changes`	Non rileva le derive sugli attributi elencati
`replace_triggered_by`	Forza la sostituzione quando la risorsa referenziata cambia
`precondition`	Valida le assunzioni prima dell'apply
`postcondition`	Valida i risultati dopo l'apply

Pattern comuni

Cicli e condizionali

```
# for_each with a map
resource "aws_iam_user" "users" {
  for_each = toset(["alice", "bob"])
  name     = each.value
}
# conditional resource
count = var.create_db ? 1 : 0
```

Funzioni utili

`file("key.pub")`	Legge il contenuto del file
`join(", ", list)`	Unisce una lista in una stringa
`lookup(map, key, default)`	Ricerca in una mappa con fallback
`length(list)`	Numero di elementi
`toset(["a", "b"])`	Converte una lista in set (per for_each)
`try(expr, fallback)`	Restituisce fallback se expr genera errori
`templatefile(path, vars)`	Renderizza un file template