shiftavenue

Rightsizing your Terraform Modules

Rene Schach - 03.06.2025
HashiDays London

shiftavenue

Introduction

RENE SCHACH

SENIOR CLOUD CONSULTANT

Google Cloud

Kubernetes

IaC

shiftavenue

What does „Rightsizing“ mean ?

RIGHTSIZING

shiftavenue

Scope considerations

How “big” should a module be ?

Code considerations

defining inputs/outputs, versioning

Security considerations

Input validation, defining level of configurability

Module testing

Nobody likes it, but it’s essential

MODULE SCOPE

- Listen to your module users/customers

- Keep modules cohesive and loosely coupled
  - For reusable modules: separation of concerns

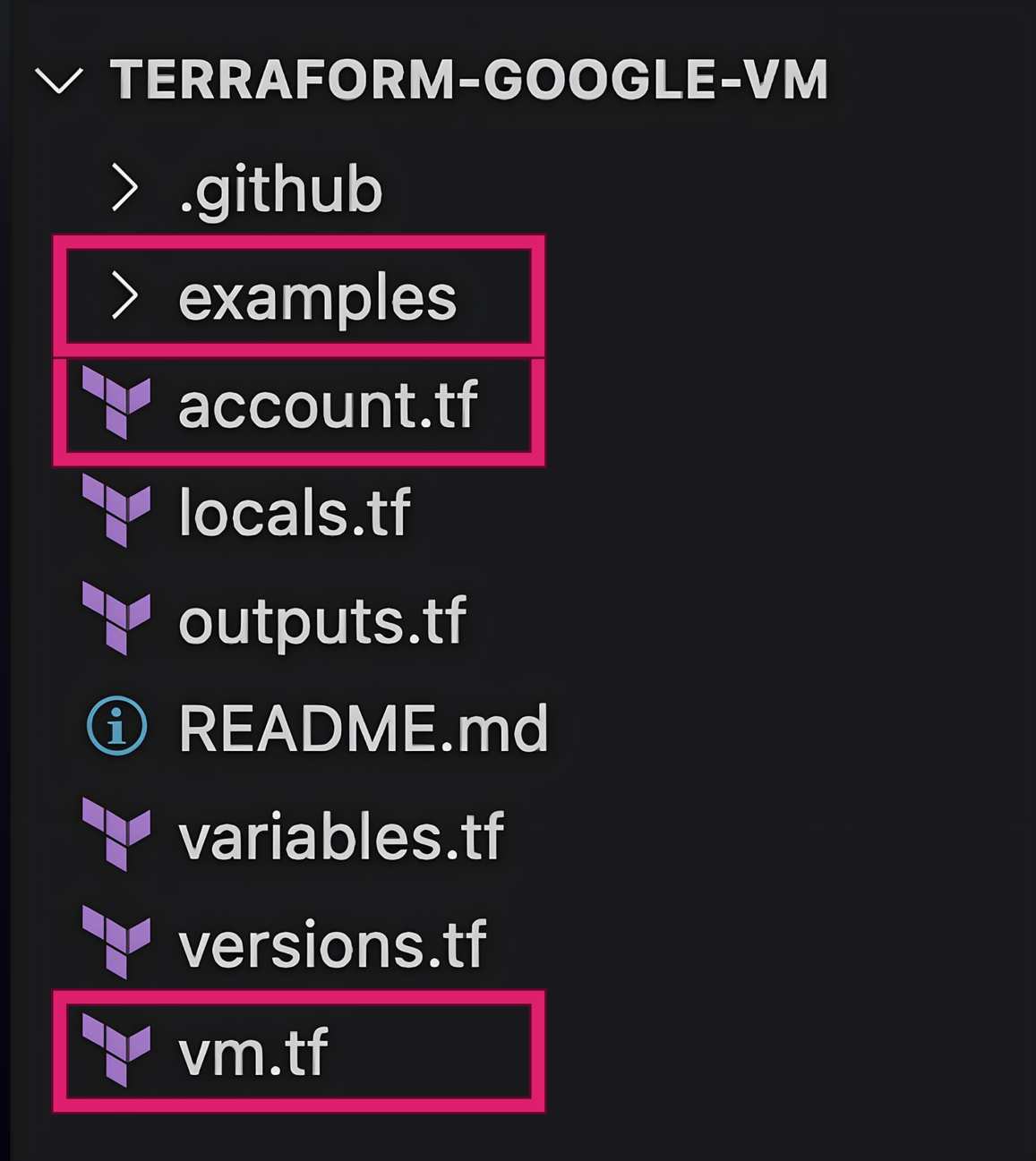
- Recommendations:
  - Group resources that belong together
  - Split resources based on their volatility

shiftavenue

MODULE STRUCTURE

- Opinion: split resources into files
- Sidenote: create usage examples

shiftavenue



CODE CONSIDERATIONS

- Defining inputs
  - Opinion: follow Terraform resource schemas
- Local variables
  - Opinion: all in one *locals.tf*
- Defining outputs
  - Opinion: output everything

shiftavenue

```
variable "instance_name" {
  description = "The VM name"
  type        = string
}

variable "location" {
  description = "The VM location"
  default     = "europe-west3-a"
}

variable "boot_disk_specs" {
  description = "Boot disk attributes"
  type = object({
    image = string # Operating system
    size  = optional(string, "100")
    type  = optional(string, "pd-ssd")
  })
}
```

VERSIONING

- Treat modules as a piece of software
- Common: semantic versioning
- Scenarios:
  - Input change
  - Output change
  - Resource changes
  - Provider upgrades

shiftavenue

SECURITY CONSIDERATIONS

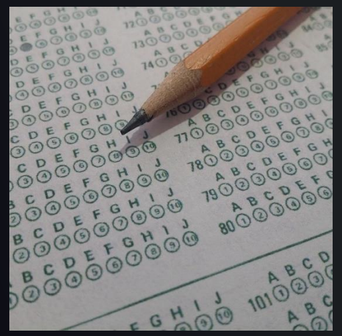
- Validation of submitted configuration
  - Validating input variables
  - Validation during *terraform plan*
  - In any case: validation before running the apply/rollout
- Defining the level of configurability of the module

```
variable "location" {
  description = "The VM location"
  type        = string

  validation {
    condition     = startswith(var.location, "europe-west3")
    error_message = "Frankfurt is the only allowed location"
  }
}
```

shiftavenue

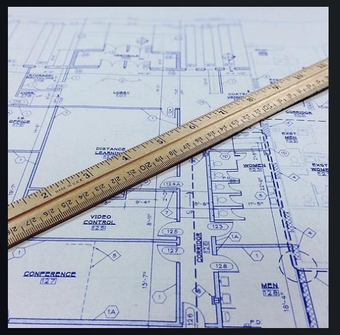
MODULE TESTING



Use the built-in *terraform test* framework

Implement into CI/CD setup (e.g., for pull request changes)

Don’t test each and every input scenario, focus on use-cases



Test module examples (examples directory)

WHAT ELSE ?

- Don’t hide your modules from development teams
  - improve contribution (issues, pull requests)
  - Improves acceptance of modules
- Define decision records (ADRs) that reflect the module implementation

FINAL WORDS

- Every organization has its own requirements
- Know when to not create a module
- Module development is a process
  - start with an MVP
  - make use of standard software development principles

shiftavenue

THANK YOU !