Terraform

Taming Complexity

Presented by Sean P. Kane

https://techlabs.sh/

Release 2024-02-16-1510

Instructor

Sean P. Kane

@spkane

techlabs

https://techlabs.sh

Follow Along Guide

Textual Slides

O'Reilly Online Sandbox VM

https://learning.oreilly.com/interactive-lab/devops-tools-sandbox/9781098126469

NOTE: *VM sessions will expire after 60 minutes!*

Prerequisites (1 of 3)

NOTE: You *MUST* have open access to Github if you want to participate in the hands-on portion of class from your local computer system.

Prerequisites (2 of 3)

- A recent computer and OS
  - Recent/Stable Linux, macOS, or Windows 10+
  - Reliable and fast internet connectivity
- Hashicorp Terraform

Prerequisites (2 of 2)

- A graphical web browser
- A text editor
- A software package manager
- `git` client
- General comfort with the command line will be helpful.
- [optional] `tar`, `wget`, `curl`, `jq`, `ssh` client
  - `curl.exe` for Windows - https://curl.se/windows/

A Note for Powershell Users

Terminal commands reflect the Unix bash shell. PowerShell users will need to adjust the commands.

- Unix Shell Variables
  - `export MY_VAR='test'`
  - `echo ${MY_VAR}`
- PowerShell Variables
  - `$env:my_var = "test"`
  - `Get-ChildItem Env:my_var`
  - `Remove-Item Env:\my_var`

Translation Key

`\` - Unix Shell Line Continuation
`` ` `` - Powershell Line Continuation (sort of)

`${MY_VAR}` - Is generally a placeholder in the slides.

A Note About Proxies & VPNs

Proxies can interfere with some activities if they are not configured correctly and VPNs can increase audio and video streaming issues in class.

- Configure `HTTP_PROXY`, `HTTPS_PROXY`, and your web browser.
- Terraform
- Docker
- Docker-Compose

Instructor Environment

- **Operating System**: macOS (v14.X.X+)
- **Terminal**: iTerm2 (Build 3.X.X+) - https://www.iterm2.com/
- **Shell Prompt Theme**: Starship - https://starship.rs/
- **Shell Prompt Font**: Fira Code - https://github.com/tonsky/FiraCode
- **Text Editor**: Visual Studio Code (v1.X.X+) - https://code.visualstudio.com/
- `export BUILDKIT_COLORS=run=green:warning=yellow:error=red:cancel=cyan`

Code Setup

- If you missed the first class.

```
$ cd ${HOME}
$ mkdir class
$ cd ${HOME}/class
$ git clone https://github.com/spkane/todo-for-terraform
$ cd todo-for-terraform
```

Spin up the Todo Server

```
$ cd ${HOME}/class
$ cd todo-for-terraform/terraform-infrastructure-aws
$ . ~/bin/ns1_personal
$ terraform apply
$ . ./bin/ip_vars.sh
$ cd ..
```

- Note: Students **CAN NOT** run `terraform` in the `terraform-infrastructure-aws` directory. This is for the instructor only.

Copy the Code

- `cd $HOME/class/todo-for-terraform`
- `mkdir -p tf-code`
- `cp -a terraform-tests-2 tf-code`
- `cd ./tf-code/terraform-tests-2`

A Real World Provider

- GitHub provider
  - https://registry.terraform.io/providers/integrations/github/latest/docs

Creating a GitHub Account

- https://github.com/signup
  - You will need to provide an:
    - email address
    - strong password
    - username (globally unique)
  - and answer one question and complete one small robot test.
- You will then receive an email with a code that you will need to enter.

Login to GitHub

- https://github.com/login

Create a Token

- Creating a GitHub Personal Token
  - https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token
- Keep this secure, but don't lose it!
  - You will need it next week as well.

Token Settings

- **Navigate to**: https://github.com/settings/tokens
- **Click**: `Generate new token` - `Generate new token (classic)`
  - **Note**: `Terraform (Repo Admin)`
  - **Expiration**: `7 Days`
  - **Select scopes**:
    - `repo` (all sub-sections)
    - `delete_repo`

Add an SSH Key

If you do not already have an SSH key in Github, then you will probably want to consider adding one for next week's class.

- Navigate to:
  - https://github.com/settings/keys
- Click `New SSH key`
- Set `Title` to anything useful
- Drop your **OpenSSH Public Key** into the `Key` text box
- Click `Add SSH Key`
- Need to create an SSH key? https://techlabs.sh

Add the GitHub Provider

- in `main.tf`

```
terraform {
  required_providers {
    todo = {
…
    }
    github = {
      source  = "integrations/github"
      version = "~> 5.0"
    }
  }
}
```

Configure the GitHub Provider

- in `main.tf`

```
provider "github" {
  token = var.github_token
}
```

- https://registry.terraform.io/providers/integrations/github/latest/docs#authentication
- **Note**: It is actually better to avoid using terraform variable for authentication secrets when possible. We will cover this more in the next class

Add Variable for GitHub Token

- in `variables.tf`

```
variable "github_token" {
    type = string
    description = "The token required to authenticate against GitHub"
}
```

- **Note**: It is actually better to avoid using terraform variable for authentication secrets when possible. We will cover this more in the next class.

Define Token Variable in Environment

```
$ export TF_VAR_github_token=${GITHUB_TOKEN}
```

- I am going to run `$ . ~/bin/gh_personal` to set this up locally.
- **Note**: The GitHub provider support reading the environment variable `GITHUB_TOKEN` directly, without relying on Terraform variables. We are using a variable for this example, primarily so people do not need to potentially over ride that env var if they are already using it on their local systems.

Initialize the Provider

```
$ terraform init

Initializing the backend…
Initializing provider plugins…
…
Terraform has been successfully initialized!
…
```

Introducing a Real World Module

- GitHub Repository module
  - https://github.com/spkane/terraform-github-repository

Configure the module

- in `main.tf` (*combine broken lines*)

```
module "simple_github_repo" {
  source  = "github.com/spkane/terraform-github-repository?
ref=8b72dcbac2c4287672a22435e36bbc27a869db5c"

  name            = "testing-terraform-modules"
  description     = "Testing GitHub repo automation via Terraform.
This repo should probably be deleted."
  visibility      = "private"
  auto_init       = true
  default_branch = "main"
  has_issues      = "false"
}
```

Module Sources Documentation

- https://developer.hashicorp.com/terraform/language/modules/sources

Add Output for module

- in `outputs.tf`

```
output "repo_url" {
  value = module.simple_github_repo.github_repository.html_url
}
```

- https://github.com/spkane/terraform-github-repository/blob/main/output.tf
- https://registry.terraform.io/providers/integrations/github/latest/docs/resources/repository#attributes-reference

Initialize the Module

```
$ terraform init

Initializing modules…
…
Initializing the backend…
Initializing provider plugins…
…
Terraform has been successfully initialized!
…
```

Plan for GitHub Module

```
$ terraform plan

…
Plan: 6 to add, 0 to change, 0 to destroy.
…
```

Apply for GitHub Module

```
$ terraform apply

…
Apply complete! Resources: 6 added, 0 changed, 0 destroyed.

Outputs:

repo_url = "https://github.com/${USERNAME}/testing-terraform-modules"
…
```

View the New GitHub Repo

- In your web browser, open up:
  - `https://github.com/${USERNAME}/testing-terraform-modules`

Explore the Module in Depth

- GitHub Repository module
  - https://github.com/spkane/terraform-github-repository

Meta-Argument: `count`

- in `main.tf`

```
module "simple_github_repo" {
  …
  count = var.create_repo ? 1 : 0
  …
}
```

Optional Creation Variable

- in `variables.tf`

```
variable "create_repo" {
    type        = bool
    description = "Should we create the repo?"
    default     = true
}
```

Fix The Module Output

- in `outputs.tf`

```
output "repo_url" {
  value = module.simple_github_repo[0].github_repository.html_url
}
```

Apply Module with Count

- `terraform apply`
- It wants to delete our repo. 😕
- There are a few solutions.

The Moved Block

- in `main.tf`

```
moved {
  from = module.simple_github_repo
  to   = module.simple_github_repo[0]
}
```

- `terraform plan`
  - If you applied this you could remove the moved block after that.

Prepare The State

```
$ terraform state mv module.simple_github_repo \
    module.simple_github_repo[0]
$ terraform plan
```

Logic

- meta-arguments (*e.g.* `count`, `for_each`, `depends_on`, *etc.*)
  - https://developer.hashicorp.com/terraform/language/meta-arguments/
- expressions (*e.g.* `for`, `[*]`, `? :`, *etc.*)
  - https://developer.hashicorp.com/terraform/language/expressions
- functions (*e.g.* `join()`, `upper()`, `max()`, *etc.*)
  - https://developer.hashicorp.com/terraform/language/functions

Meta-Argument: `for_each`

- in `main.tf`

```
resource "todo_todo" "step_2" {
  for_each    = {
    for repo in tolist(module.simple_github_repo[*]) :
    repo.github_repository.name => ({
      url = repo.github_repository.html_url
    })
  }

  description = "${each.value.url}: ${var.purpose} todo"
  completed = false
}
```

`for_each` Outputs

- in `outputs.tf`

```
output "todo_ids" {
  value = {
    for k, v in todo_todo.step_2 : k => v.id
  }
}
```

`for_each` Plan

```
$ terraform plan
…
Plan: 1 to add, 0 to change, 5 to destroy.

Changes to Outputs:
  ~ todo_ids = [
      - 19,
      - 16,
      - 20,
      - 17,
      - 18,
    ] -> {
      + testing-terraform-modules = (known after apply)
    }
```

for_each Apply

```
$ terraform apply
…
Apply complete! Resources: 1 added, 0 changed, 5 destroyed.

Outputs:
repo_url = "https://github.com/${USERNAME}/testing-terraform-modules"
todo_ids = {
  "testing-terraform-modules" = 21
}
```

`for_each` Todo Output

```
$ terraform state show 'todo_todo.step_2["testing-terraform-modules"]'

# todo_todo.step_2["testing-terraform-modules"]
resource "todo_todo" "step_2" {
    completed   = false
    description = "https://github.com/${USERNAME}/
testing-terraform-modules: testing-modules todo"
    id          = 21
}
```

Multiple Repos

- in `main.tf` (*combine broken lines*)

```
module "simple_github_repo" {
  source  = "github.com/spkane/terraform-github-repository?
ref=8b72dcbac2c4287672a22435e36bbc27a869db5c"

  count           = var.repo_count
  name            = "testing-terraform-modules-${count.index}"
  description     = "Testing GitHub repo automation via Terraform.
This repo should probably be deleted.
Repo #${count.index + 1} out of ${var.repo_count}"
  …
}
```

Multiple Repo Variables

- in `variables.tf`
  - replace `create_repo` with:

```
variable "repo_count" {
    type        = number
    description = "The number of repos that we want."
    default     = 1
    validation {
      condition     = ( var.repo_count >= 0 && var.repo_count <= 3 )
      error_message = "Must be an integer between 0 and 3."
    }
}
```

Multiple Repo Outputs

- in `outputs.tf`

```
output "repo_url" {
  value = module.simple_github_repo[*].github_repository.html_url
}
```

Multiple Repo Apply

```
$ terraform apply
…
Plan: 1 to add, 1 to change, 1 to destroy.

Changes to Outputs:
  ~ repo_url = "https://github.com/${USERNAME}/testing-terraform-modules" ->
    [
      + "https://github.com/${USERNAME}/testing-terraform-modules-0",
    ]
  ~ todo_ids = {
      - testing-terraform-modules   = 21 -> null
      + testing-terraform-modules-0 = (known after apply)
    }
```

Pass in an Invalid Variable

```
$ terraform apply -var="repo_count=5"
…
╷
│ Error: Invalid value for variable
│
│   on variables.tf line 12:
│   12: variable "repo_count" {
│     ├────────────────
│     │ var.repo_count is 5
│
│ Must be an integer between 0 and 3.
│
│ This was checked by the validation rule at variables.tf:16,5-15.
```

Pass in a Valid Variable

```
$ terraform apply -var="repo_count=3"
…
Plan: 4 to add, 1 to change, 0 to destroy.

Changes to Outputs:
  ~ repo_url = [
        "https://github.com/${USERNAME}/testing-terraform-modules-0",
      + (known after apply),
      + (known after apply),
    ]
  ~ todo_ids = {
      + testing-terraform-modules-1 = (known after apply)
      + testing-terraform-modules-2 = (known after apply)
        # (1 unchanged element hidden)
    }
```

Multiple Repo Results

```
…
Apply complete! Resources: 4 added, 1 changed, 0 destroyed.
…
repo_url = [
  "https://github.com/${USERNAME}/testing-terraform-modules-0",
  "https://github.com/${USERNAME}/testing-terraform-modules-1",
  "https://github.com/${USERNAME}/testing-terraform-modules-2",
]
todo_ids = {
  "testing-terraform-modules-0" = 22
  "testing-terraform-modules-1" = 23
  "testing-terraform-modules-2" = 24
}
```

Multiple Repo Todos

```
$ terraform state show 'todo_todo.step_2["testing-terraform-modules-0"]'
$ terraform state show 'todo_todo.step_2["testing-terraform-modules-1"]'
$ terraform state show 'todo_todo.step_2["testing-terraform-modules-2"]'
```

Making Variables Stick

- Running `terraform plan` will report that it is going to delete 2 of our 3 repos and todos.

```
Plan: 0 to add, 1 to change, 4 to destroy.
```

- Let's fix that!

Auto Variables

- create `local-vars.auto.tfvars`

```
repo_count=3
```

- https://developer.hashicorp.com/terraform/language/values/variables#variable-definitions-tfvars-files

Working Auto Variables

- Running `terraform plan` will now report "No changes".

```
No changes. Your infrastructure matches the configuration.
```

Locals

- Locals can be used for constants or simplifying complex things that you will need in various places.

```
locals {
  class = "O'Reilly"
}
```

Locals Example

- in `main.tf` (*combine broken lines*)

```
locals {
  class = "O'Reilly"
  remaining_todos = [for t in todo_todo.step_2 : t.description if t.completed == false]
}

resource "todo_todo" "remaining" {
  description = "${local.class} has asked us to complete these!
 (${join(",", local.remaining_todos)})"
  completed = false
}
```

Remaining Todo Outputs

- in `outputs.tf`

```
output "remaining_todo_details" {
  value = todo_todo.remaining
}
```

Apply

```
$ terraform apply
```

The Terraform Console

- https://developer.hashicorp.com/terraform/tutorials/cli/console

```
$ terraform console

> local.remaining_todos
"https://github.com/spkane/testing-terraform-modules-0: testing-modules todo,
https://github.com/spkane/testing-terraform-modules-1: testing-modules todo,
https://github.com/spkane/testing-terraform-modules-2: testing-modules todo"

> join(" !!!! ", local.remaining_todos)
"https://github.com/spkane/testing-terraform-modules-0: testing-modules todo
 !!!! https://github.com/spkane/testing-terraform-modules-1: testing-modules todo
 !!!! https://github.com/spkane/testing-terraform-modules-2: testing-modules todo"
```

More Terraform Console

```
> var.repo_count
3

> [for s in module.simple_github_repo[*].github_repository.html_url : upper(s)]
[
  "HTTPS://GITHUB.COM/SPKANE/TESTING-TERRAFORM-MODULES-0",
  "HTTPS://GITHUB.COM/SPKANE/TESTING-TERRAFORM-MODULES-1",
  "HTTPS://GITHUB.COM/SPKANE/TESTING-TERRAFORM-MODULES-2",
]

> exit
```

The Meta-Arguments

- `count`
- `for_each`
- `depends_on`
- `lifecycle`
- `provider`

Meta-Argument: `depends_on`

- in `main.tf`

```
resource "todo_todo" "create_first" {
  description = "An important todo."
  completed = false
}

resource "todo_todo" "create_second" {
  description = "Forced dependency"
  completed = false
  depends_on =[ todo_todo.create_first ]
}

resource "todo_todo" "create_third" {
  description = "Automatic dependency"
  completed = todo_todo.create_second.completed
}
```

`depends_on` Apply

```
…
todo_todo.create_first: Creating…
todo_todo.create_first: Creation complete after 0s
todo_todo.create_second: Creating…
todo_todo.create_second: Creation complete after 0s
todo_todo.create_third: Creating…
todo_todo.create_third: Creation complete after 0s
…
```

Meta-Argument: `lifecycle`

- in `main.tf`

```
resource "todo_todo" "create_second" {
  description = "Forced dependency"
  completed = true
  depends_on =[ todo_todo.create_first ]
}

resource "todo_todo" "create_third" {
  description = "Automatic dependency"
  completed = todo_todo.create_second.completed
  lifecycle {
    ignore_changes = [
      completed,
    ]
  }
}
```

`lifecycle` Apply

```
…
Terraform will perform the following actions:

  # todo_todo.create_second will be updated in-place
  ~ resource "todo_todo" "create_second" {
      ~ completed   = false -> true
        id          = 6
        # (1 unchanged attribute hidden)
    }

Plan: 0 to add, 1 to change, 0 to destroy.
…
```

- Note that `todo_todo.create_third` was not updated, despite the fact that the value for `todo_todo.create_second.completed` changed.

Meta-Argument: `provider`

- Example: Adding Providers

```
provider "github" {
  alias = "personal"
  token = var.personal_github_token
}

provider "github" {
  alias  = "org"
  token  = var.org_github_token
}
```

Using Provider Aliases

```
resource "github_repository" "personal" {
  provider    = github.personal
  name        = "my-personal-repo"
…
}

resource "github_repository" "org" {
  provider    = github.org
  name        = "my-org-repo"
…
}
```

Provisioners

- https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax
  - Last Resort. Avoid in most cases.
- `file`
- `local-exec`
- `remote-exec`

Tear Down the Resources

```
$ terraform destroy
$ cd ../..
```

- **NOTE**: You should do this now, BEFORE we tear-down the AWS infrastructure in the next slide.

Tear Down the Todo Server

```
$ cd terraform-infrastructure-aws
$ terraform destroy
$ cd ..
```

- Note: Students **CAN NOT** run `terraform` in the `terraform-infrastructure-aws` directory. This is for the instructor only.

What We Have Learned

- How to use real world providers, like GitHub.
- How to use real world modules.
- How to use the various meta-arguments like `for_each`.
- How to use various functions like `tolist()` and `join()`.
- How to use various expressions like `for` and `[*]`.
- How to fix state with `state mv` or the `moved{}` block.
- How to validate variables and auto consume variables.
- and more...

Additional Reading

Terraform: Up & Running
Terraform Documentation

Additional Learning Resources

https://learning.oreilly.com/

Student Survey

Please take a moment to fill out the class survey linked to from the bottom of the ON24 audience screen.

O’Reilly and I value your comments about the class.

Thank you!

Any Questions?

Sean P. Kane

Hands-on technical training and engineering

https://techlabs.sh/