# TERRAFORM

* Why terraform ?

⇒ We create infrastructure in multiple provider using terraform. like AWS, Azure. etc.

| Terraform | Ansible. |
|---|---|
| * It is IAC (infrastructure as code) tool. | * You can define configuration of server as a code. |
| * Here we can create infrastructure. | |
| * Maintain the State of resources | |
| * We can directly create or delete the file. | * for delete we need to creat seperate file again. |
| * It is infrastructure as code tool. | * It is configuration management tool. |

## * Terraform installation :-

Go to → terraform.io/downloads.html.

download it.

extract zip file and set environmental path.

Go to cmd and run it.

» terraform --help.

Note :- Terraform file extension is :- .tf

i.e, Hello.tf

→ Terraform file must start with

```
block "lable1" lable2 .... {
    identifiers = expression
}
```

→

```
output "hello1" {
    value = "Hello world"
}
```

To execute it.

Go cmd & run >> terraform plan

outputs,

hello1 = Hello world.

→ You can terraform in JSON file.

→ Multiple blocks in single terraform file.

```
output "hello1" {
    value = "Hello"
}
output "hello2" {
    value = "world"
}
```

→ Multiple terraform file in same directory

→ Variables :——

Ex1: variable username { }

output printname {

value = var.username

}

o/p: var.username

Enter a value: Appu

Outputs:

printname = "Appu"

2) value = "Hello, ${var.username}"

o/p: Enter a value: Appu

printname = "Hello, Appu"

→ Pass variable value from command :-

>> terraform plan -var "username = Appu"

o/p :-

printname = "Hello, Appu"

→ Default variable :-

variable username {

default = "World"

}

>> terraform plan

o/p: printname = "Hello, World"

>> terraform plan -var "username = Appu"

o/p : printname = "Hello, Appu"

→ Multiple variable :-

output printname {

value = "Hello, ${var.username},

your age is ${var.age}"

}

>> terrform plan -var "username = Appu"

-var "age = 23"

o/p :- printname = Hello, Appu, Your age is 23.

→ Variable type :-

```
variable username {
    type = string
}

variable age {
    type = number
}
```

Note: type maybe,

- → String, number, bool
- → list < type >
- → map < type >
- → set < type >
- → object ({ < ATTR name > = < type >, .... })
- → tuple ([ < type, .... ])

→ List variable :-

```
variable users {
    type = list
}

output printname {
    value = "first user is ${var.users[0]}"
}
```

» terraform plan

var.users

Enter a value: [Appu, Sindhu, Shiva]

Output: printname = first user is Appu

→ Functions:

```
variable users {
    type = list
    default = [ "Appu", "Shinchu", "shivu" ]
}

output fun_join {
    value = "${join(">", var.users)}"
}

output fun_upper {
    value = "${upper(var.user[0])}"
}

output fun_lower {
    value = "${lower(var.user[1])}"
}

output fun_title {
    value = "${title(var.user[2])}"
}
```

join the variable using seperator.

Seperator btw value

Make all letters in Upper case.

All letters to lower ca

Make first letter of wor to Upper case.

>>> terraform plan

changes to outputs:

fun_join = "Appu > Shinchu > shivu

fun_lower = "shinchu"

fun_title = "Shivu"

fun_upper = "APPU"

Note: There are more number of different types of functions are their.

1) Numeric functions:

abs, ceil, floor, log, max, min, parseint, power, signum.

2) String functions:

chomp, format, formatlist, indent, join, lower, upper, regex, regexall, replace, split, strrev, substr, title, trim, trimprefix, trimsuffix, trimspace,

3) Collection functions :-

alltrue, anytrue, chunklist, coalesce, coalescelist, compact, concat, contains, distinct, element, flatten, index, key, length, list, lookup, map, matchkeys, merge, one, range, reverse, setintersection, setproduct, setsubstract, setunion, slice, ~~short~~, sort, sum, transpose, values, zipmap.

4) Encoding functions :-

base64decode, base64encode, base64gzip, csvdecode, jsondecode, jsonencode, textdecodebase64, textencodebase64, urlencode, yamldecode, yamlencode.

5) Filesystem functions :-

abspath, dirname, pathexpand, basenam file, fileexists, fileset, filebase64, templatefile.

6) Date & time function :-

formatdate, timeadd, timestamp.

7) Hash & crypto functions :

8) IP network functions :-

9) Type conversion functions :-

can, defaults, nonsensitive, sensitive, tobool tolist, tomap, tonumber, toset, tostring, try, type.