Terraform Workspaces - Complete Guide

1. Introduction

Terraform modules are containers for multiple resources that are used together. A module can be a single file or a directory containing multiple .tf files. They help in organizing and reusing infrastructure code.

2. Why Use Modules?

- Promote reusability of infrastructure code
- Organize complex configurations
- Simplify maintenance
- Improve readability
- Enable sharing across teams

3. Module Structure

A module typically consists of:

- main.tf: Contains the main resources
- variables.tf: Defines input variables
- outputs.tf: Defines output values
- README.md: Documentation

4. Calling a Module

You can call a module using the "module" block:

```
module "ec2_instance" {
  source        = "./modules/ec2"
  instance_type = "t2.micro"
  ami           = "ami-084e7e1456028650e"
}
```

5. Example Module (EC2)

Directory: modules/ec2/

variables.tf:

```
variable "instance_type" { type = string }
variable "ami" { type = string }
```

main.tf:

```
resource "aws_instance" "this" {
  ami           = var.ami
  instance_type = var.instance_type
}
```

outputs.tf:

```
output "instance_id" {
  value = aws_instance.this.id
}
```

6. Using Public Registry Modules

Terraform Registry hosts many pre-built modules:

Example:

```
module "vpc" {
  source  = "terraform-aws-modules/vpc/aws"
  version = "3.0.0"
  name    = "my-vpc"
  cidr    = "10.0.0.0/16"
  azs     = ["us-east-1a", "us-east-1b", "us-east-1c"]
  public_subnets = ["10.0.1.0/24", "10.0.2.0/24", "10.0.3.0/24"]
}
```

7. Best Practices

[OK] Keep modules small and focused

[OK] Use meaningful variable and output names

[OK] Document input/output in README

[OK] Use version pinning for registry modules

[OK] Avoid hardcoding values

8. Limitations

[Warning] Over-modularization can add complexity

[Warning] Too many nested modules may affect readability

[Warning] Not all registry modules are well-maintained

9. Full Workflow Example

```
# Create a module directory
mkdir -p modules/ec2
# Add main.tf, variables.tf, outputs.tf inside it
# Reference it in root configuration
terraform init
terraform apply
```