Lab: Terraform Básico

Todas as interações com o Terraform ocorrem por meio da CLI. O Terraform é uma ferramenta local (executada na máquina atual). O ecossistema terraform também inclui provedores para muitos serviços em nuvem e um repositório de módulos. A Hashicorp também possui produtos para ajudar as equipes a gerenciar o Terraform: Terraform Cloud e Terraform Enterprise.

Alsguns comandos terraform básicos, são::

- `terraform init`
- `terraform validate`
- `terraform plan`
- `terraform apply`
- `terraform destroy`

Esses comandos compõem o fluxo de trabalho do terraform que abordaremos neste curso. Será benéfico para nós explorar alguns comandos básicos agora para que trabalhem lado a lado e implantem nossas configurações.

- Tarefa 1: Verifique a instalação e a versão do Terraform
- Tarefa 2: inicializar o diretório de trabalho do Terraform: `terraform init`
- Tarefa 3: Validando uma Configuração: `terraform validate`
- Tarefa 4: Gerando um Plano Terraform: `terraform plan`
- Tarefa 5: Aplicando um Plano Terraform: `terraform apply`
- Tarefa 6: Destruição de Terraform: `terraform destroy`

Tarefa 1: verifique a instalação e a versão do Terraform

Você pode obter a versão do Terraform rodando em sua máquina com o seguinte comando:

```
terraform -version
```

Se você precisar recuperar um subcomando específico, poderá obter uma lista de comandos e argumentos disponíveis com o argumento de ajuda.

```
terraform -help
```

Tarefa 2: Terraform Init

Inicializar seu espaço de trabalho é usado para inicializar um diretório de trabalho contendo arquivos de configuração do Terraform.

Copie o trecho de código abaixo no arquivo chamado **main.tf** Este trecho aproveita o provedor aleatório, mantido pela HashiCorp, para gerar uma string aleatória.

`main.tf`

```
resource "random_string" "random" {
  length = 16
}
```

Uma vez salvo, você pode retornar ao seu shell e executar o comando init mostrado abaixo. Isso diz ao Terraform para escanear seu código e baixar qualquer coisa que precise localmente.

```
terraform init
```

Depois que seu duretório de trabalho do Terraform for inicializado, você estará pronto para começar a planejar e provisionar seus recursos.

> Nota: Você pode validar se seu diretório de trabalho foi inicializado procurando a presença de um diretório **.terraform** Este é um diretório oculto, que o Terraform usa para gerenciar plug-ins e módulos do provedor em cache, registrar qual diretório de trabalho está ativo no momento e registrar a última configuração de back-end conhecida caso precise migrar o estado. Este diretório é gerenciado automaticamente pelo Terraform e é criado durante a inicialização.

Tarefa 3: Validando a Configuração

O comando **terraform validate** valida os arquivos de configuração em seu diretório de trabalho. Para validar que não há problemas de sintaxe com nosso arquivo de configuração do terraform, execute:

```
terraform validate
```

```
Success! The configuration is valid.
```

Tarefa 5: Criando um plano Terraform

O Terraform possui um modo de teste em que você pode visualizar o que o Terraform mudará sem fazer nenhuma alteração real em sua infraestrutura. Este teste é executado executando `terraform plan`.

Em seu terminal, você pode executar um plano conforme mostrado abaixo para ver as alterações necessárias para o Terraform atingir o estado desejado definido em seu código. Isso é equivalente a executar o Terraform no modo “dry”.

```
terraform plan
```

Se você revisar a saída, verá que 1 alteração será feita, que é gerar uma única string aleatória.

```
Terraform will perform the following actions:

  # random_string.random will be created
  + resource "random_string" "random" {
      + id          = (known after apply)
      + length      = 16
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + result      = (known after apply)
      + special     = true
      + upper       = true
    }

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

> Note: O Terraform também tem o conceito de planejar alterações em um arquivo. Isso é útil para garantir que você aplique apenas o que foi planejado anteriormente. Tente executar um plano novamente, mas desta vez passando um sinalizador -out conforme mostrado abaixo.

```
terraform plan -out myplan
```

Isso criará um arquivo de plano que o Terraform pode usar durante um `apply`.

Tarefa 6: Aplicabdo um plano Terraform

Execute o comando abaixo para criar os recursos em seu arquivo de plano.

```
terraform apply myplan
```

Depois de concluído, você verá que o Terraform construiu com sucesso seu recurso de string aleatória com base no que estava em seu arquivo de plano.

O Terraform também pode executar um **apply** sem um arquivo de plano. Para experimentá-lo, modifique seu arquivo main.tf para criar uma string aleatória com um comprimento de 10 em vez de 16, conforme mostrado abaixo:

```
resource "random_string" "random" {
  length = 10
}
```

E execute o commando `terraform apply`

```
terraform apply
```

Observe que agora você verá uma saída semelhante a quando executou um `terraform plan` mas agora você será perguntado se deseja prosseguir com essas alteraçõess. Para prosseguir entre `yes`.

```
Terraform used the selected providers to generate the following execution
    plan. Resource actions are indicated with the
following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # random_string.random must be replaced
-/+ resource "random_string" "random" {
      ~ id          = "XW>5m{w8Ig96d1A&" -> (known after apply)
      ~ length      = 16 -> 10 # forces replacement
      ~ result      = "XW>5m{w8Ig96d1A&" -> (known after apply)
        # (8 unchanged attributes hidden)
    }

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

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value:
```

Depois de concluído, o recurso de string aleatória será criado com os atributos especificados no arquivo de configuração main.tf.

Tarefa 7: Terraform Destroy

O commando `terraform destroy` é uma maneira conveniente de destruir todos os objetos remotos gerenciados por uma configuração específica do Terraform. Ele não exclui seu(s) arquivo(s) de configuração, `main.tf`, etc. Destrói os recursos construídos a partir do seu código Terraform.

Execute o comando conforme mostrado abaixo para executar uma destruição planejada:

```
terraform plan -destroy
```

```
Terraform will perform the following actions:

  # random_string.random will be destroyed
  - resource "random_string" "random" {
      - id          = "1HIQs)moC0" -> null
      - length      = 10 -> null
      - lower       = true -> null
      - min_lower   = 0 -> null
      - min_numeric = 0 -> null
      - min_special = 0 -> null
      - min_upper   = 0 -> null
      - number      = true -> null
      - result      = "1HIQs)moC0" -> null
      - special     = true -> null
      - upper       = true -> null
    }

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

Você notará que está planejando destruir seu recurso criado anteriormente. Para realmente destruir a string aleatória que você criou, você pode executar um comando destroy conforme mostrado abaixo.

```
terraform destroy
```

```
Terraform will perform the following actions:

  # random_string.random will be destroyed
  - resource "random_string" "random" {
      - id          = "1HIQs)moC0" -> null
      - length      = 10 -> null
      - lower       = true -> null
      - min_lower   = 0 -> null
      - min_numeric = 0 -> null
      - min_special = 0 -> null
      - min_upper   = 0 -> null
      - number      = true -> null
      - result      = "1HIQs)moC0" -> null
      - special     = true -> null
      - upper       = true -> null
    }

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

Do you really want to destroy all resources?
```

```
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value:
```

> Note: Assim como quando você executou uma **apply**, você será solicitado a prosseguir com a destruição digitando “yes”.