Lab Exercise 6– Terraform Variables

Objective:

Learn how to define and use variables in Terraform configuration.

Prerequisites:

- **Install Terraform on your machine.**

Steps:

1. Create a Terraform Directory:

- **Create a new directory for your Terraform project.**

```
mkdir terraform-variables

cd terraform-variables
```

2. Create a Terraform Configuration File:

- **Create a file named main.tf within your project directory.**

 # main.tf

```
resource "aws_instance" "myinstance-1" {
    ami = var.myami
instance_type = var.my_instance_type count
    = var.mycount
tags = {
Name= "My Instance"
    }
}
```

```
resource "aws_instance" "myinstance-1" {
    ami = var.myami
    instance_type = var.my_instance_type
    count = var.mycount
    tags = {
      Name= "My Instance"
    }

}
```

3. Define Variables:

- **Open a new file named variables.tf. Define variables for region, ami, and instance_type.**

 # variables.tf

```
variable "myami" {
  type = string
  default = "ami-08718895af4dfa033"
}

variable "mycount" {

  type = number
  default = 5
}

variable "my_instance_type" {
   type = string
   default = "t2.micro"
}
```

Welcome | main.tf | variables.tf

variables.tf > variable "my_instance_type"

```
variable "myami" {
 type = string
 default = "ami-00bb6a80f01f03502"
}

variable "mycount" {

 type = number
 default = 5
}

variable "my_instance_type" {
 type = string
  default = "t2.micro"
}
```

4. Initialize and Apply:

- **Run the following Terraform commands to initialize and apply the configuration.**

```
terraform init
```

```
C:\Users\Lenovo\OneDrive\Desktop\SPCM Lab\LAB 6\terraform-variables>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

C:\Users\Lenovo\OneDrive\Desktop\SPCM Lab\LAB 6\terraform-variables>
```

terraform plan

```
      + cpu_options (known after apply)

      + ebs_block_device (known after apply)

      + enclave_options (known after apply)

      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

Plan: 5 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to t
you run "terraform apply" now.

C:\Users\Lenovo\OneDrive\Desktop\SPCM Lab\LAB 6\terraform-variables>
```

```
Plan: 5 to add, 0 to change, 0 to destroy.
aws_instance.myinstance-1[3]: Creating...
aws_instance.myinstance-1[2]: Creating...
aws_instance.myinstance-1[4]: Creating...
aws_instance.myinstance-1[1]: Creating...
aws_instance.myinstance-1[0]: Creating...
aws_instance.myinstance-1[3]: Still creating... [10s elapsed]
aws_instance.myinstance-1[4]: Still creating... [10s elapsed]
aws_instance.myinstance-1[2]: Still creating... [10s elapsed]
aws_instance.myinstance-1[1]: Still creating... [10s elapsed]
aws_instance.myinstance-1[0]: Still creating... [10s elapsed]
aws_instance.myinstance-1[1]: Creation complete after 12s [id=i-0caa7a0ace57f2cd4]
aws_instance.myinstance-1[2]: Still creating... [20s elapsed]
aws_instance.myinstance-1[3]: Still creating... [20s elapsed]
aws_instance.myinstance-1[4]: Still creating... [20s elapsed]
aws_instance.myinstance-1[0]: Still creating... [20s elapsed]
aws_instance.myinstance-1[0]: Creation complete after 21s [id=i-04d62fe873e44bf59]
aws_instance.myinstance-1[4]: Still creating... [30s elapsed]
aws_instance.myinstance-1[2]: Still creating... [30s elapsed]
aws_instance.myinstance-1[3]: Still creating... [30s elapsed]
aws_instance.myinstance-1[3]: Creation complete after 31s [id=i-0596eb806a9a6721c]
aws_instance.myinstance-1[2]: Creation complete after 31s [id=i-0c4168ec74514f2b0]
aws_instance.myinstance-1[4]: Creation complete after 32s [id=i-058b95a6e301a6c59]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

C:\Users\Lenovo\OneDrive\Desktop\SPCM Lab\LAB 6\terraform-variables>
```

Observe how the region changes based on the variable override.

5. Clean Up:

After testing, you can clean up resources.

terraform destroy

Confirm the destruction by typing yes.

```
aws_instance.myinstance-1[1]: Destroying... [id=i-0caa7a0ace57f2cd4]
aws_instance.myinstance-1[3]: Destroying... [id=i-0596eb806a9a6721c]
aws_instance.myinstance-1[0]: Still destroying... [id=i-04d62fe873e44bf59, 10s elapsed]
aws_instance.myinstance-1[4]: Still destroying... [id=i-058b95a6e301a6c59, 10s elapsed]
aws_instance.myinstance-1[2]: Still destroying... [id=i-0c4168ec74514f2b0, 10s elapsed]
aws_instance.myinstance-1[3]: Still destroying... [id=i-0596eb806a9a6721c, 10s elapsed]
aws_instance.myinstance-1[1]: Still destroying... [id=i-0caa7a0ace57f2cd4, 10s elapsed]
aws_instance.myinstance-1[0]: Still destroying... [id=i-04d62fe873e44bf59, 20s elapsed]
aws_instance.myinstance-1[4]: Still destroying... [id=i-058b95a6e301a6c59, 20s elapsed]
aws_instance.myinstance-1[2]: Still destroying... [id=i-0c4168ec74514f2b0, 20s elapsed]
aws_instance.myinstance-1[1]: Still destroying... [id=i-0caa7a0ace57f2cd4, 20s elapsed]
aws_instance.myinstance-1[3]: Still destroying... [id=i-0596eb806a9a6721c, 20s elapsed]
aws_instance.myinstance-1[4]: Still destroying... [id=i-058b95a6e301a6c59, 30s elapsed]
aws_instance.myinstance-1[0]: Still destroying... [id=i-04d62fe873e44bf59, 30s elapsed]
aws_instance.myinstance-1[1]: Still destroying... [id=i-0caa7a0ace57f2cd4, 30s elapsed]
aws_instance.myinstance-1[2]: Still destroying... [id=i-0c4168ec74514f2b0, 30s elapsed]
aws_instance.myinstance-1[3]: Still destroying... [id=i-0596eb806a9a6721c, 30s elapsed]
aws_instance.myinstance-1[1]: Destruction complete after 30s
aws_instance.myinstance-1[4]: Still destroying... [id=i-058b95a6e301a6c59, 40s elapsed]
aws_instance.myinstance-1[0]: Still destroying... [id=i-04d62fe873e44bf59, 40s elapsed]
aws_instance.myinstance-1[3]: Still destroying... [id=i-0596eb806a9a6721c, 40s elapsed]
aws_instance.myinstance-1[2]: Still destroying... [id=i-0c4168ec74514f2b0, 40s elapsed]
aws_instance.myinstance-1[3]: Destruction complete after 40s
aws_instance.myinstance-1[2]: Destruction complete after 40s
aws_instance.myinstance-1[0]: Destruction complete after 40s
aws_instance.myinstance-1[4]: Destruction complete after 40s

Destroy complete! Resources: 5 destroyed.
```

6. Conclusion:

This lab exercise introduces you to Terraform variables and demonstrates how to use them in your configurations. Experiment with different variable values and overrides to understand their impact on the infrastructure provisioning process.