Terraform Workspaces - Complete Guide

1. Introduction

In Terraform, workspaces provide a way to manage multiple state files within the same configuration directory. They are useful when you need to deploy the same infrastructure in different environments (e.g., dev, staging, prod) without creating separate directories or repositories.

2. Why Use Workspaces?

Without workspaces, all changes affect the same state file. Workspaces allow multiple isolated environments using the same configuration. This avoids duplication of .tf files and enables easy testing before production changes.

Use cases:
- Separate dev/staging/prod deployments
- Multi-region deployments
- Customer-specific environments

3. Workspace Commands

```
terraform workspace list   -> Lists all workspaces
terraform workspace show   -> Shows current workspace
terraform workspace new <name> -> Creates a new workspace
terraform workspace select <name> -> Switches workspace
terraform workspace delete <name> -> Deletes a workspace
```

4. Workspace State Files

Terraform stores state files separately for each workspace inside terraform.tfstate.d/ ensuring environments don't overwrite each other.

5. Using terraform.workspace in Code

Terraform provides a built-in variable "terraform.workspace" which contains the current workspace name.
Example:

```
variable "environment" { type = string, default = "dev" }
resource "aws_instance" "example" {
  ami = "ami-xxxx"
  instance_type = terraform.workspace == var.environment ? "t2.micro" : "t2.medium"
}
```

6. Your Example Explained

```
provider "aws" { region = "ap-south-1" }
variable "environment" { type = string, default = "dev" }
resource "aws_instance" "second-ec2" {
  ami = "ami-084e7e1456028650e"
  instance_type = var.environment == "${terraform.workspace}" ? "t2.micro" : "t2.medium"
}
```

When in 'dev' workspace -> t2.micro, otherwise -> t2.medium.

7. Best Practices

[OK] Use workspaces for identical infra in multiple environments
[OK] Name them clearly: dev, staging, prod
[OK] Combine with variable files for more customization
[OK] Always check current workspace before applying

8. Limitations

[Warning] Workspaces are not a substitute for separate accounts/projects in high-security setups
[Warning] They only isolate state files, not resources themselves
[Warning] Variables/backend may still need per-environment overrides

9. Example Full Workflow

```
terraform init
terraform workspace new dev
terraform workspace new prod
terraform workspace select dev
terraform apply
terraform workspace select prod
terraform apply
```