Using a Module for Common Configurations

Ned Bellavance

HashiCorp Ambassador

@ned1313 | nedinthecloud.com

Module Overview

What is a module?

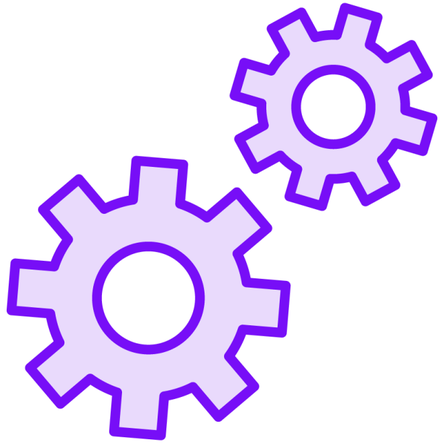
Globomantics updates

Using existing modules

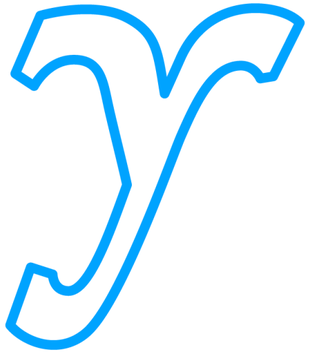
Creating new modules

Terraform Modules

Input variables

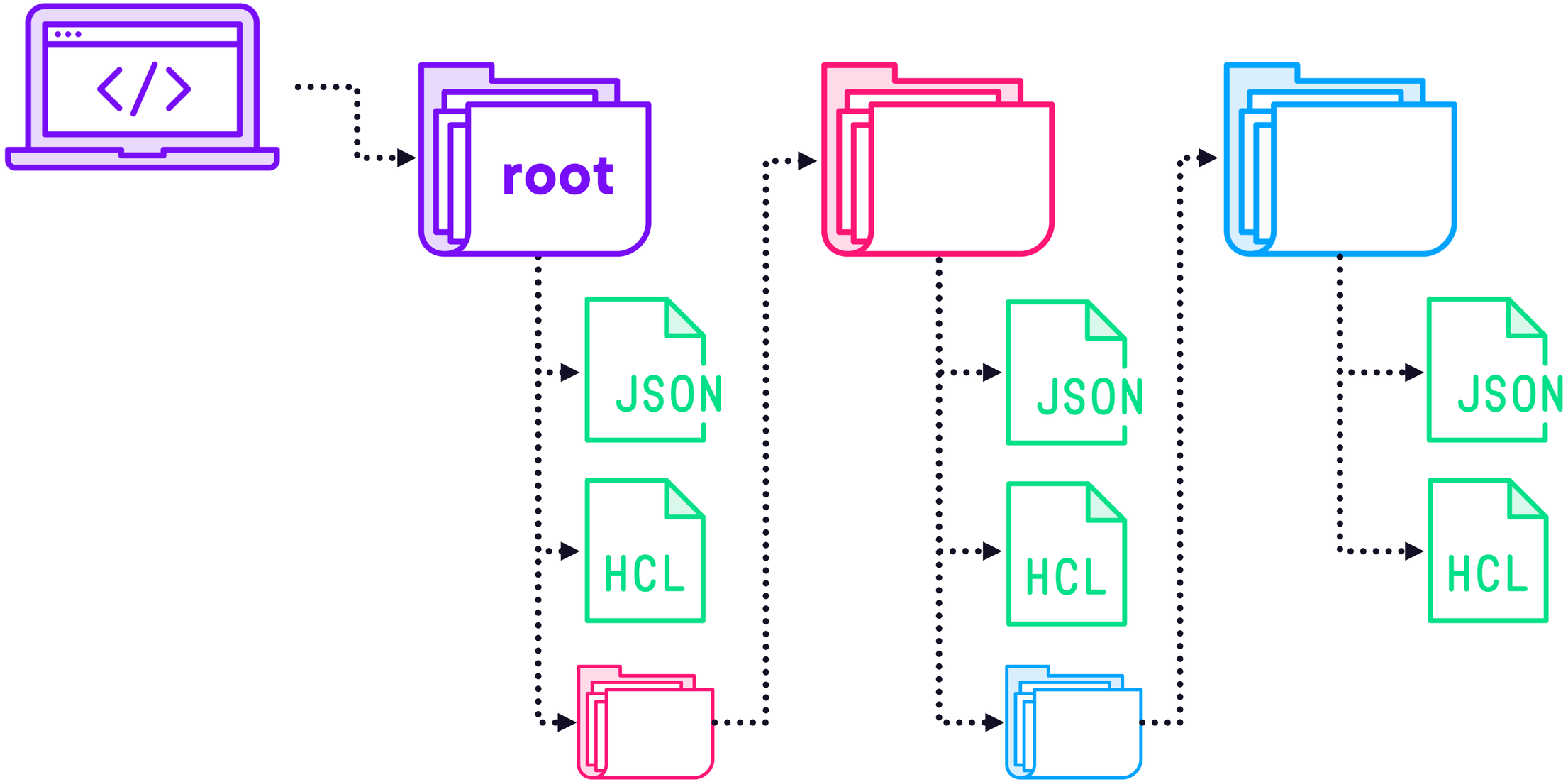


Resources and data sources

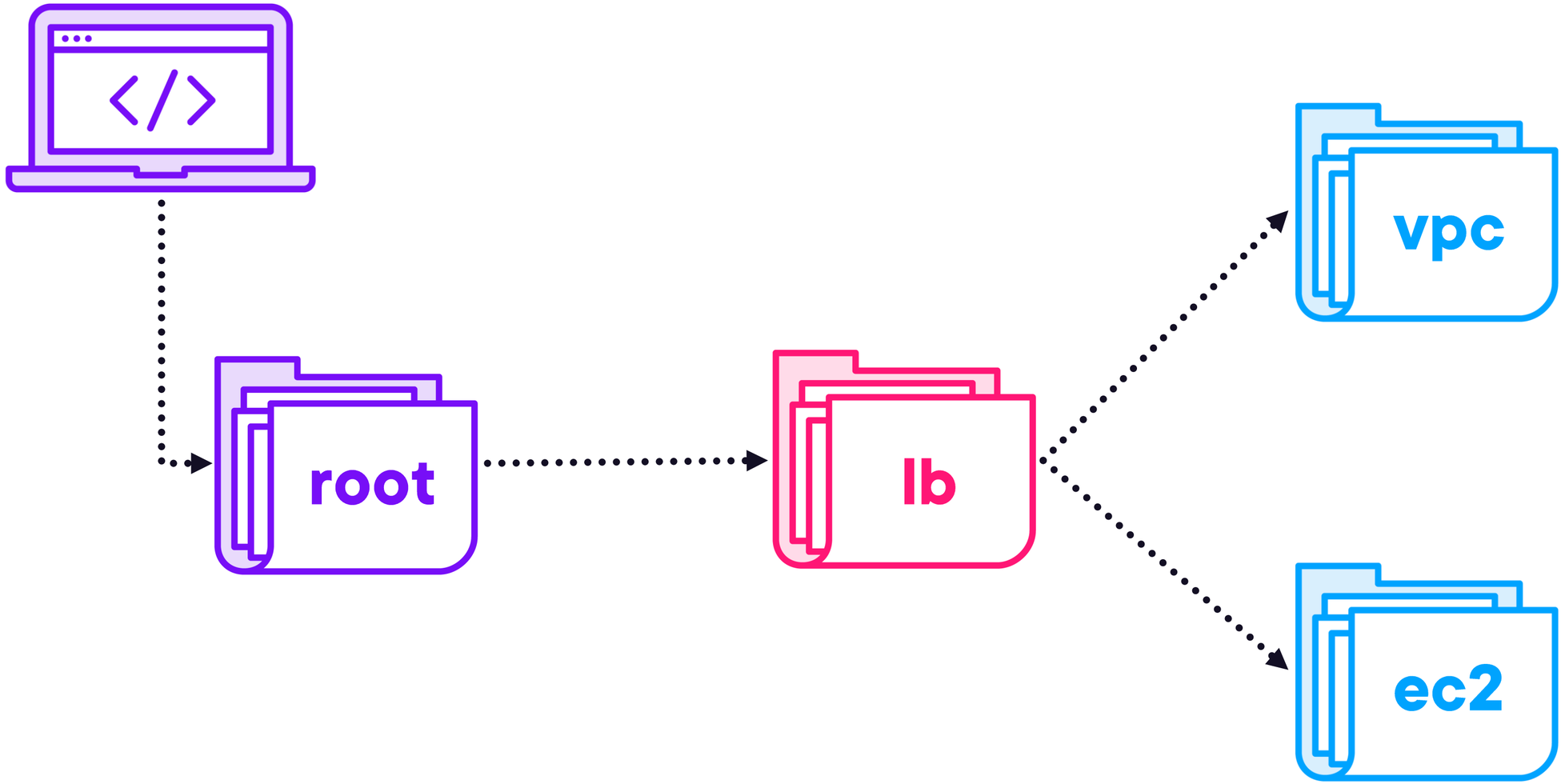


Output values

Terraform Modules – File Structure



Terraform Modules – Child Modules



Terraform Modules

Code reuse

Remote or local source

Versioning

Terraform init

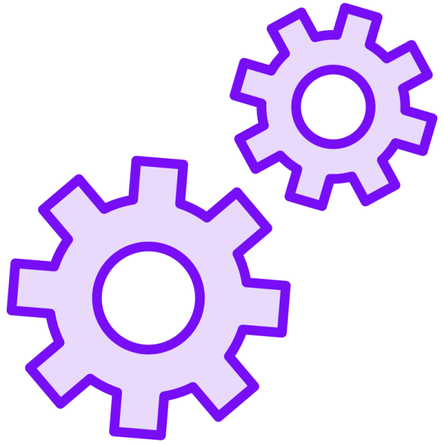
Multiple instances

Inherit provider instances

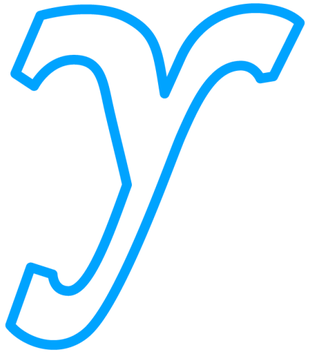
- Providers argument

Terraform Modules

Input variables



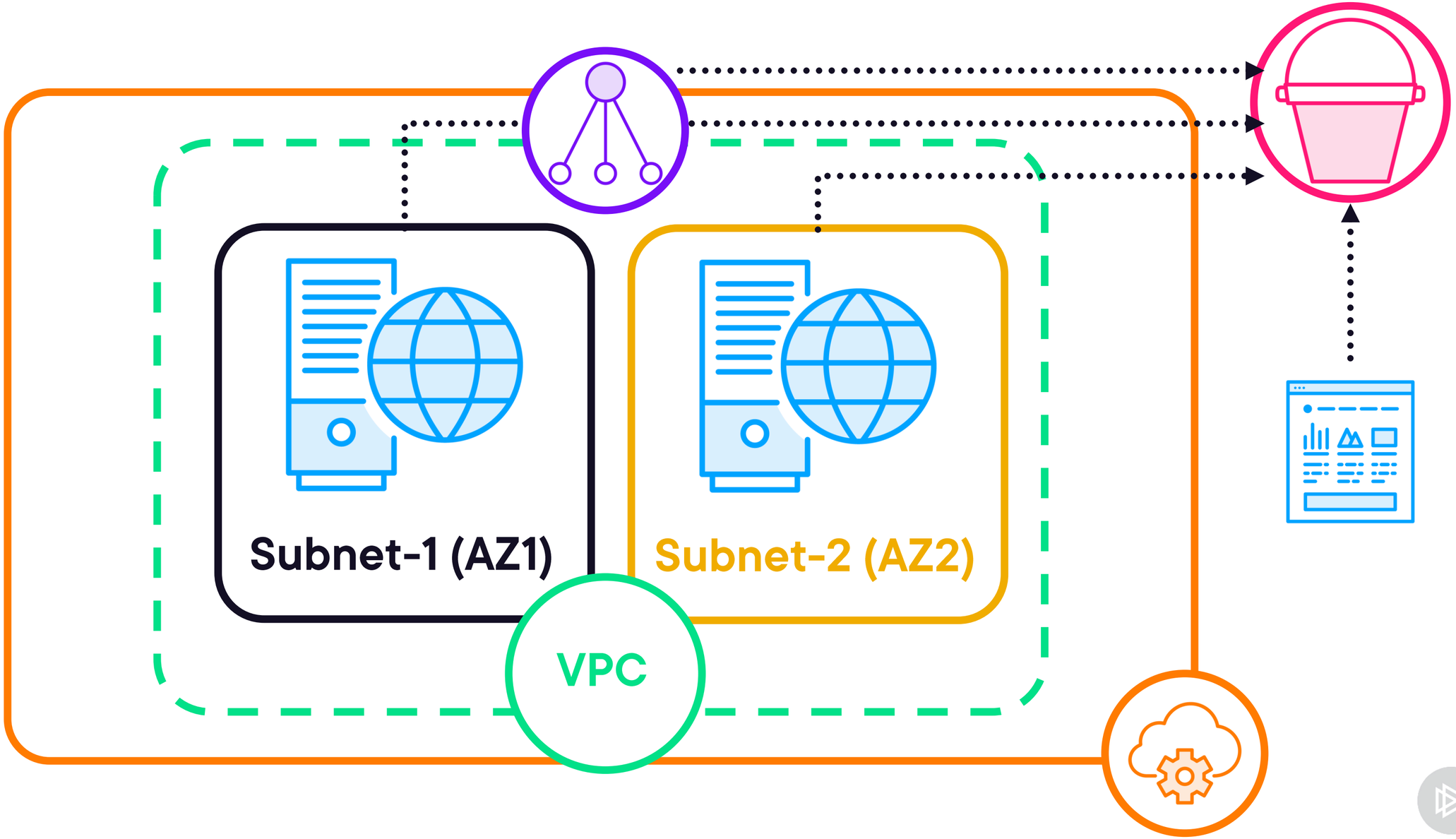
Resources and data sources



Output values

Globomantics Scenario

Deployment Architecture



Potential Improvements

Leverage the VPC module for networking

Create a module for S3 buckets

- Include load balancer permissions
- Include instance profile permissions

Module Structure and Syntax

Module Structure

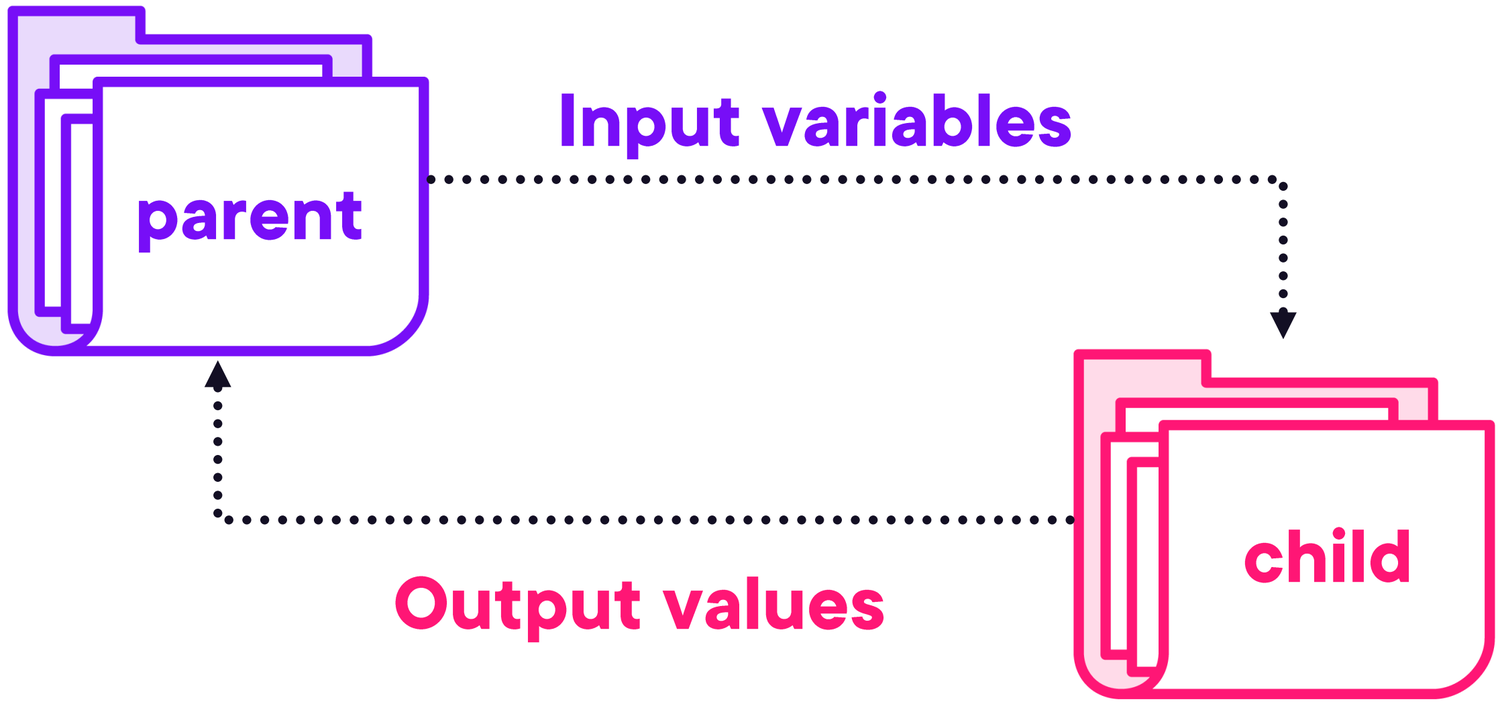
s3/main.tf

```
variable “bucket_name" {}

resource "aws_s3_bucket" "main"{
  name = var.bucket_name
}

output "bucket_id" {
  value = aws_s3_bucket.bucket.id
}
```

Module Scoping



Module Block Syntax

s3.tf

```
module "name_label" {
  source = "local_or_remote_source"
  version = "version_expression"
  providers = {
    module_provider = parent_provider
  }

  # Input variable values...
}
```

Module Block Syntax

s3.tf

```
module "tacos" {
  source = "./modules/s3"

  # Input variable values...
  bucket_name = "taco_bucket"
}
```

Module Scoping

Parent modules cannot directly reference child module objects

modules/s3/main.tf

```
variable “bucket_name" {}

resource "aws_s3_bucket" "main"{
  name = var.bucket_name
}

output "bucket_id" {
  value = aws_s3_bucket.bucket.id
}
```

main.tf

```
module "tacos" {
  source = "./modules/s3"
  bucket_name = "taco_bucket"
}

locals {
  taco_id =
module.tacos.aws_s3_bucket.main.id
}
```

Module Scoping

Parent modules cannot directly reference child module objects

modules/s3/main.tf

```
variable “bucket_name" {}

resource "aws_s3_bucket" "main"{
  name = var.bucket_name
}

output "bucket_id" {
  value = aws_s3_bucket.bucket.id
}
```

main.tf

```
module "tacos" {
  source = "./modules/s3"
  bucket_name = "taco_bucket"
}

locals {
  taco_id = module.tacos.bucket_id
}
```

Module Scoping

Parent modules cannot directly reference child module objects

modules/s3/main.tf

```
variable “bucket_name" {}

resource "aws_s3_bucket" "main"{
  name = var.bucket_name
}

output "bucket_id" {
  value = aws_s3_bucket.bucket.id
}
```

main.tf

```
module "tacos" {
  source = "./modules/s3"
  bucket_name = "taco_bucket"
}

locals {
  taco_id = module.tacos.bucket_id
}
```

For expressions create a new collection based off an existing one, transforming data as needed.

For Expressions

Input types
List, set, tuple, map, or object

Result types
Tuple or object

Filtering with if statement

For Expression Tuple Result

main.tf

```
# Create a tuple
[ for item in items : tuple_element ]
```

```
# Example
locals {
  toppings = ["cheese", "lettuce", "salsa"]
}

[ for t in local.toppings : "Globo ${t}" ]

# Result
["Globo cheese", "Globo lettuce" , "Globo salsa"]
```

For Expression Object Result

main.tf

```
# Create an object
{ for key, value in map : obj_key => obj_value }
```

```
# Example
locals {
  prices = {
    taco = "5.99"
    burrito = "9.99"
    enchilada = "7.99"
  }
}

{ for i, price in local.prices : i => ceil(price) }

# Result
{ taco = "6", burrito = "10", enchilada = "8"}
```

S3 Module

```
# Input variables - variables.tf
"bucket_name" # Name of bucket
"elb_service_account_arn" # ARN of ELB service account
"common_tags" # Tags to apply to resources

# Resources - main.tf
"aws_s3_bucket"
"aws_s3_bucket_policy"
"aws_iam_role"
"aws_iam_role_policy"
"aws_iam_instance_profile"

# Outputs - outputs.tf
"web_bucket" # Full bucket object
"instance_profile" # Full instance profile object
```

Moving Resources

```
# Option 1 - use terraform state mv command
terraform state mv OLD_ADDR NEW_ADDR
terraform state mv aws_subnet.public_subnets[0] module.app.aws_subnet.public[0]

# Option 2 - use moved block
moved {
  from = aws_subnet.public_subnets[0]
  to   = module.app.aws_subnet.public[0]
}
```

Module Summary

- Modules enable code reuse
- Common configurations
- Root module

Course Summary

Build infrastructure automagically

Ensure consistent repeatable deployment

Reuse existing configurations

Increase your productivity

Make your job better or find a better job!

Next Steps

- Terraform courses on Pluralsight
- Terraform Associate certification
- HashiCorp Vault or Packer
- Terraform Tuesdays

Go Build Something Great!

nedinthecloud.com