Module 4

Variables and Functions

HashiCorp
Terraform

Agenda

- Introduction to Variables
- Input Variables
- Output Values
- Functions
- References to Named Values
- Interpolation
- Lab 4

Introduction to Variables

- Input variable act as parameters for Modules – “input parameters”

  Input variables are similar to function arguments

- Output Values return values from a module – “outputs”

  Output values are similar to function return values

HashiCorp Terraform

Input Variables

Input variables for a module are declared in a **variable block**

label unique name for variable within the module block

type defines the value types accepted

default apply this value if no other value is passed

```
variable “az_names" {
  type    = list(string)
  default = [“eu-west-1a"]
}
```


HashiCorp Terraform

Variable value precedence order

- Environment variable value
- Default variable value
- Terraform.tfvars value
- <name>.auto.tfvar value
- <name>.tfvars value
- -var value entered at runtime

Output Variables

- An output block is used to declare a value to be exported
- Can be used to output values to the command line
  Root module may return information back to the CLI
- Can be used to output values to other terraform configurations. A child module can pass some of its attributes back to the parent module where they can be referenced.
  Syntax: module.<module name>.<outputname>
  Example: module.instance_create.priv_ip_add
- Outputs are typically defined in outputs.tf file stored in module directory
- Query outputs with terraform output command

```
output “priv_ip_add" {
   value = aws_instance.myec2.private_ip
}
```

```
$ terraform output
priv_ip_add = “10.1.0.123"
```


HashiCorp
Terraform

Functions

- Terraform provide a number of functions that can be called within your expression
- Function syntax is <name_of_function> (arguments)
- Some of the function types are:

Numeric Functions	String Functions
Collection Functions	Encoding Functions
Filesystem Functions	Date and Time Functions
Hash and Crypto Functions	IP Network Functions
Type Conversion Functions	

- The Terraform language does not support user-defined functions

```
format("Your Region is, %s!", var.name)
Your Region is, eu-west-2!
```

```
Timestamp()
2022-05-13T07:08:45Z
```


HashiCorp Terraform

Reference to Named Values

- A name is an expression that references a value.
  - Can be a single expression or combined with other expressions

- They can be used as:
  - Resources
  - Input Variables
  - Local Values
  - Child Module outputs
  - Data Sources
  - Filesystem & Workspace info

Named Value Examples

1. **Conditional expression**. If we are using the ‘test’ workspace then deploy 2 VMs, otherwise deploy 10
2. **Index value.** Create instances named Server0 to Server1 or Server9 dependent upon workspace name.
3. **for_each string**. Extract each string from the set and use as the IAM user name.

1

```
resource "aws_instance" "small_ec2" {
  count = "${terraform.workspace == "test" ? 2 : 10}"
```

2

```
resource "aws_instance" "small_ec2" {
  count = "${terraform.workspace == "test" ? 2 : 10}"

  ami           = "ami-a1b2c3d4"
  instance_type = "t2.micro"

  tags = {
   Name = "Server ${count.index}"
  }
```

3

```
resource "aws_iam_user" “newiamusers" {
  for_each = toset( ["Bob", "Jo", “Charlie", "Scooby"] )
  name     = each.key
}
```


HashiCorp Terraform

Interpolation

- Interpolation is deprecated with effect from Ver .012 and is replaced by Configuration Language Expressions & Configuration Language Functions
- Configuration Language Expressions can be simple or complex
- Strings like “Hello World” or numerical values like 15 are simple expressions
- Complex expressions include condition evaluation, built-in functions and data exported from resources.

Expression value Data Types

- **String** a set of Unicode characters within double-quotes
- **Number** may be integer or include decimal point
- **Bool** true or false
- **List** a sequence of values where each element can be identified by its index number starting with 0 **[**valA, valB, valC**]** (index 1 would return valB)
- **Map** groups of values identified by its label **{** Region = “UK,” City = “Newcastle”**}**
- **Null**

Lab Group Discussion & Explore

Scoping

- What Type of Resources are you going to create ?
- What specific Properties do we need to consider for these resources ?

Workflow

- Are there any Implicit or Explicit dependencies ?
- What TF *Provider*, *Resource,* and *Modules* are required ?

Research

- Identify the TF documentation that will assist in developing the solution

HashiCorp Terraform

Lab 4

VPC deployment with variables and iterations

In this lab you will modify a ‘hard-coded’ deployment of a VPC, incorporating variables and iterations to allow for better code reusability and efficiency. You will also examine variable precedence.

HashiCorp Terraform