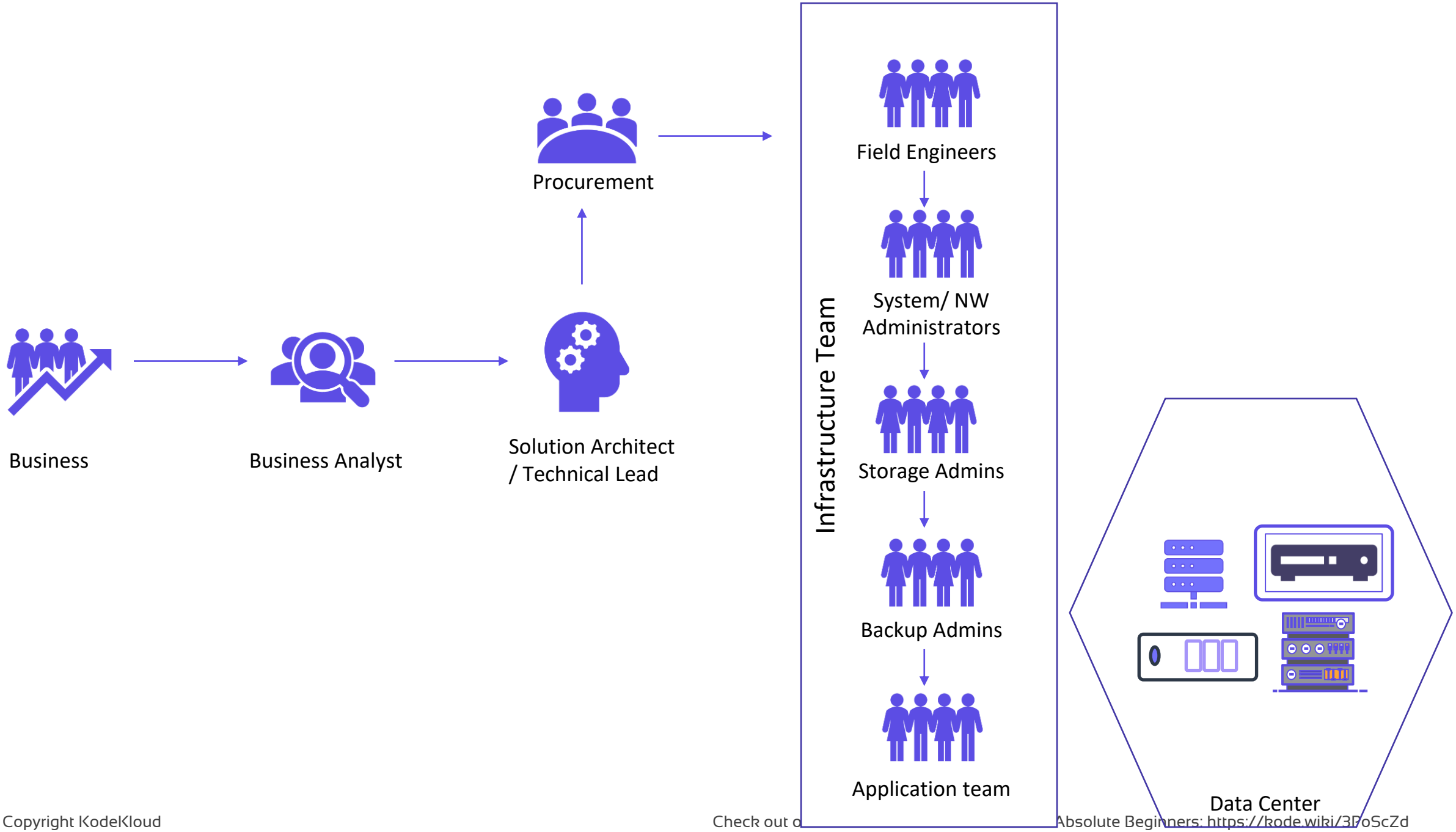
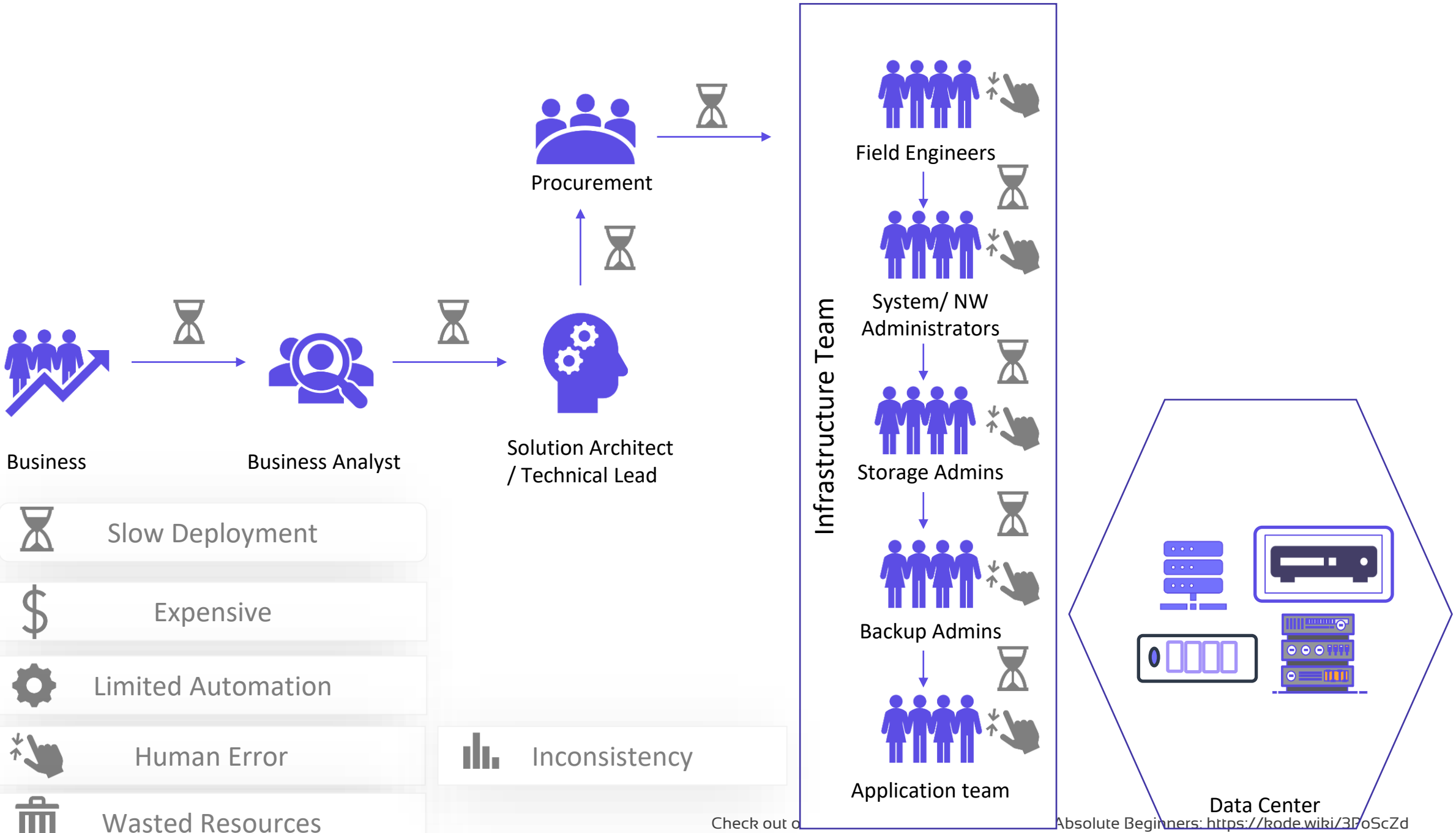
Traditional IT & Challenges





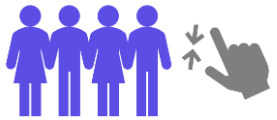
Business

Business Analyst

Solution Architect
/ Technical Lead

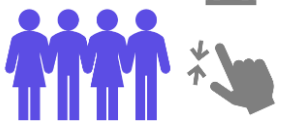
Procurement

Infrastructure Team

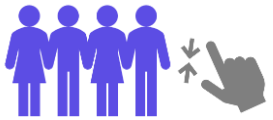


Field Engineers

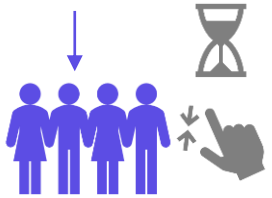
System/ NW
Administrators



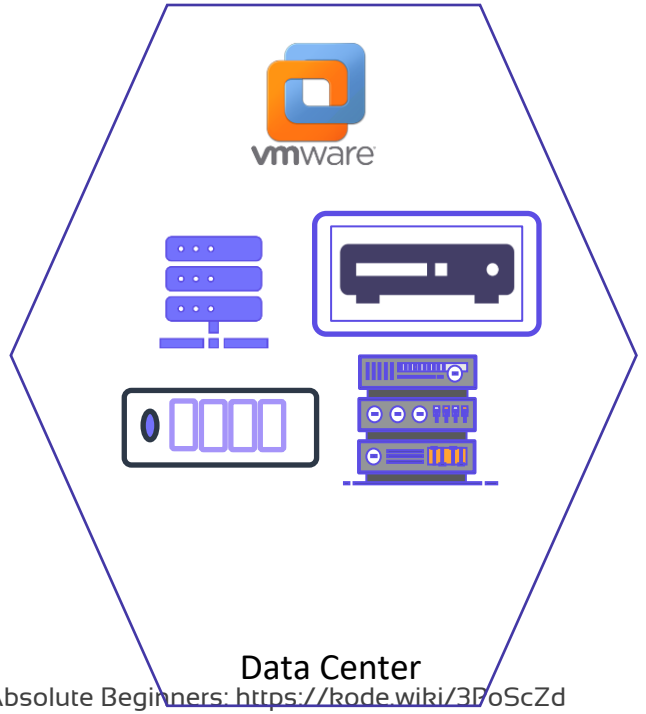
Storage Admins



Backup Admins

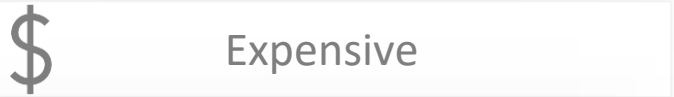


Application team



Data Center

Slow Deployment



Expensive

Limited Automation

Human Error

Inconsistency

Wasted Resources

1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Add Tags | 6. Configure Security Group | 7. Review

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

AMI Details

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0b1e2eeb33ce3d66f

Free tier eligible

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extra

Root Device Type: ebs Virtualization type: hvm

Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Security Groups

Security group name launch-wizard-1
Description launch-wizard-1 created 2020-07-09T15:48:36.426-04:00

Type	Protocol	Port Range	Source	Description

This security group has no rules

Instance Details

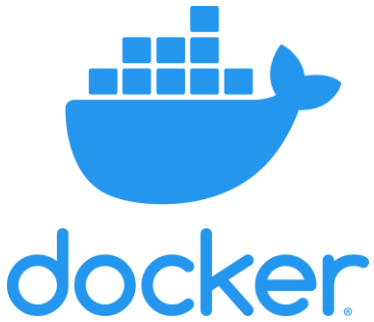
Number of instances 1
Network vpc-fe3baa86
Subnet No preference (default subnet in any Availability Zone)

Purchasing option On demand

Shell Python Ruby

Perl Powershell

Infrastructure as Code



Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

KodeKloud

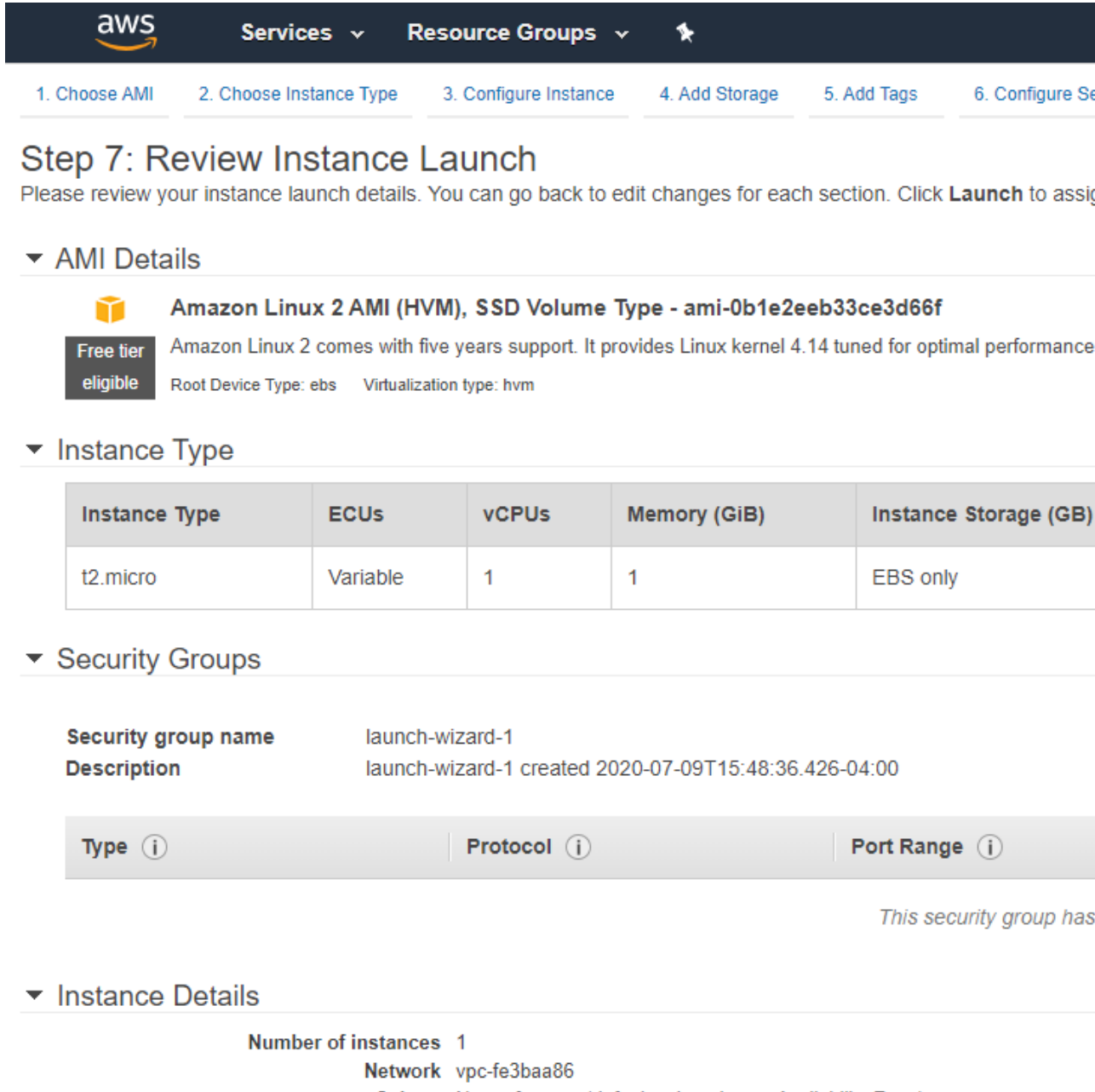
Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Infrastructure as Code

Infrastructure as Code

ec2.sh

```
#!/bin/bash

IP_ADDRESS="10.2.2.1"

EC2_INSTANCE=$(ec2-run-instances --instance-type
t2.micro ami-0edab43b6fa892279)

INSTANCE=$(echo ${EC2_INSTANCE} | sed 's/*INSTANCE //'
| sed 's/ .*//')

# Wait for instance to be ready
while ! ec2-describe-instances $INSTANCE | grep -q
"running"
do
  echo Waiting for $INSTANCE is to be ready...
done

# Check if instance is not provisioned and exit
if [ ! $(ec2-describe-instances $INSTANCE | grep -q
"running") ]; then
  echo Instance $INSTANCE is stopped.
  exit
fi

ec2-associate-address $IP_ADDRESS -i $INSTANCE

echo Instance $INSTANCE was created successfully!!!
```



Infrastructure as Code

ec2.sh

```
#!/bin/bash

IP_ADDRESS="10.2.2.1"

EC2_INSTANCE=$(ec2-run-instances --instance-type
t2.micro ami-0edab43b6fa892279)

INSTANCE=$(echo ${EC2_INSTANCE} | sed 's/*INSTANCE //'
| sed 's/ .*//')

# Wait for instance to be ready
while ! ec2-describe-instances $INSTANCE | grep -q
"running"
do
  echo Waiting for $INSTANCE is to be ready...
done

# Check if instance is not provisioned and exit
if [ ! $(ec2-describe-instances $INSTANCE | grep -q
"running") ]; then
  echo Instance $INSTANCE is stopped.
  exit
fi

ec2-associate-address $IP_ADDRESS -i $INSTANCE

echo Instance $INSTANCE was created successfully!!!
```

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}
```

Infrastructure as Code

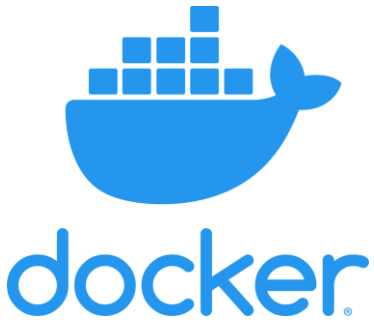
ec2.yaml

```yaml
- amazon.aws.ec2:
    key_name: mykey
    instance_type: t2.micro
    image: ami-123456
    wait: yes
    group: webserver
    count: 3
    vpc_subnet_id: subnet-29e63245
    assign_public_ip: yes
```

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}
```

Types of IAC Tools



Types of IAC Tools

Configuration Management

Server Templating

Provisioning Tools

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Types of IAC Tools

Configuration Management

Designed to Install and Manage Software

Maintains Standard Structure

Version Control

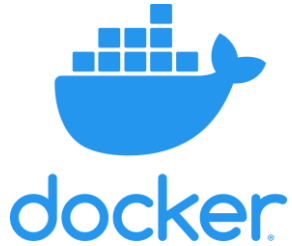
Idempotent

Server Templating Tools

Pre Installed Software and Dependencies

Virtual Machine or Docker Images

Immutable Infrastructure



Provisioning Tools

Deploy Immutable Infrastructure resources

Servers, Databases, Network Components etc.

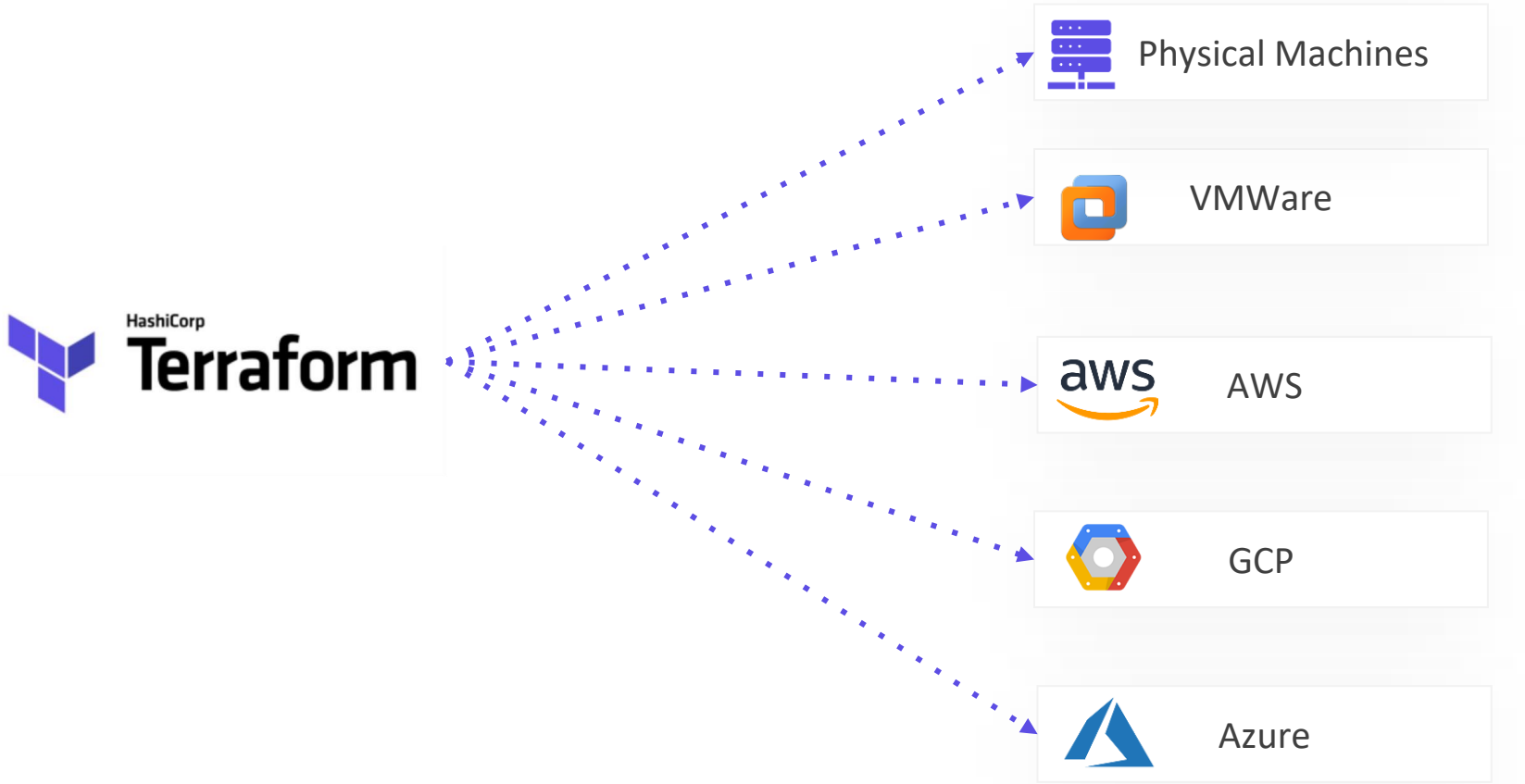
Multiple Providers

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

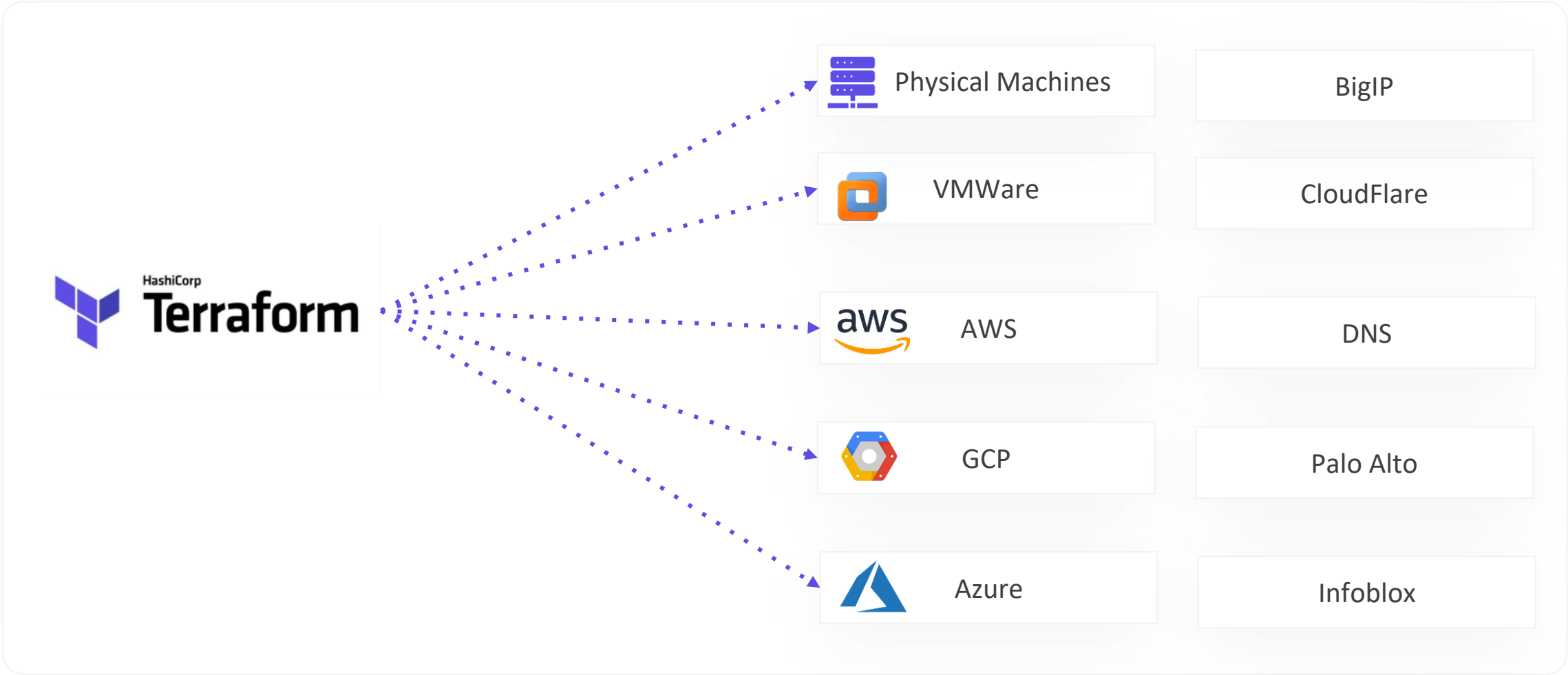
Terraform

Why Terraform?

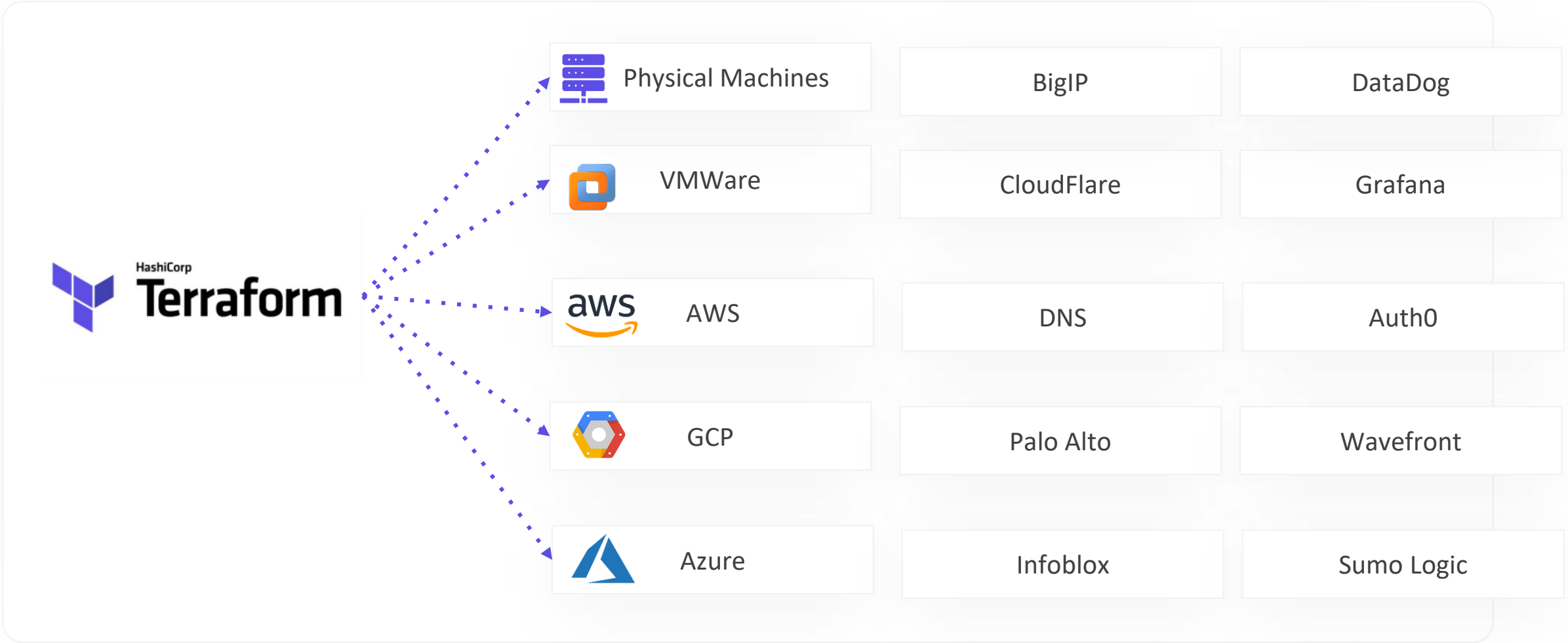


Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

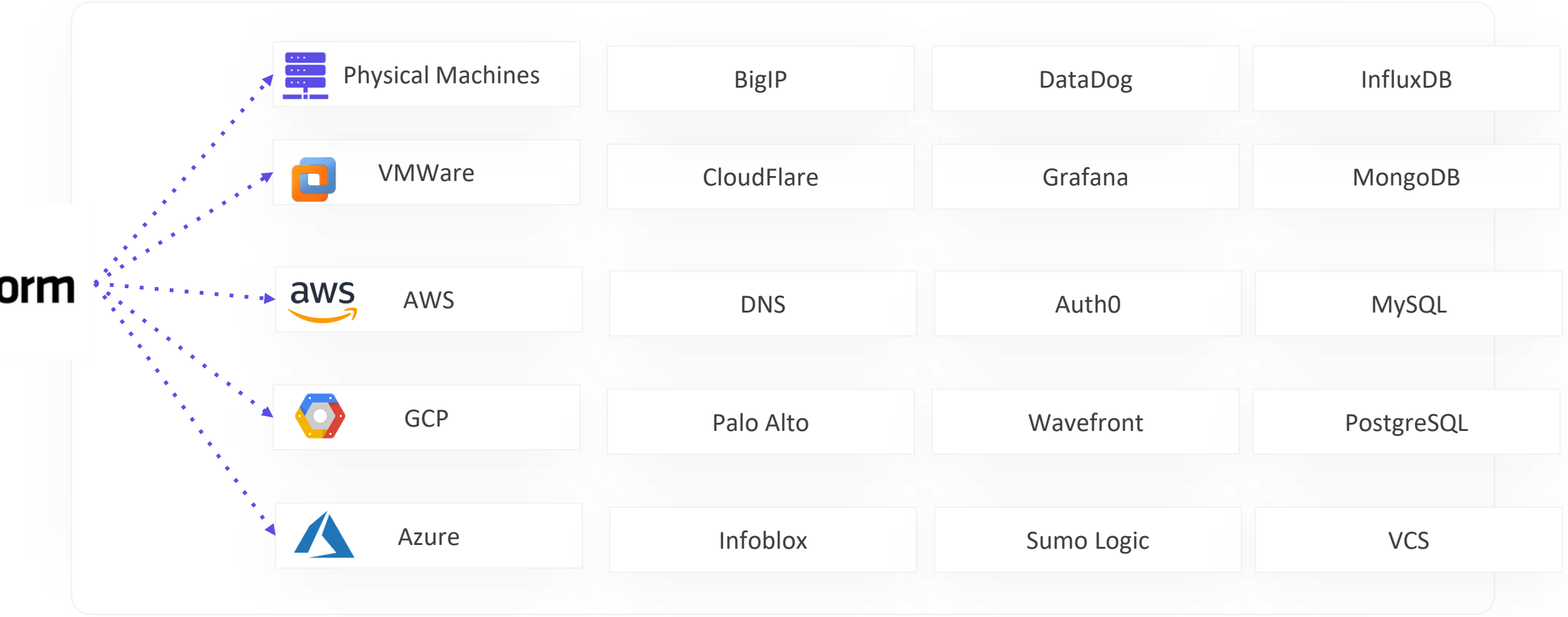
Providers



Providers



Providers



HashiCorp Configuration Language

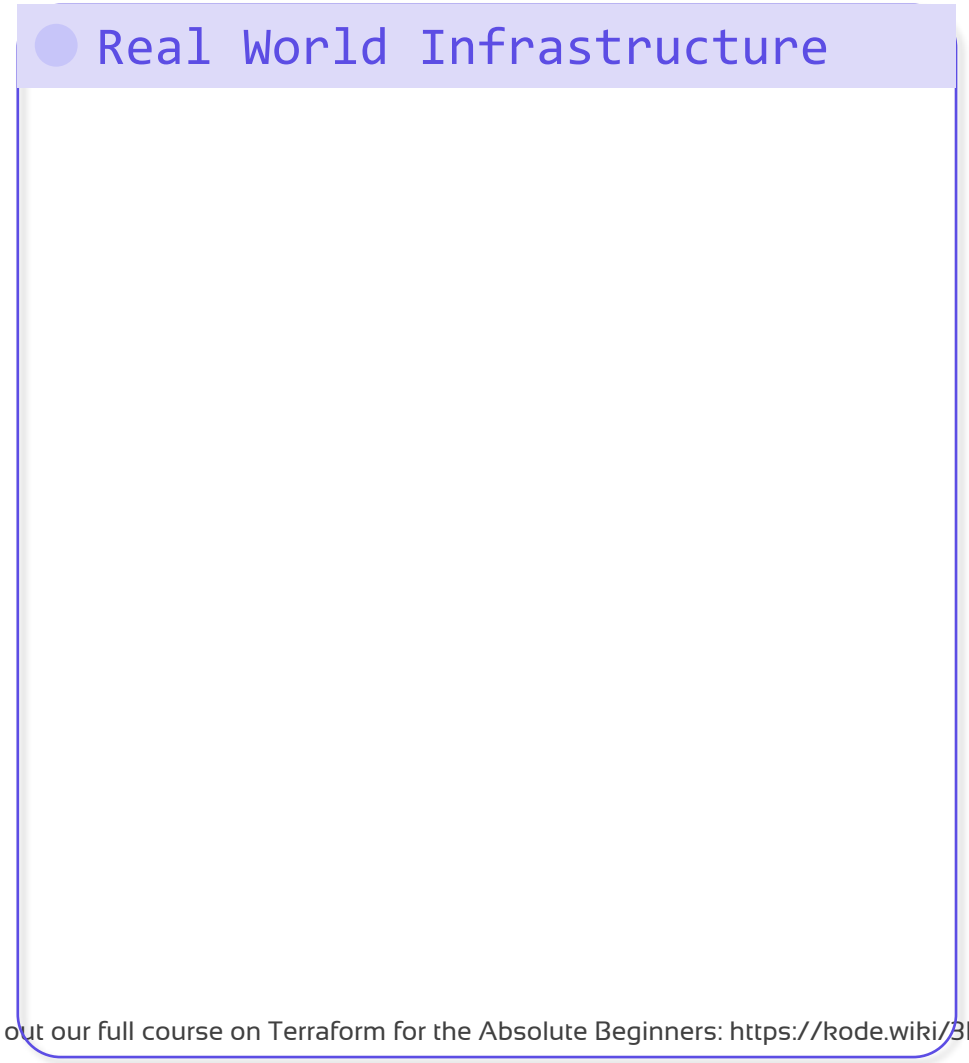
main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}

resource "aws_s3_bucket" "finance" {
    bucket = "finanace-21092020"
    tags   = {
        Description = "Finance and Payroll"
    }
}
 resource "aws_iam_user" "admin-user" {
     name = "lucy"
     tags = {
       Description = "Team Leader"
     }
 }
```

Declarative

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}

resource "aws_s3_bucket" "finance" {
    bucket = "finanace-21092020"
    tags   = {
        Description = "Finance and Payroll"
    }
}
 resource "aws_iam_user" "admin-user" {
     name = "lucy"
     tags = {
       Description = "Team Leader"
     }
 }
```



Declarative

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}

resource "aws_s3_bucket" "finance" {
    bucket = "finanace-21092020"
    tags   = {
        Description = "Finance and Payroll"
    }
}
 resource "aws_iam_user" "admin-user" {
     name = "lucy"
     tags = {
       Description = "Team Leader"
     }
 }
```

Init

Plan

Apply

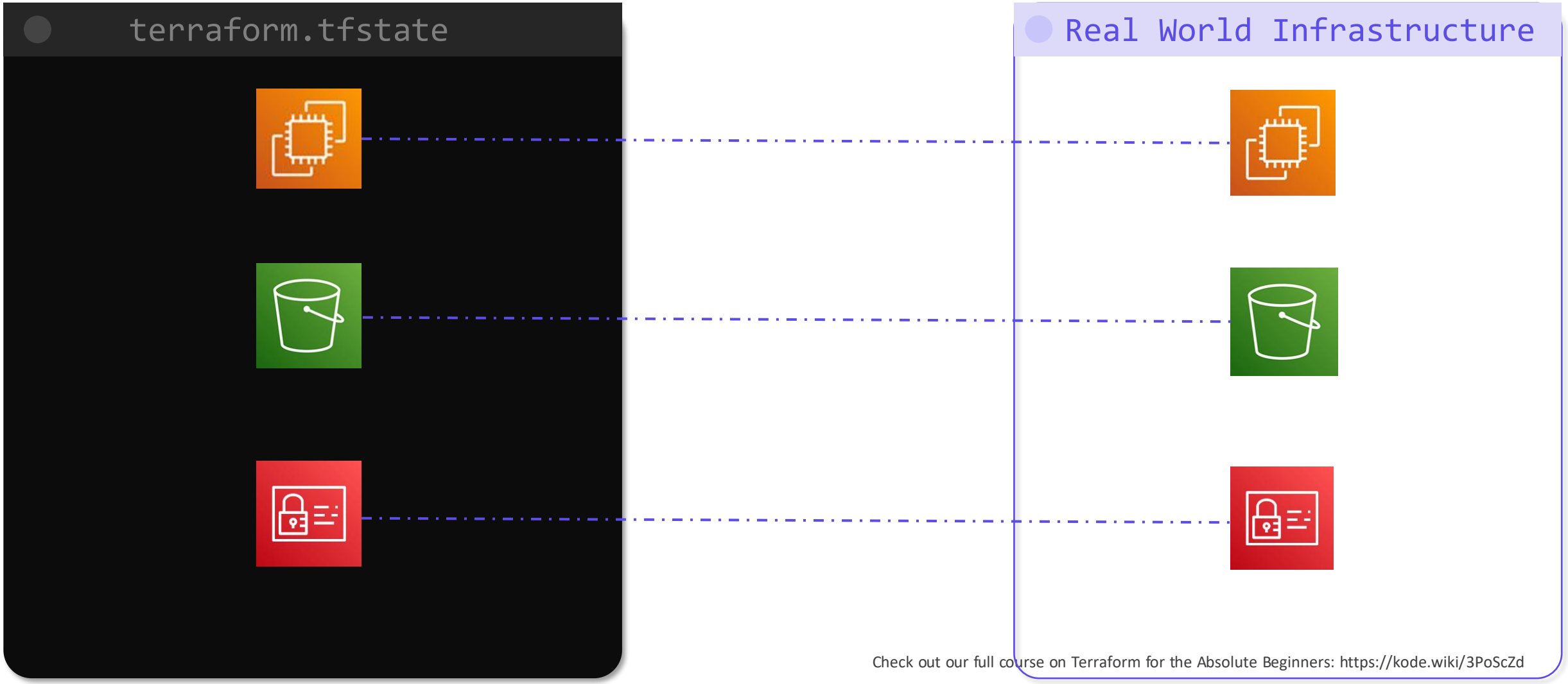
Real World Infrastructure

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Resource

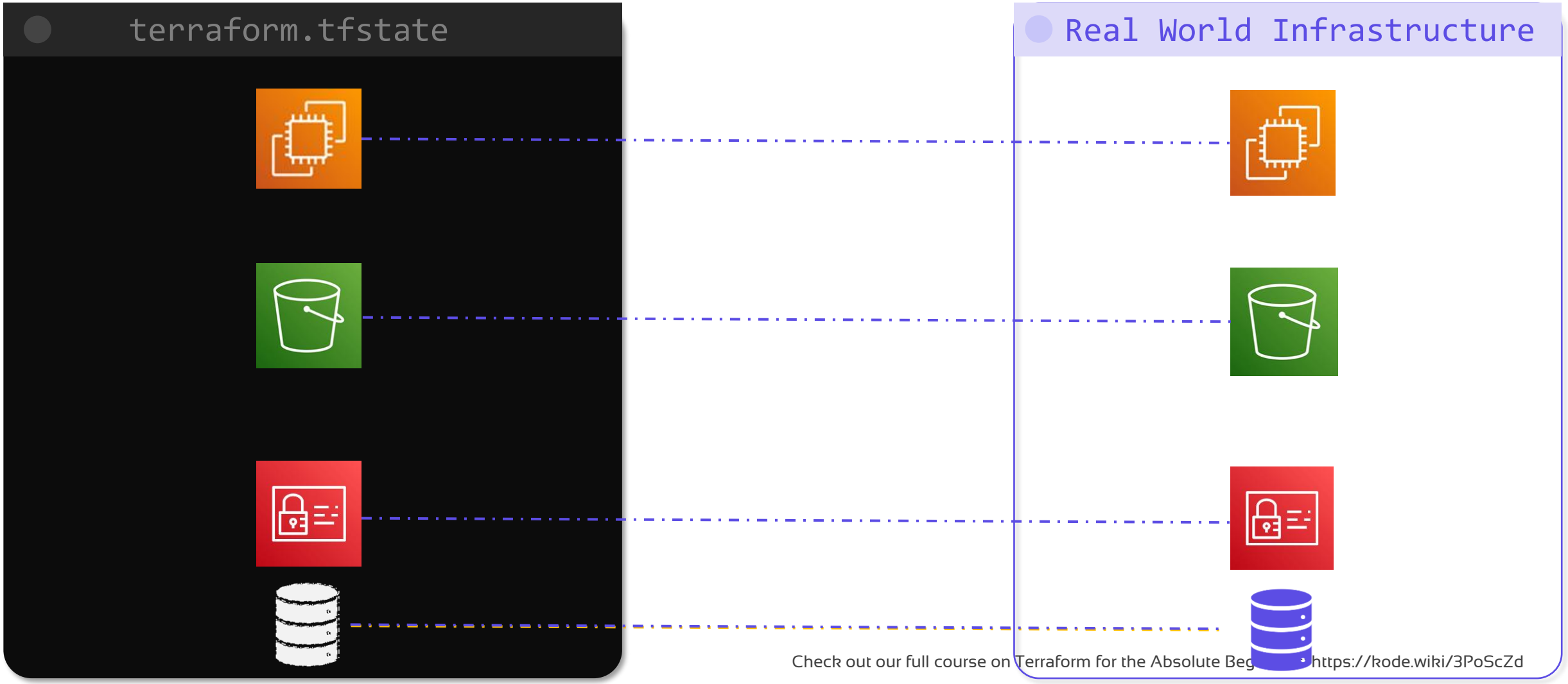
Real World Infrastructure

Terraform State

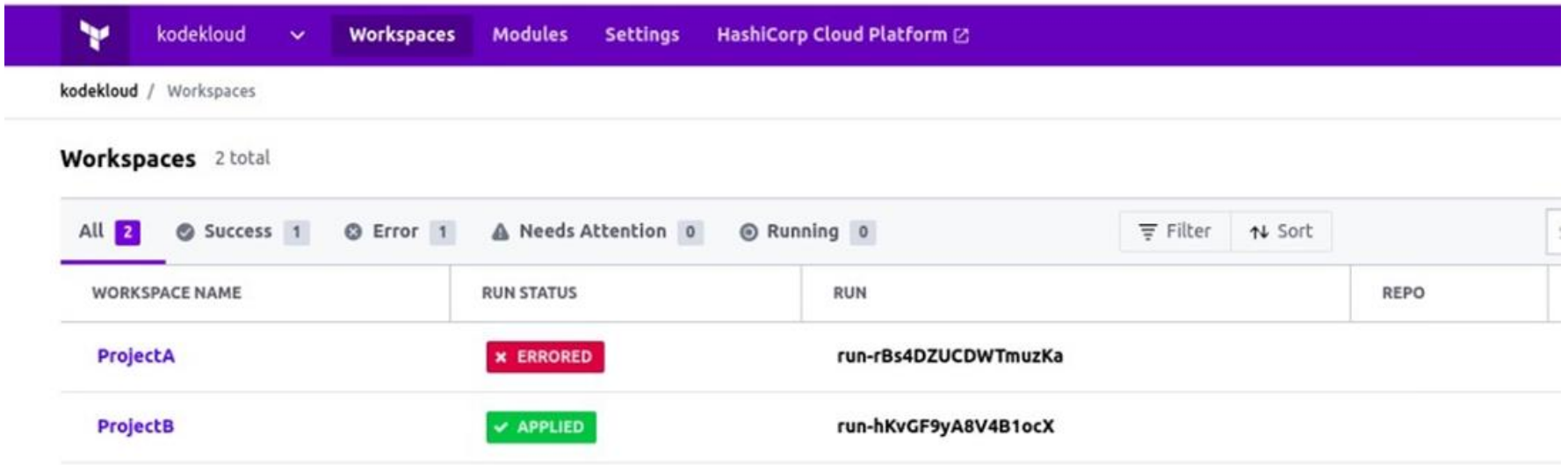


Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Terraform Import



Terraform Cloud and Terraform Enterprise

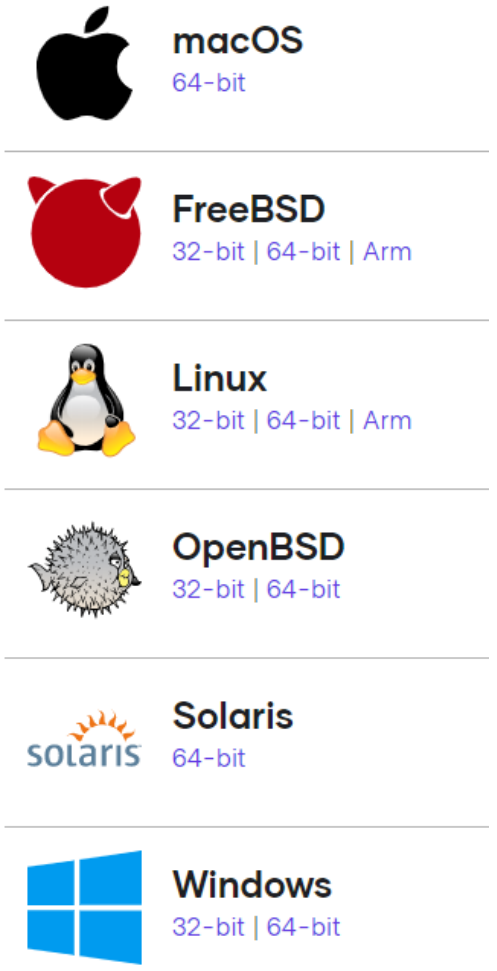


KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Installing Terraform

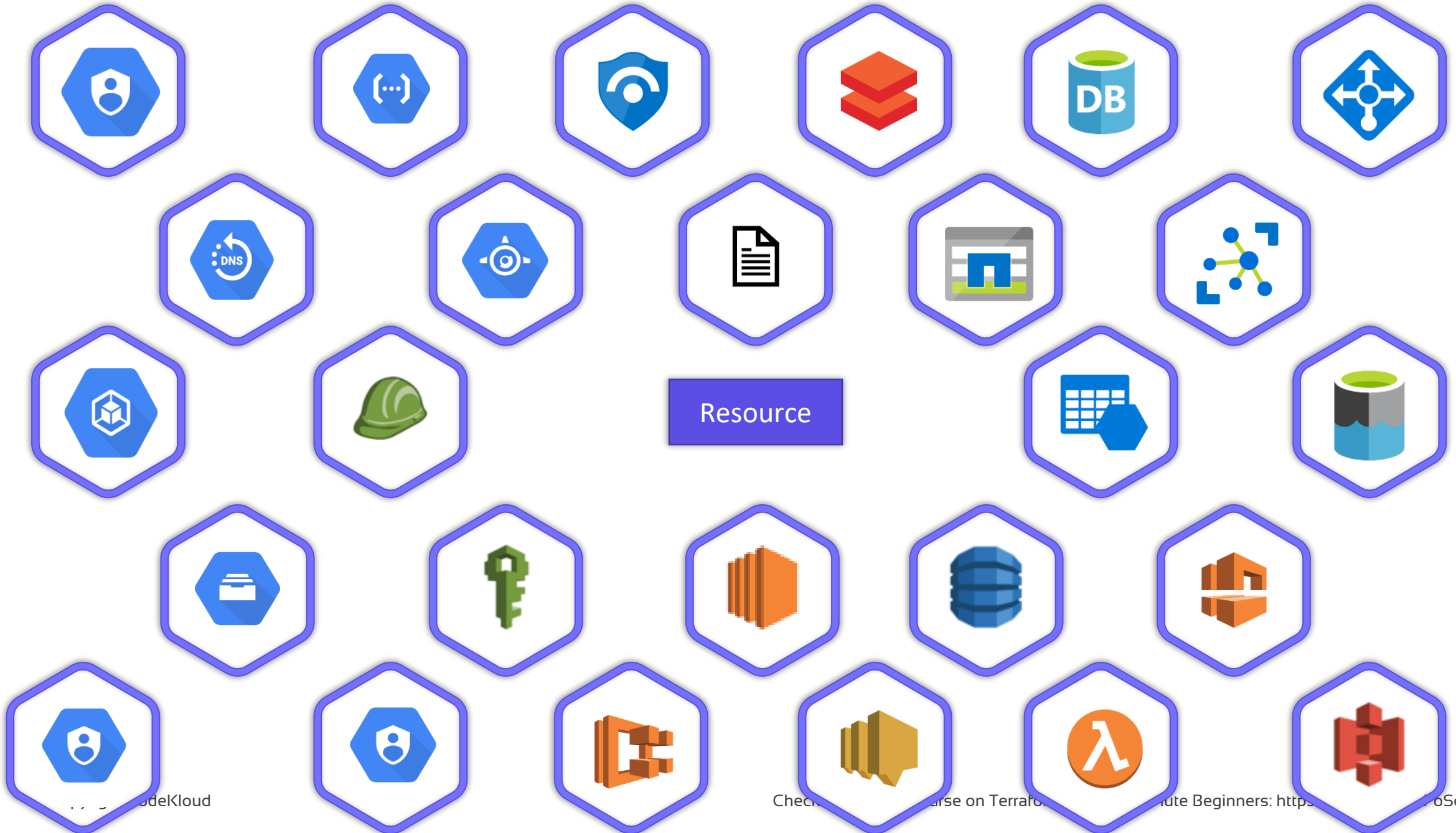
```
>_

$ wget https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
$ unzip terraform_0.13.0_linux_amd64.zip
$ mv terraform /usr/local/bin
$ terraform version
  Terraform v0.13.0
```



HCL – Declarative Language

aws.tf

```
resource "aws_instance" "webserver" {
    ami = "ami-0c2f25c1f66a1ff4d"
    instance_type = "t2.micro"
}
```


DB
DNS
Resource

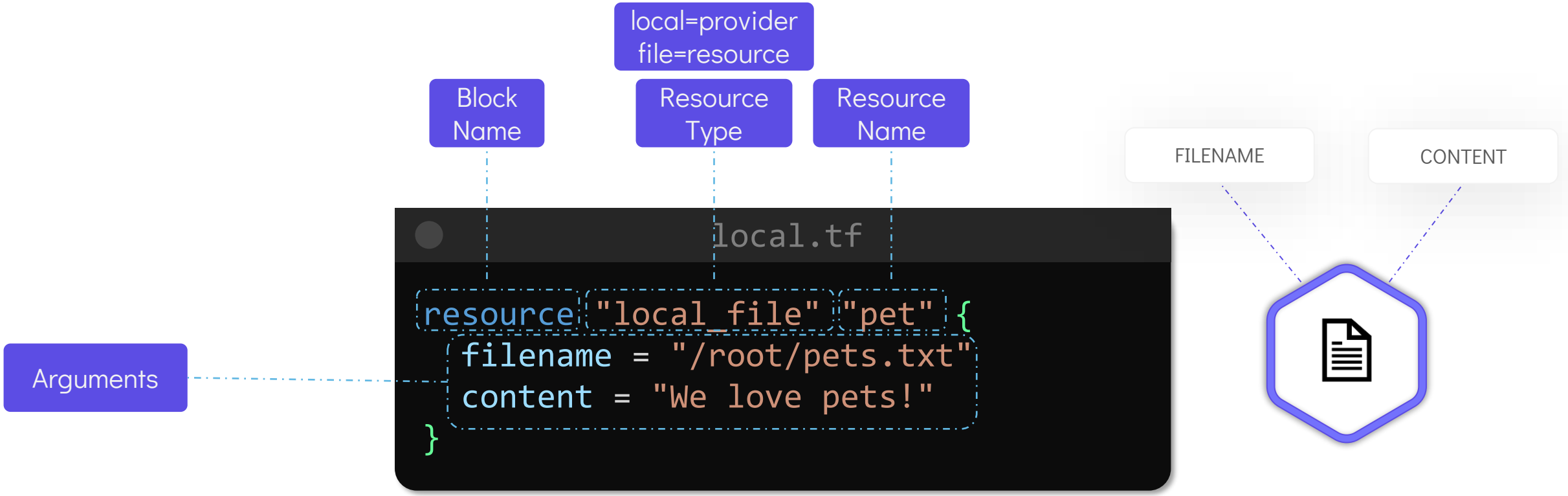
Resource

HCL Basics

```
>_
$ mkdir /root/terraform-local-file
$ cd /root/terraform-local-file
```

local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

aws-ec2.tf

```
resource "aws_instance" "webserver" {
    ami = "ami-0c2f25c1f66a1ff4d"
    instance_type = "t2.micro"
}
```

aws-s3.tf

```
resource "aws_s3_bucket" "data" {
    bucket = "webserver-bucket-org-2207"
    acl    = "private"
 }
```

local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

>_

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```

```
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

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

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so
Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

```
$ terraform apply
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 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: yes

local_file.new_file: Creating...
local_file.new_file: Creation complete after 0s
[id=521c5c732c78cb42cc9513ecc7c0638c4a115b55]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

$ cat /root/pets.txt
We love pets!
```

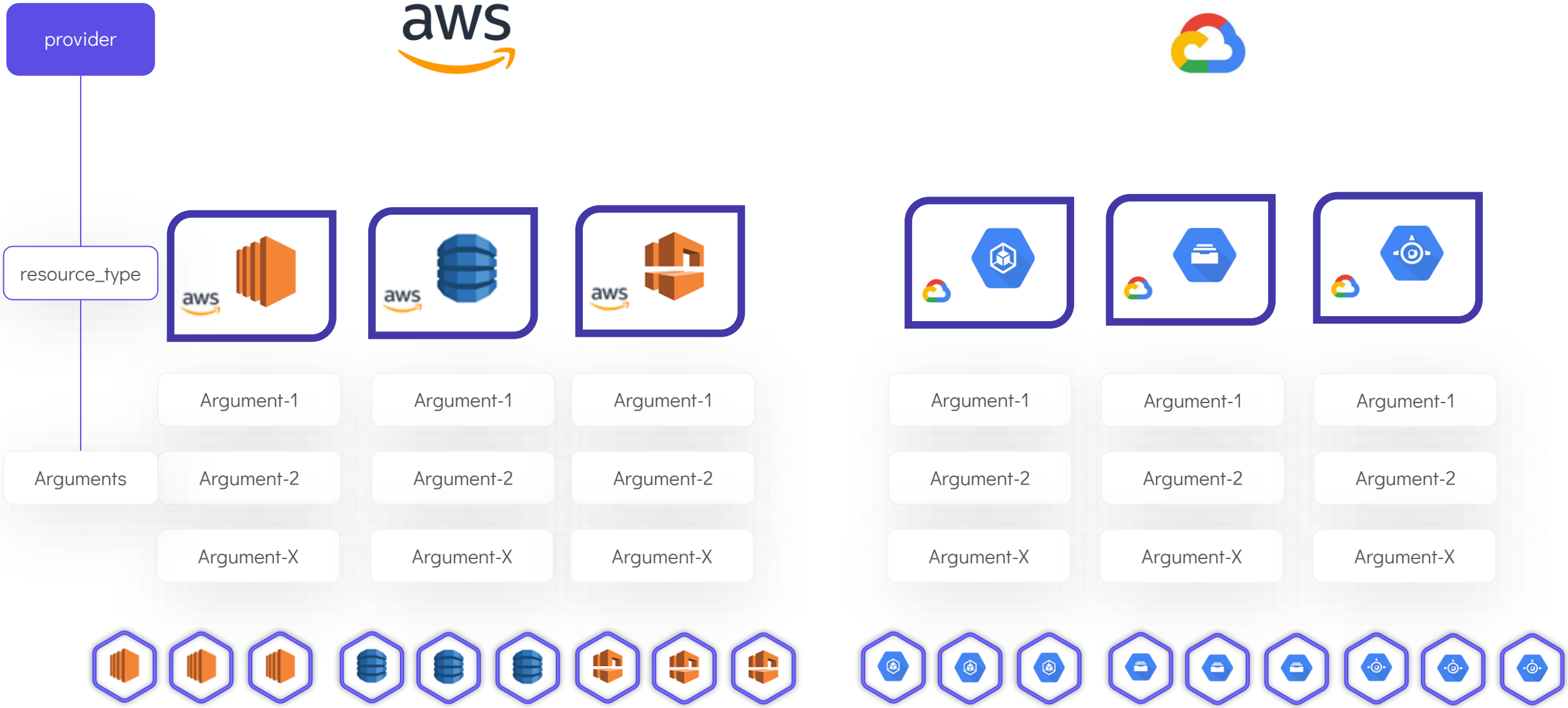
```
>_

$ terraform show
# local_file.pet:
resource "local_file" “pet" {
    content              = "We love pets!"
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
    id                   = "cba595b7d9f94ba1107a46f3f731912d95fb3d2c"
}
```

local=provider
file=resource

Resource Type

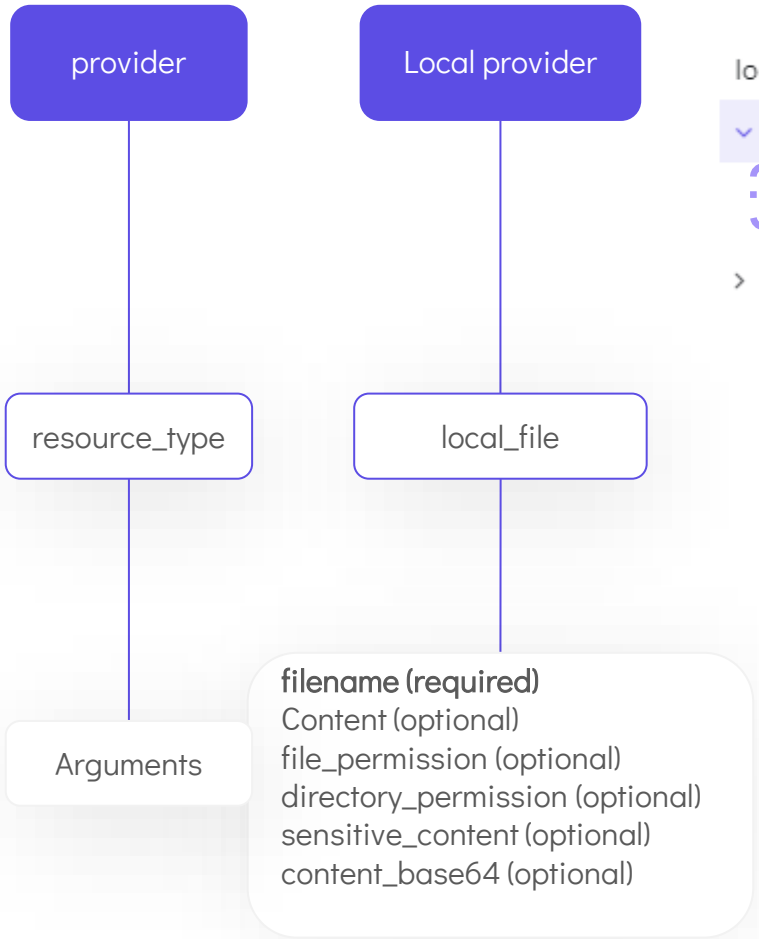
local.tf

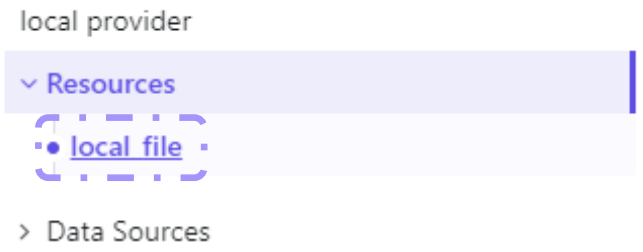
```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

https://registry.terraform.io/providers/hashicorp/local/latest/docs





Argument Reference

The following arguments are supported:

- `content` - (Optional) The content of file to create. Conflicts with `sensiti` `content_base64` .
- `sensitive_content` - (Optional) The content of file to create. Will not be c Conflicts with `content` and `content_base64` .
- `content_base64` - (Optional) The base64 encoded content of the file to c when dealing with binary data. Conflicts with `content` and `sensitive_co`
- `filename` - (Required) The path of the file to create.
- `file_permission` - (Optional) The permission to set for the created file. E string. The default value is `"0777"` .
- `directory_permission` - (Optional) The permission to set for any directori Expects a string. The default value is `"0777"` .

HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

Update and Destroy Infrastructure

local.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"

}
```

```
$ terraform plan
local_file.pet: Refreshing state...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

------------------------------------------------------------------------
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after apply)
    }

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

------------------------------------------------------------------------
Note: You didn't specify an "-out" parameter to save this plan, so
Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

```
$ ls -ltr /root/pets.txt
-rwx------  1 root root 30 Aug 17 23:20 pet.txt
```

```
$ terraform apply

# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after apply)
    }

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: yes

local_file.pet: Destroying...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

```
$ terraform destroy
local_file.pet: Refreshing state...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # local_file.pet will be destroyed
  - resource "local_file" "pet" {
      - content              = "My favorite pet is a gold fish" -> null
      - directory_permission = "0777" -> null
      - file_permission      = "0700" -> null
      - filename             = "/root/pet.txt" -> null
      - id                   = "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -
> 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: yes

local_file.pet: Destroying... [id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

Terraform Basics

Using Terraform Providers

```
>_

$ terraform init
```

Official

Verified

bigip

by: F5Networks

heroku

by: heroku

digitalocean

by: digitalocean

Community

activedirectory

ucloud

netapp-gcp

registry.terraform.io

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.0.0...
- Installed hashicorp/local v2.0.0 (signed by HashiCorp)

The following providers do not have any version constraints in
configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may
contain breaking
changes, we recommend adding version constraints in a
required_providers block
in your configuration, with the constraint strings suggested
below.

* hashicorp/local: version = "~> 2.0.0"

Terraform has been successfully initialized!
```

```
$ ls /root/terraform-local-file/.terraform
plugins
```

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.0.0...
- Installed hashicorp/local v2.0.0 (signed by HashiCorp)

The following providers do not have any version constraints in
configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may
contain breaking
changes, we recommend adding version constraints in a
required_providers block
in your configuration, with the constraint strings suggested
below.

* hashicorp/local: version = "~> 2.0.0"

Terraform has been successfully initialized!
```

To prevent automatic upgrades to new maj
contain breaking
changes, we recommend adding version con
required_providers block
in your configuration, with the constrai
below.

* hashicorp/local: version = "~> 2.0.0"

Organizational Namespace

Type

Terraform has been successfully initiali

To prevent automatic upgrades to new maj
contain breaking
changes, we recommend adding version con
required_providers block
in your configuration, with the constrai
below.

* registry.terraform.io/ hashicorp/ loc

Hostname

Organizational Namespace

Ty

Terraform has been successfully initiali

```
Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.0.0...
- Installed hashicorp/local v2.0.0 (signed by H

The following providers do not have any version
configuration,
so the latest version was installed.

To prevent automatic upgrades to new major vers
contain breaking
changes, we recommend adding version constraint
required_providers block
```

Configuration Directory

```
>_

[terraform-local-file]$ ls /root/terraform-local-file
local.tf
```

local.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}
```

cat.tf

```
resource "local_file"  "cat" {
    filename = "/root/cat.txt"
    content = "My favorite pet is Mr. Whiskers"

}
```

local.tf

cat.tf

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}

resource "local_file"  "cat" {
    filename = "/root/cat.txt"
    content = "My favorite pet is Mr. Whiskers"

}
```

File Name	Purpose
main.tf	Main configuration file containing resource definition
variables.tf	Contains variable declarations
outputs.tf	Contains outputs from resources
provider.tf	Contains Provider definition

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

Multiple Providers

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}
```

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}


resource "random_pet" "my-pet" {
    prefix = "Mrs"
    separator = "."
    length = "1"
}
```

random provider

Resources

- random_id
- random_integer
- random_password
- random_pet
- random_shuffle
- random_string
- random_uuid

Argument Reference

The following arguments are supported:

- `keepers` - (Optional) Arbitrary map of values that, w
  to be generated. See the main provider documentati
- `length` - (Optional) The length (in words) of the pet
- `prefix` - (Optional) A string to prefix the name with
- `separator` - (Optional) The character to separate wo

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Using previously-installed hashicorp/local v2.0.0
- Finding latest version of hashicorp/random...
- Installing hashicorp/random v2.3.0...
- Installed hashicorp/random v2.3.0 (signed by HashiCorp)

The following providers do not have any version constraints in
configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain
breaking
changes, we recommend adding version constraints in a required_providers
block
in your configuration, with the constraint strings suggested below.

* hashicorp/local: version = "~> 2.0.0"
* hashicorp/random: version = "~> 2.3.0"

Terraform has been successfully initialized!
```

course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

```
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but
will not be
persisted to local or remote state storage.

local_file.pet: Refreshing state...
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]

.
.
  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

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

course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

```
$ terraform apply
local_file.new_file: Refreshing state...
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

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

random_pet.my-pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mrs.hen]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Mrs.hen

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

Define Input Variables

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"

}

resource "random_pet" "my-pet" {
    prefix = "Mrs"
    separator = "."
    length = "1"
}
```

Argument	Value
filename	"/root/pets.txt"
content	"We love pets!"
prefix	"Mrs"
separator	"."
length	"1"

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"

}

resource "random_pet" "my-pet" {
    prefix = "Mrs"
    separator = "."
    length = "1"
}
```

variables.tf

```
variable "filename" {
        default = "/root/pets.txt"
}
variable "content" {
        default = "We love pets!"
}
variable "prefix" {
        default = "Mrs"
}
variable "separator" {
        default = "."
}
variable "length" {
        default = "1"
}
```

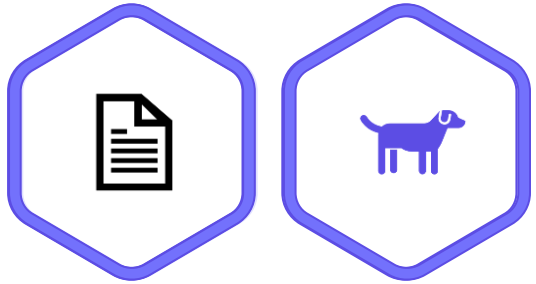
main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content = var.content

}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

variables.tf

```
variable "filename" {
        default = "/root/pets.txt"
}
variable "content" {
        default = "We love pets!"
}
variable "prefix" {
        default = "Mrs"
}
variable "separator" {
        default = "."
}
variable "length" {
        default = "1"
}
```

```
$ terraform apply

# local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = “We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pet.txt"
      + id                   = (known after apply)
    }

  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

Plan: 2 to add, 0 to change, 0 to destroy.
.
.
random_pet.my-pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mrs.ram]
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=f392b4bcf5db76684f719bf72061627a9a177de1]
```



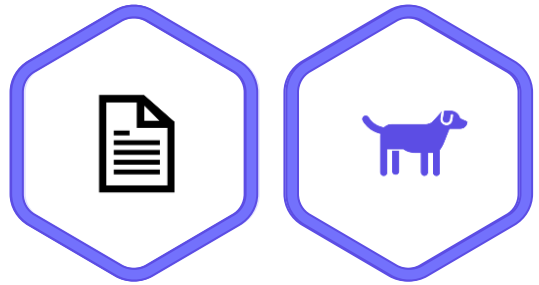
main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content = var.content

}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

variables.tf

```
variable "filename" {
    default = "/root/pets.txt"
}
variable "content" {
    default = "My favorite pet is Mrs. Whiskers"
}
variable "prefix" {
    default = "Mrs"
}
variable "separator" {
    default = "."
}
variable "length" {
    default = "2"
}
```

```
$ terraform apply
Terraform will perform the following actions:

-/+ resource "local_file" "pet" {
      ~ content              = “We love pets!" -> "My favorite pet is Mrs. Whiskers!" #
forces replacement
        directory_permission = "0777"
        file_permission      = "0777"
        filename             = "/root/pet.txt"
      ~ id                   = "bc9cabef1d8b0071d3c4ae9959a9c328f35fe697" -> (known after
apply)
    }

  # random_pet.my-pet must be replaced
-/+ resource "random_pet" "my-pet" {
      ~ id        = "Mrs.Hen" -> (known after apply)
      ~ length    = 1 -> 2 # forces replacement
        prefix    = "Mrs"
        separator = "."
    }

Plan: 2 to add, 0 to change, 2 to destroy.
random_pet.my-pet: Destroying... [id=Mrs.hen]
random_pet.my-pet: Destruction complete after 0s
local_file.pet: Destroying... [id=bc9cabef1d8b0071d3c4ae9959a9c328f35fe697]
local_file.pet: Destruction complete after 0s
random_pet.my-pet: Creating...
local_file.pet: Creating...
```



te Beginners: https://kode.wiki/3PoScZd

main.tf

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = var.instance_type
}
```

variables.tf

```
variable "ami" {
  default = "ami-0edab43b6fa892279"
}
variable "instance_type" {
  default = "t2.micro"
}
```

Understanding the Variable Block

variables.tf

```
variable "filename" {
   default = "/root/pets.txt"
}
variable "content" {
    default = "I love pets!"
}
variable "prefix" {
    default = "Mrs"
}
variable "separator" {
    default = "."
}
variable "length" {
    default = "1"
}
```

variables.tf

```
variable "filename" {
   default = "/root/pets.txt"
   type = string
   description = "the path of local file"

}
variable "content" {
    default = "I love pets!"
    type = string
    description = “the content of the file"

}
variable "prefix" {
    default = "Mrs"
    type = string
    description = "the prefix to be set"

}
variable "separator" {
    default = "."
```

variables.tf

```
variable "filename" {
   default = "/root/pets.txt"
   type = string
   description = "the path of local file"

}
variable "content" {
    default = "I love pets!"
    type = string
    description = “the content of the file"

}
variable "prefix" {
    default = "Mrs"
    type = string
    description = "the prefix to be set"

}
variable "separator" {
    default = "."
```

Type	Example
string	"/root/pets.txt"
number	1
bool	true/false
any	Default Value

variables.tf

```
variable "length" {
  default = 2
  type = number
  description = "length of the pet name"
}

variable "password_change" {
  default = true
  type = bool
}
```

Type	Example
string	"/root/pets.txt"
number	1
bool	true/false
any	Default Value
list	["cat", "dog"]
map	pet1 = cat pet2 = dog
object	Complex Data Structure
tuple	Complex Data Structure

List

variables.tf

```
variable "prefix" {
  default = ["Mr", "Mrs", "Sir"]
  type = list  0     1      2
}
```

maint.tf

```
resource "random_pet" "my-pet" {
  prefix    = var.prefix[0]
}
```

Index	Value
0	Mr
1	Mrs
2	Sir

Map

variables.tf

```
variable file-content {
  type        = map
  default     = {
      "statement1" = "We love pets!"
      "statement2" = “We love animals!"
  }
 }
```

maint.tf

```
resource local_file my-pet {
  filename  = "/root/pets.txt"
  content =  var.file-content["statement2"]
}
```

Key	Value
statement1	We love pets!
statement2	We love animals!

List of a Type

variables.tf

```
variable "prefix" {
  default = ["Mr", "Mrs", "Sir"]
  type = list(string)
}
```

variables.tf

```
variable "prefix" {
  default = ["Mr", "Mrs", "Sir"]
  type = list(number)
}
```

variables.tf

```
variable "prefix" {
  default = [1, 2, 3]
  type = list(number)
}
```

>_

```
$ terraform plan
Error: Invalid default value for variable

  on variables.tf line 3, in variable "prefix":
   3:   default     = ["Mr", "Mrs", "Sir"]

This default value is not compatible with the
variable's type constraint: a number is required.
```

Map of a Type

variables.tf

```
variable "cats" {
  default = {
    "color" = "brown"
    "name" = "bella"
  }
  type = map(string)
}
```

variables.tf

```
variable "pet_count" {
  default = {
    "dogs" = 3
    "cats" = 1
    "goldfish" = 2
  }
  type = map(number)
}
```

Set

variables.tf

```
variable "prefix" {
  default = ["Mr", "Mrs", "Sir"]
  type = set(string)
}
```

variables.tf

```
variable "prefix" {
  default = ["Mr", "Mrs", "Sir", "Sir"]
  type = set(string)
}
```

variables.tf

```
variable "fruit" {
  default = ["apple", "banana"]
  type = set(string)
}
```

variables.tf

```
variable "fruit" {
  default = ["apple", "banana", "banana"]
  type = set(string)
}
```

variables.tf

```
variable "age" {
  default = [10, 12, 15]
  type = set(number)
}
```

variables.tf

```
variable "age" {
  default = [10, 12, 15, 10]
  type = set(number)
}
```

Objects

Key	Example	Type
name	bella	string
color	brown	string
age	7	number
food	["fish", "chicken", "turkey"]	list
favorite_pet	true	bool

variables.tf

```
variable  "bella"  {
  type = object({
      name = string
      color = string
      age = number
      food = list(string)
      favorite_pet = bool
  })

  default = {
      name = "bella"
      color = "brown"
      age = 7
      food = ["fish", "chicken", "turkey"]
      favorite_pet = true
  }
}
```

Tuples

variables.tf

```
variable kitty {
  type        = tuple([string, number, bool])
  default     = ["cat", 7, true]

}
```

variables.tf

```
variable kitty {
  type        = tuple([string, number, bool])
  default     = ["cat", 7, true, "dog"]

}
```

>_

```
$ terraform plan

  Error: Invalid default value for variable

    on variables.tf line 3, in variable "kitty":
     3:   default     = ["cat", 7, true, "dog"]

  This default value is not compatible with the
  variable's type constraint:
  tuple required.
```

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

Using Variables in Terraform

main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content = var.content

}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

variables.tf

```
variable "filename" {
        default = "/root/pets.txt"
}
variable "content" {
        default = "We love pets!"
}
variable "prefix" {
        default = "Mrs"
}
variable "separator" {
        default = "."
}
variable "length" {
        default = 2
}
```

main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content = var.content

}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

variables.tf

```
variable "filename" {

}
variable "content" {

}
variable "prefix" {

}
variable "separator" {

}
variable "length" {

}
```

Interactive Mode

```
$ terraform apply
var.content
  Enter a value: We love Pets!

var.filename
  Enter a value:  /root/pets.txt

var.length
  Enter a value:  2

var.prefix
  Enter a value:  Mrs.

var.separator
  Enter a value: .
```

Command Line Flags

```
$ terraform apply -var "filename=/root/pets.txt" -var "content=We love
Pets!"  -var "prefix=Mrs" -var "separator=." -var "length=2"
```

Environment Variables

```
$ export TF_VAR_filename="/root/pets.txt"
$ export TF_VAR_content="We love pets!"
$ export TF_VAR_prefix="Mrs"
$ export TF_VAR_separator="."
$ export TF_VAR_length="2"
$ terraform apply
```

Variable Definition Files

terraform.tfvars

```
filename = "/root/pets.txt"
content = "We love pets!"
prefix = "Mrs"
separator = "."
length = "2"
```

>_

```
$ terraform apply -var-file variables.tfvars
```

Automatically Loaded

terraform.tfvars	terraform.tfvars.json
*.auto.tfvars	*.auto.tfvars.json

Variable Definition Precedence

main.tf

```
resource local_file pet {
  filename = var.filename
}
```

variables.tf

```
variable filename {
  type    = string
}
```

>_

```
$ export TF_VAR_filename="/root/cats.txt"
```

?

terraform.tfvars

```
filename = "/root/pets.txt"
```

?

variable.auto.tfvars

```
filename = "/root/mypet.txt"
```

?

>_

```
$ terraform apply -var "filename=/root/best-pet.txt"
```

?

Variable Definition Precedence

Order	Option
1	Environment Variables
2	terraform.tfvars
3	*.auto.tfvars (alphabetical order)
4	-var or –var-file (command-line flags)

```
>_
$ export TF_VAR_filename="/root/cats.txt" 1
```

```
terraform.tfvars
filename = "/root/pets.txt" 2
```

```
variable.auto.tfvars
filename = "/root/mypet.txt" 3
```

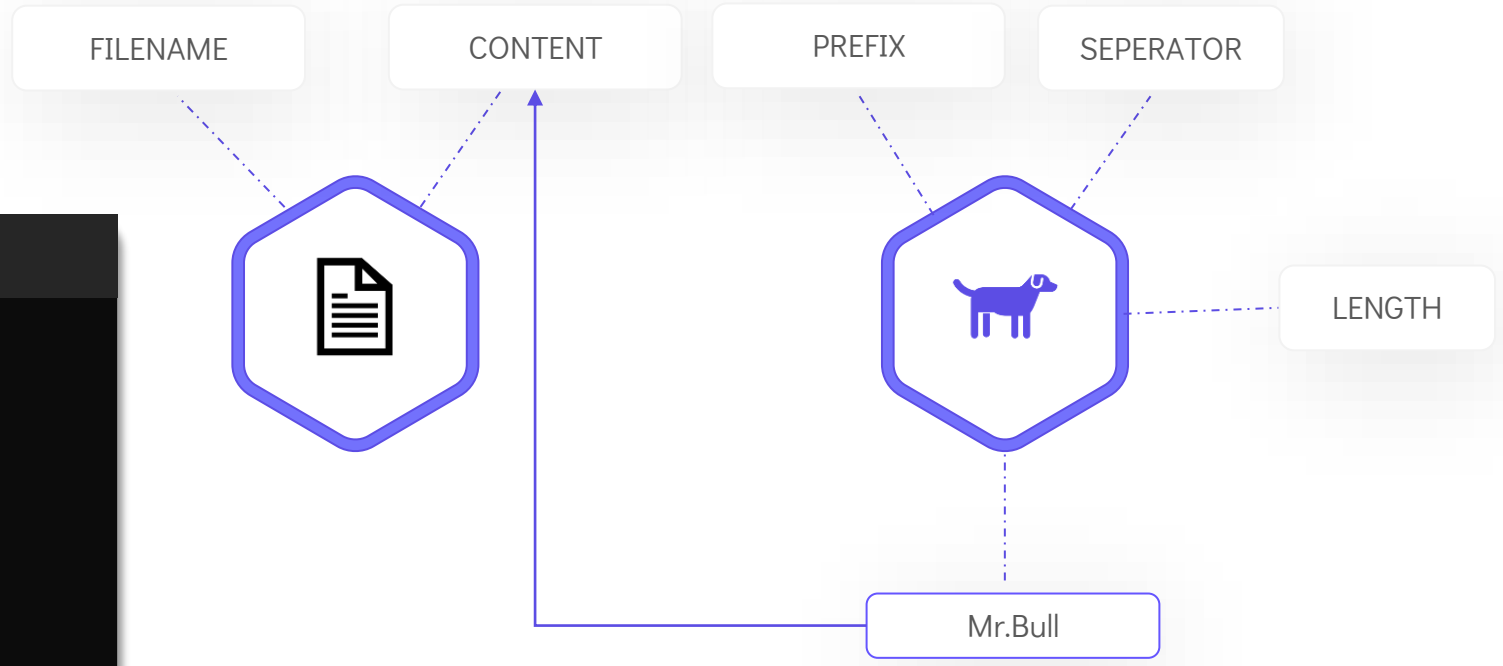
```
>_
$ terraform apply -var "filename=/root/best-pet.txt" 4
```

HANDS-ON LABS

t our full course on Terraform for the Absolute Beginners.

oScZd

Resource Attribute Reference

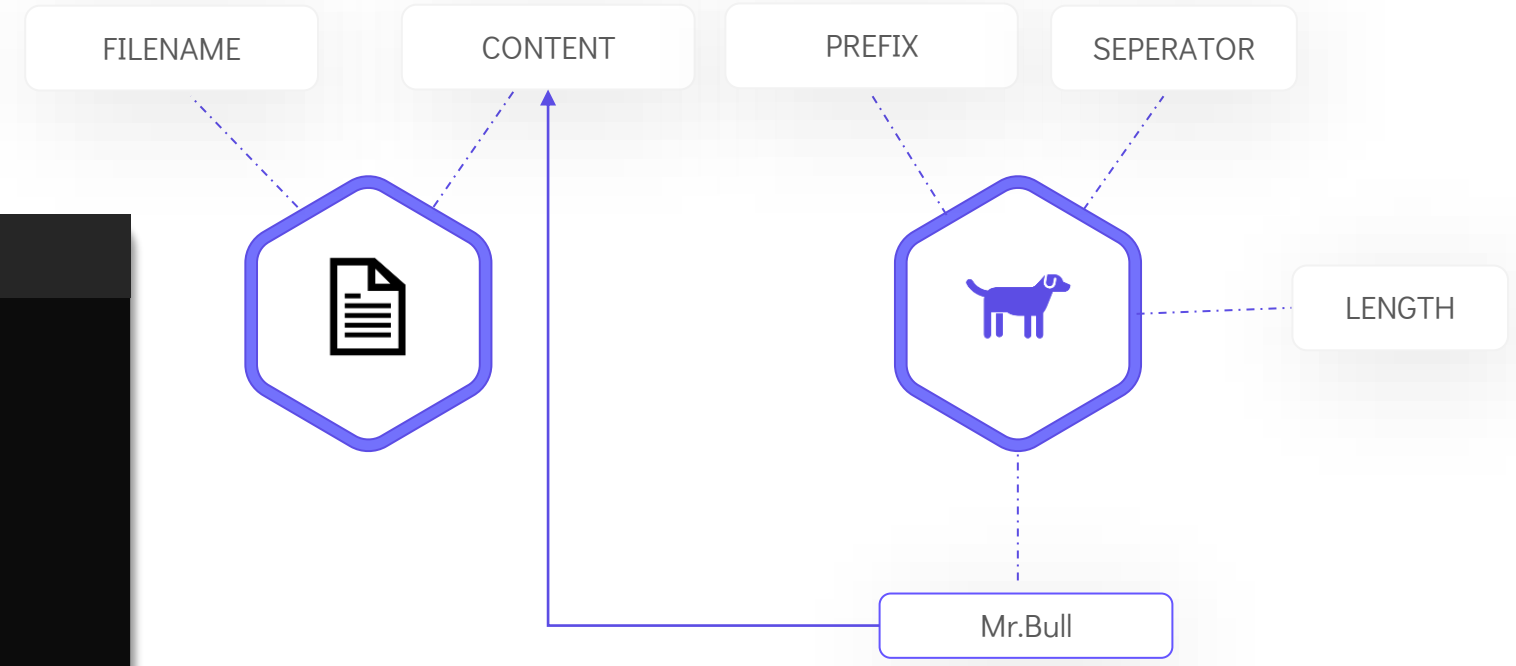


main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content =  "My favorite pet is Mr.Cat"
}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

>_

```
random_pet.my-pet: Creating...
local_file.pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mr.bull]
local_file.pet: Creation complete after 0s
[id=059090e865809f9b6debfda7aebf48fdce2220a6]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```



main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content =  "My favorite pet is Mr.Cat"
}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

```
>_

random_pet.my-pet: Creating...
local_file.pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mr.bull]
local_file.pet: Creation complete after 0s
[id=059090e865809f9b6debfda7aebf48fdce2220a6]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

Attribute Reference

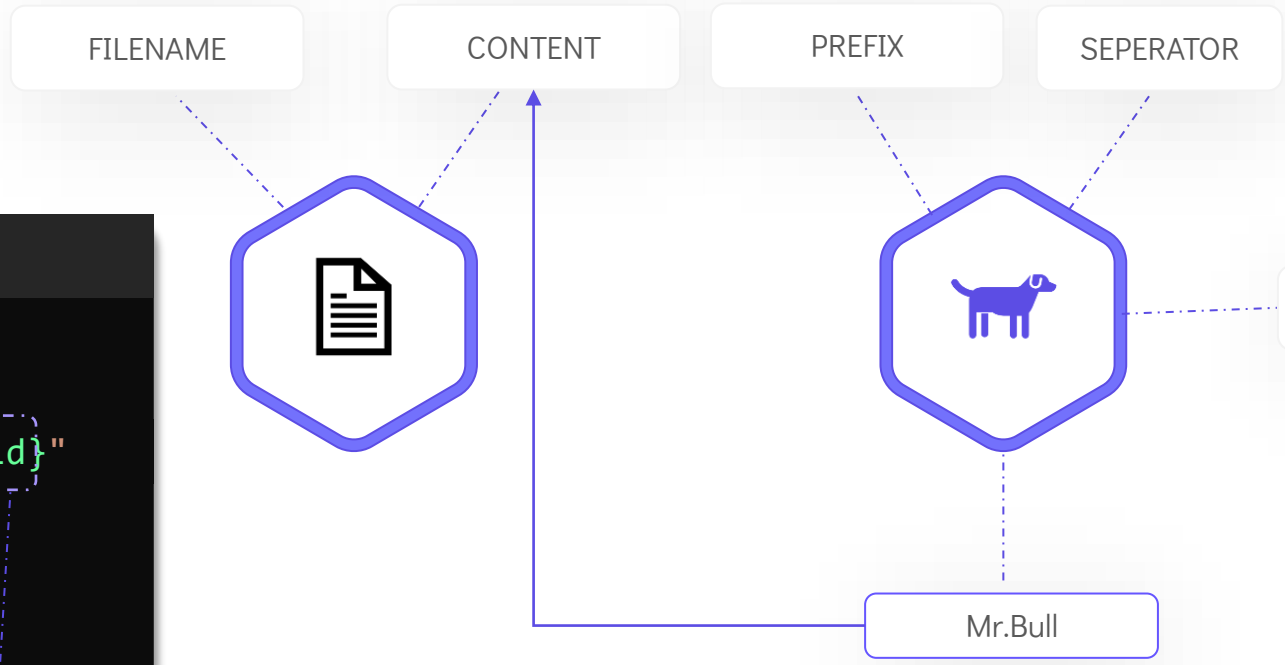
The following attributes are supported:

- `id` - (string) The random pet name

main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content =  "My favorite pet is ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```

```
>_

random_pet.my-pet: Creating...
local_file.pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mr.bull]
local_file.pet: Creation complete after 0s
[id=059090e865809f9b6debfda7aebf48fdce2220a6]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

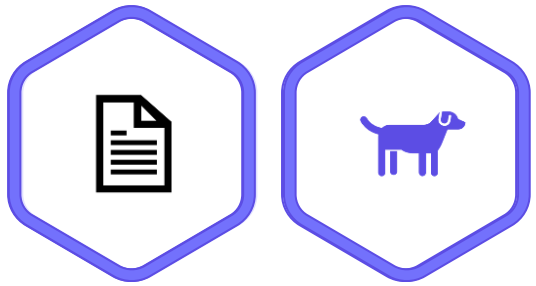


Attribute Reference

The following attributes are supported:

- `id` - (string) The random pet name

```
_file" "pet" {
var.filename

My favorite pet is Mr.Bull"


om_pet" "my-pet" {
r.prefix
var.separator
r.length
```



```
$ terraform apply
.
.
.
  # local_file.pet must be replaced
-/+ resource "local_file" "pet" {
      ~ content              = "My favorite pet is Mrs.Cat!" ->
"My favorite pet is Mr.bull" # forces replacement
        directory_permission = "0777"
        file_permission      = "0777"
        filename             = "/roots/pets.txt"
      ~ id                   =
"98af5244e23508cffd4a0c3c46546821c4ccbbd0" -> (known after
apply)
    }
.
.
local_file.pet: Destroying...
[id=98af5244e23508cffd4a0c3c46546821c4ccbbd0]
local_file.pet: Destruction complete after 0s
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=e56101d304de7cf1b1001102923c6bdeaa60c523]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

HANDS-ON LABS

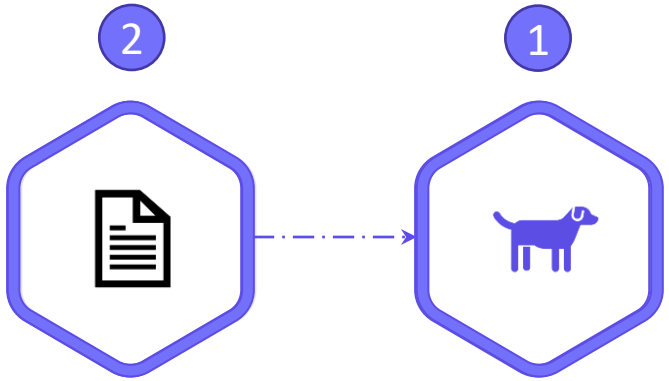
course on Terraform for the Absolute Beginners: htt ScZd

Resource Dependencies

Implicit Dependency

main.tf

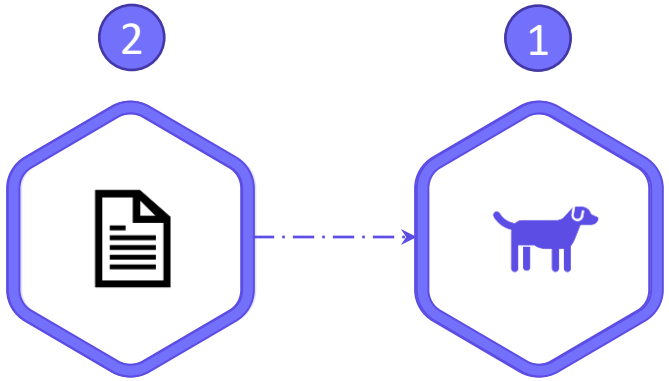
```
resource "local_file" "pet" {
    filename = var.filename
    content =  "My favorite pet is ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```



Explicit Dependency

main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content =  "My favorite pet is Mr.Cat"
    depends_on = [
      random_pet.my-pet
    ]


}

resource "random_pet" "my-pet" {
    prefix = var.prefix
    separator = var.separator
    length = var.length
}
```



HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

Output Variables

main.tf

```
resource "local_file" "pet" {
   filename = var.filename
   content =  "My favorite pet is ${random_pet.my-pet.id}"
}

resource "random_pet" "my-pet" {
   prefix = var.prefix
   separator = var.separator
   length = var.length
}

output pet-name {
  value       = random_pet.my-pet.id
  description = "Record the value of pet ID generated by the
 random_pet resource"
}
```

variables.tf

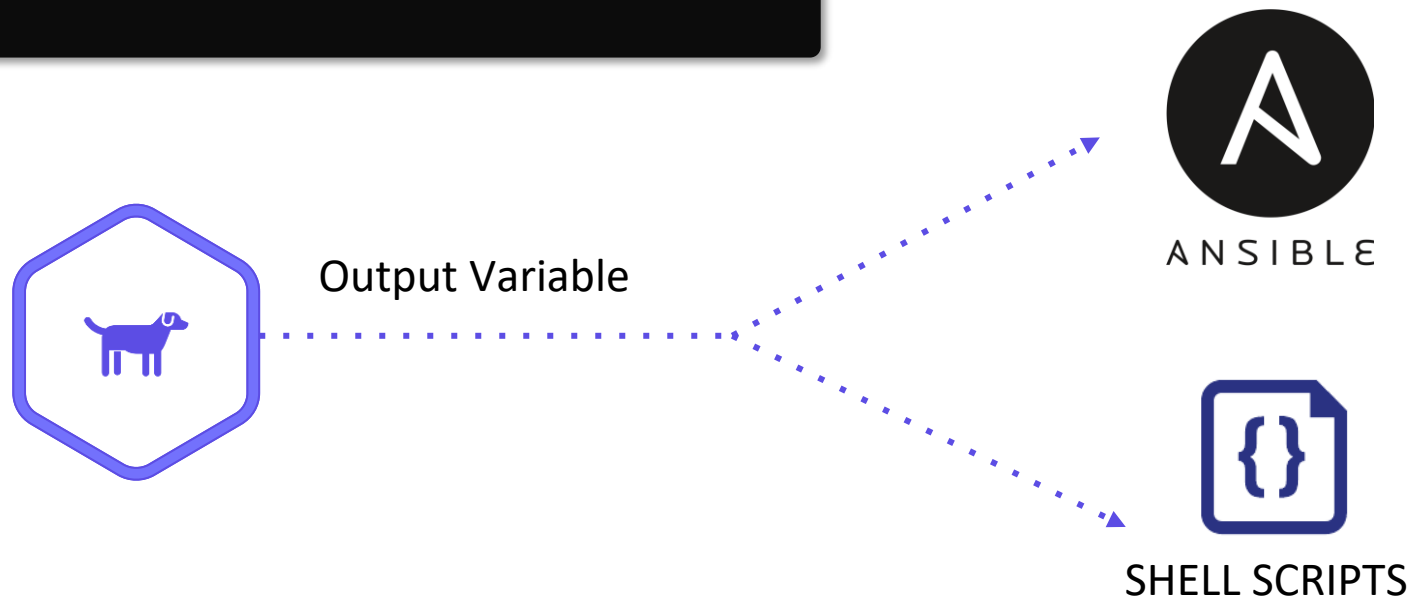
```
variable "filename" {
         default = "/root/pets.txt"
}
variable "content" {
         default = "I love pets!"
}
variable "prefix" {
         default = "Mrs"
}
variable "separator" {
         default = "."
}
variable "length" {
         default = "1"
}
```

```
output "<variable_name>" {
  value = "<variable_value>"
  <arguments>
}
```

```
>_
 $ terraform apply

 .
 .

Outputs:

pet-name = Mrs.gibbon
```

```
>_
$ terraform output
pet-name = Mrs.gibbon
```

```
>_
$ terraform output pet-name
Mrs.gibbon
```



HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

Introduction to Terraform State

```
>_
$ ls terraform-local-file
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content  = var.content
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "I love pets!"
}
```


```
>_
$ cd terraform-local-file

[terraform-local-file]$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints
in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that
may contain breaking
changes, we recommend adding version constraints in a
required_providers block
in your configuration, with the constraint strings
suggested below.

* hashicorp/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```


Check

```
>_
$ ls terraform-local-file
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content  = var.content
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "I love pets!"
}
```


```
>_
[terraform-local-file]$ terraform plan

Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan,
persisted to local or remote state storage.


------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbol
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "I love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

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

------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save thi
```


Check

```
>_
$ ls terraform-local-file
main.tf variables.tf
```

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content  = var.content
}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "I love pets!"
}
```


```
>_
[terraform-local-file]$ terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbol
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "I love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 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: yes

local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]
```


Check

```
>_

[terraform-local-file]$ cat /root/pets
I love pets!
```


```
>_

[terraform-local-file]$ terraform apply
local_file.pet: Refreshing state...
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]

Apply complete! Resources: 0 added, 0 changed, 0 destro
```


Check

```
>_
[terraform-local-file]$ ls
main.tf variables.tf terraform.tfstate
```


```
>_
[terraform-local-file]$ cat  terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 1,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31"
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "provider":
"provider[\"registry.terraform.io/hashicorp/local\"
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "I love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            "file_permission": "0777",
            "filename": "/root/pets.txt",
            "id":
"7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68",
            "sensitive_content": null
          },
          "private": "bnVsbA=="
```


Check

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "We love pets!"
}
```

>_

```
$ terraform plan
Refreshing Terraform state in-memory
prior to plan...
The refreshed state will be used to
calculate this plan, but will not be
persisted to local or remote state
storage.

local_file.pet: Refreshing state...
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e
1b68]
.
.
.
[Output Truncated ]
```

>_

```
[terraform-local-file]$ cat  terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 1,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31"
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "provider":
"provider[\"registry.terraform.io/hashicorp/local\"
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "I love pets!",
            "content_base64": null,
            "directory_permission": "0777",
            "file_permission": "0777",
            "filename": "/root/pets.txt",
            "id":
"7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68",
            "sensitive_content": null
          },
          "private": "bnVsbA=="
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "We love pets!"
}
```

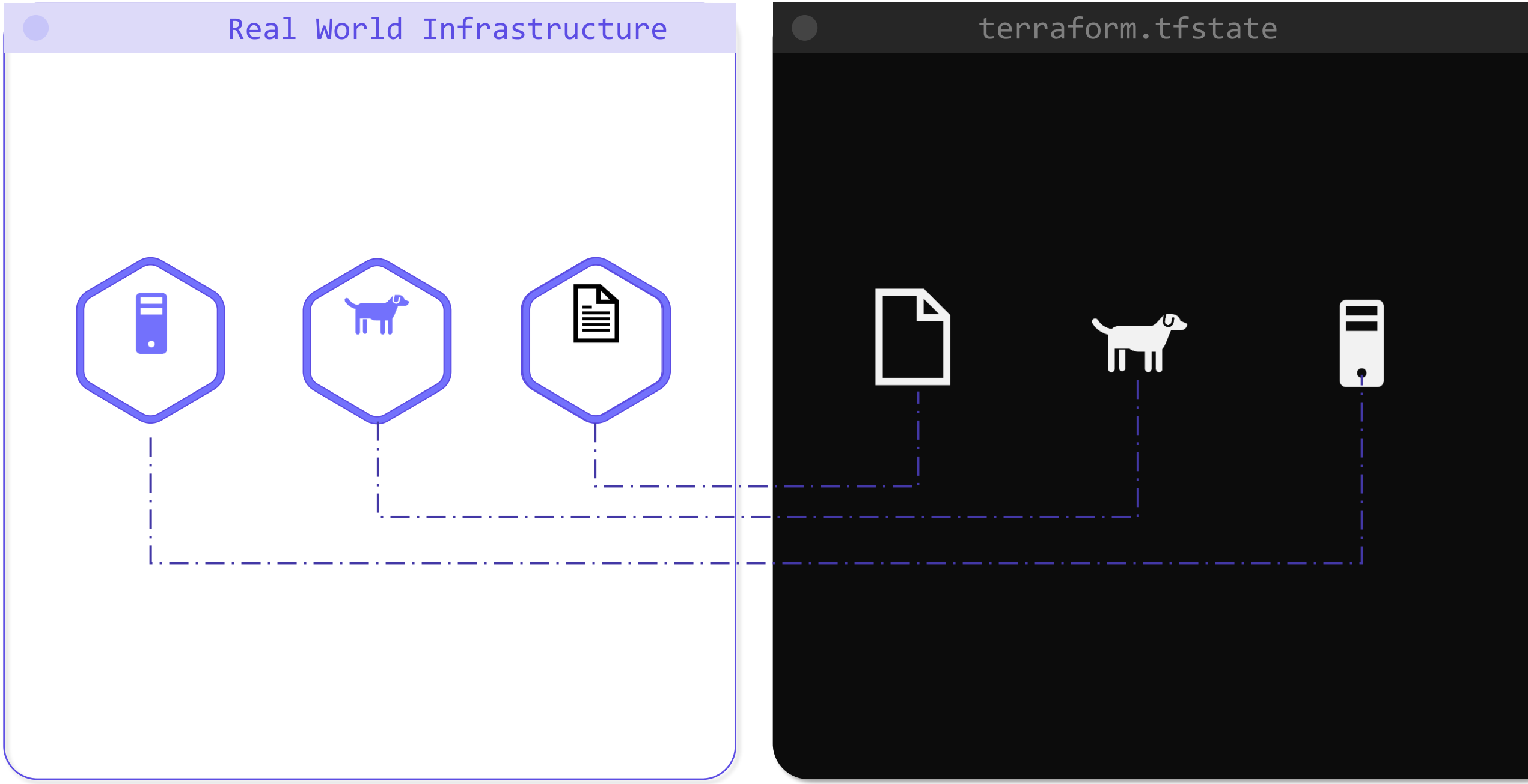
>_

```
$ terraform apply
local_file.pet: Refreshing state...
[id=7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68]

Terraform will perform the following actions:

  # local_file.pet must be replaced
-/+ resource "local_file" "pet" {
      ~ content              = "I love pets!" -
> "We love pets!" # forces replacement
        directory_permission = "0777"
        file_permission      = "0777"
        filename             = "/root/pets.txt"
      ~ id                   =
"7e4db4fbfdbb108bdd04692602bae3e9bd1e1b68" ->
(known after apply)
    }
```

>_

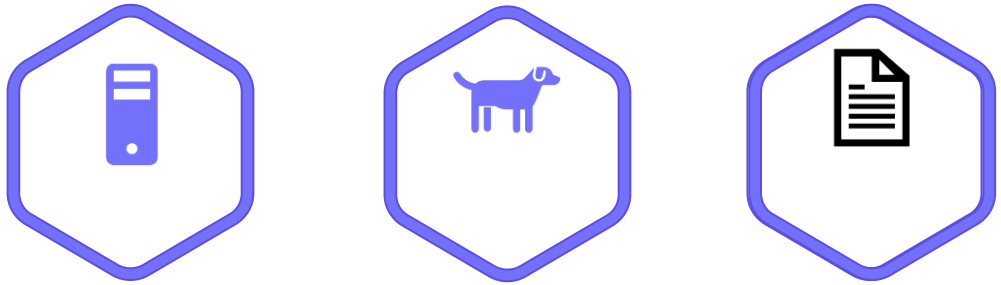
```
[terraform-local-file]$ cat  terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 1,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31"
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "provider":
"provider[\"registry.terraform.io/hashicorp/local\"
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!“,
            "content_base64": null,
            "directory_permission": "0777",
            "file_permission": "0777",
            "filename": "/root/pets.txt",
            "id":
"7e4db4fbfdbb108bdd04692602bae3e9bc4d1c14",
            "sensitive_content": null
          },
          "private": "bnVsbA=="
```

Check

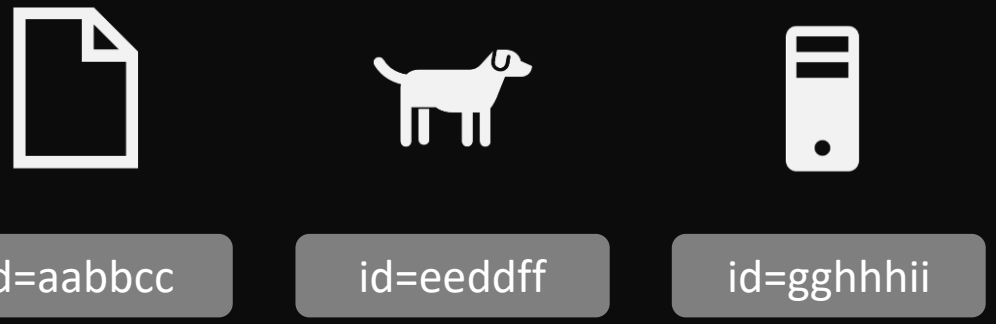


Purpose of State

Real World Infrastructure



terraform.tfstate

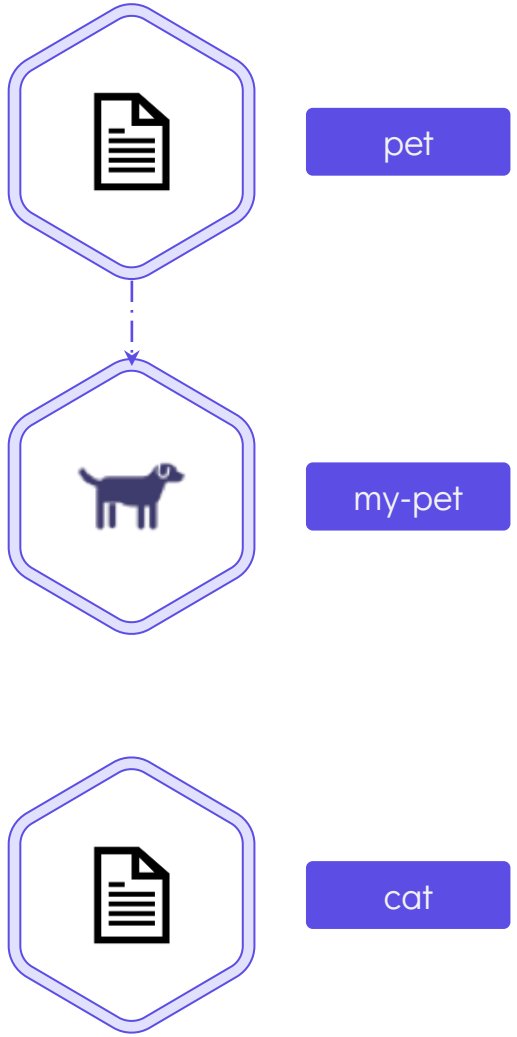


Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Tracking Metadata

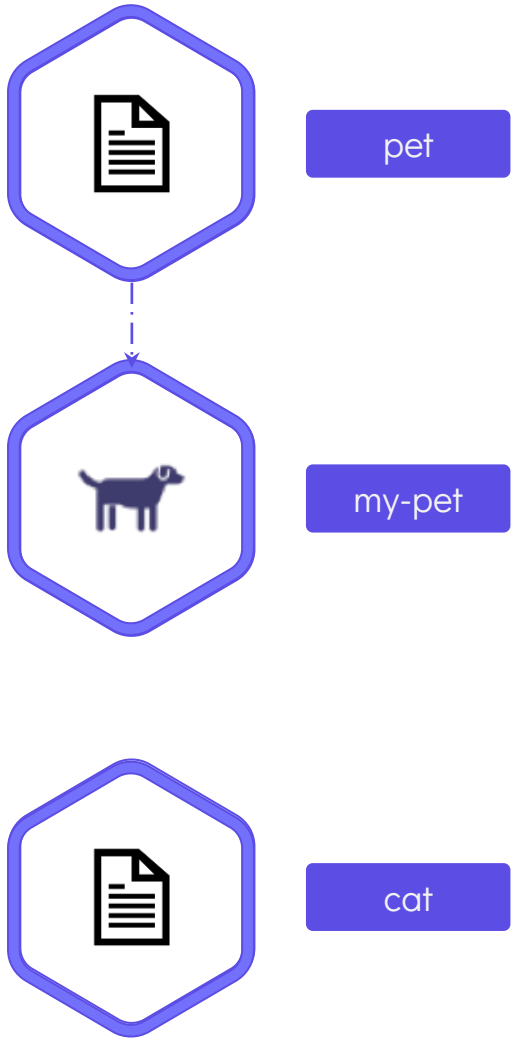
main.tf

```
resource "local_file" "pet" {
  filename = "/root/pet.txt"
  content  = "My favorite pet is ${random_pet.my-pet.id}!"
}
resource "random_pet" "my-pet" {
  length = 1
}
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```



Tracking Metadata

```
$ terraform apply

.
.
.
Plan: 3 to add, 0 to change, 0 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: yes

local_file.cat: Creating...
random_pet.my-pet: Creating...
local_file.cat: Creation complete after 0s
[id=fe448888891fc40342313bc44a1f1a8986520c89]
random_pet.my-pet: Creation complete after 0s [id=yak]

local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=28b373c6c1fa3fce132a518eadd0175c98f37f20]

Apply complete! Resources: 3 added, 0 changed, 0
destroyed.
```

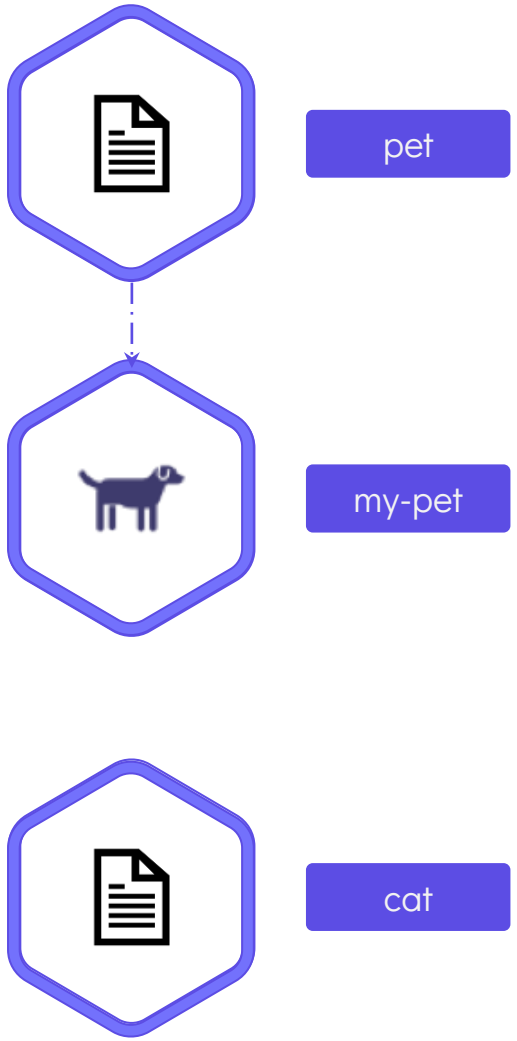


Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Tracking Metadata

main.tf

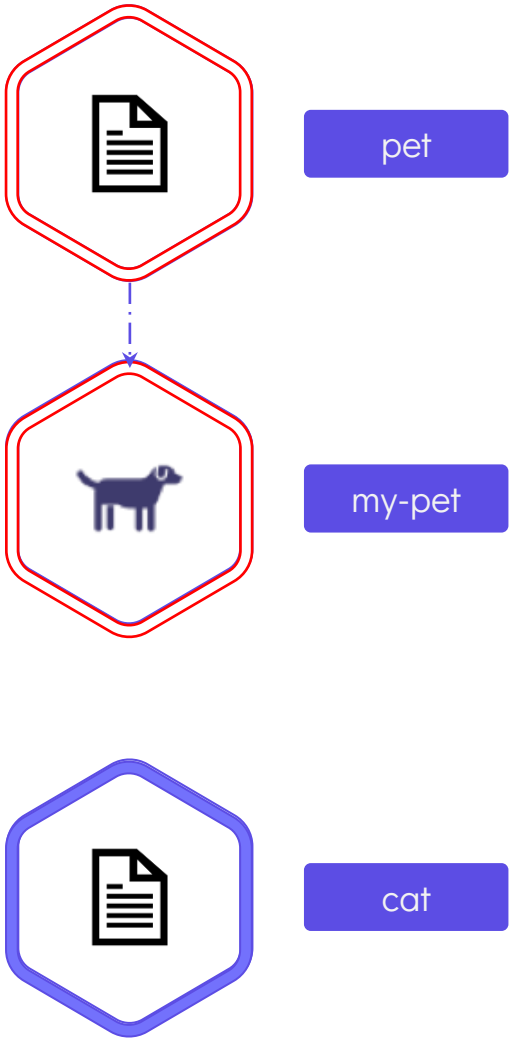
```
resource "local_file" "pet" {
  filename = "/root/pet.txt"
  content  = "My favorite pet is ${random_pet.my-pet.id}!"
}
resource "random_pet" "my-pet" {
  length = 1
}
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```



Tracking Metadata

main.tf

```
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```



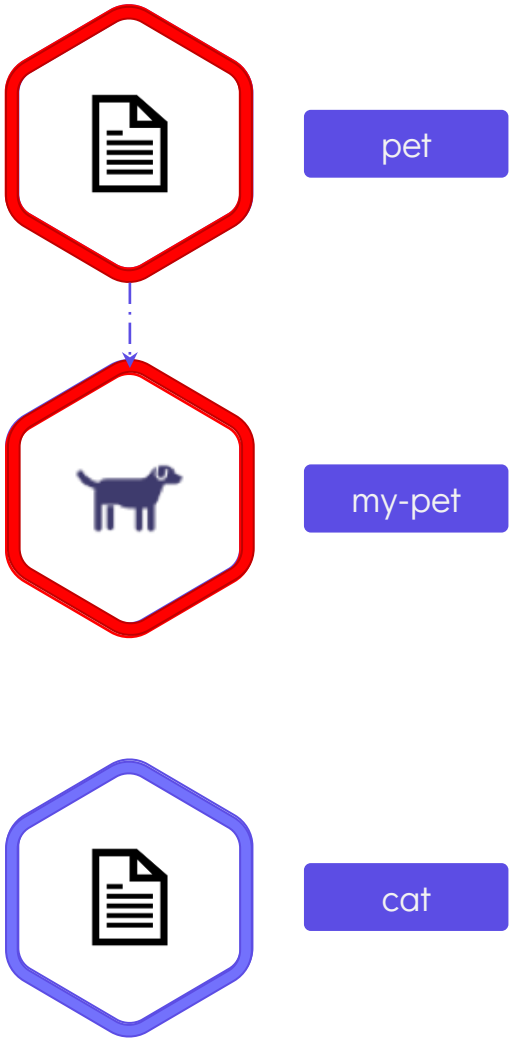
Tracking Metadata

main.tf

```
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```

>_

```
$ cat terraform.tfstate
{
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "My favorite pet is yak!",
.           },
          "private": "bnVsbA==",
          "dependencies": [
            "random_pet.my-pet"
          ]
```



ut our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

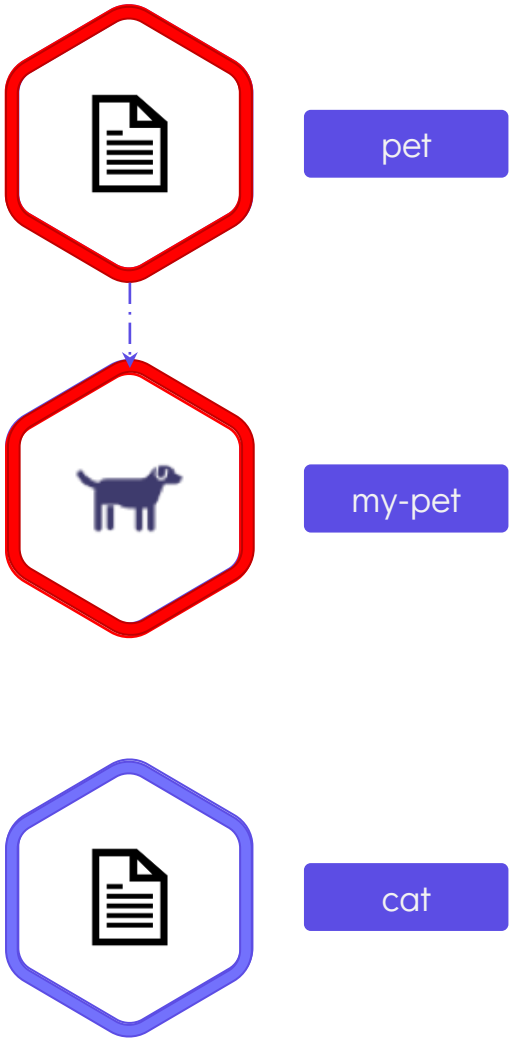
Tracking Metadata

main.tf

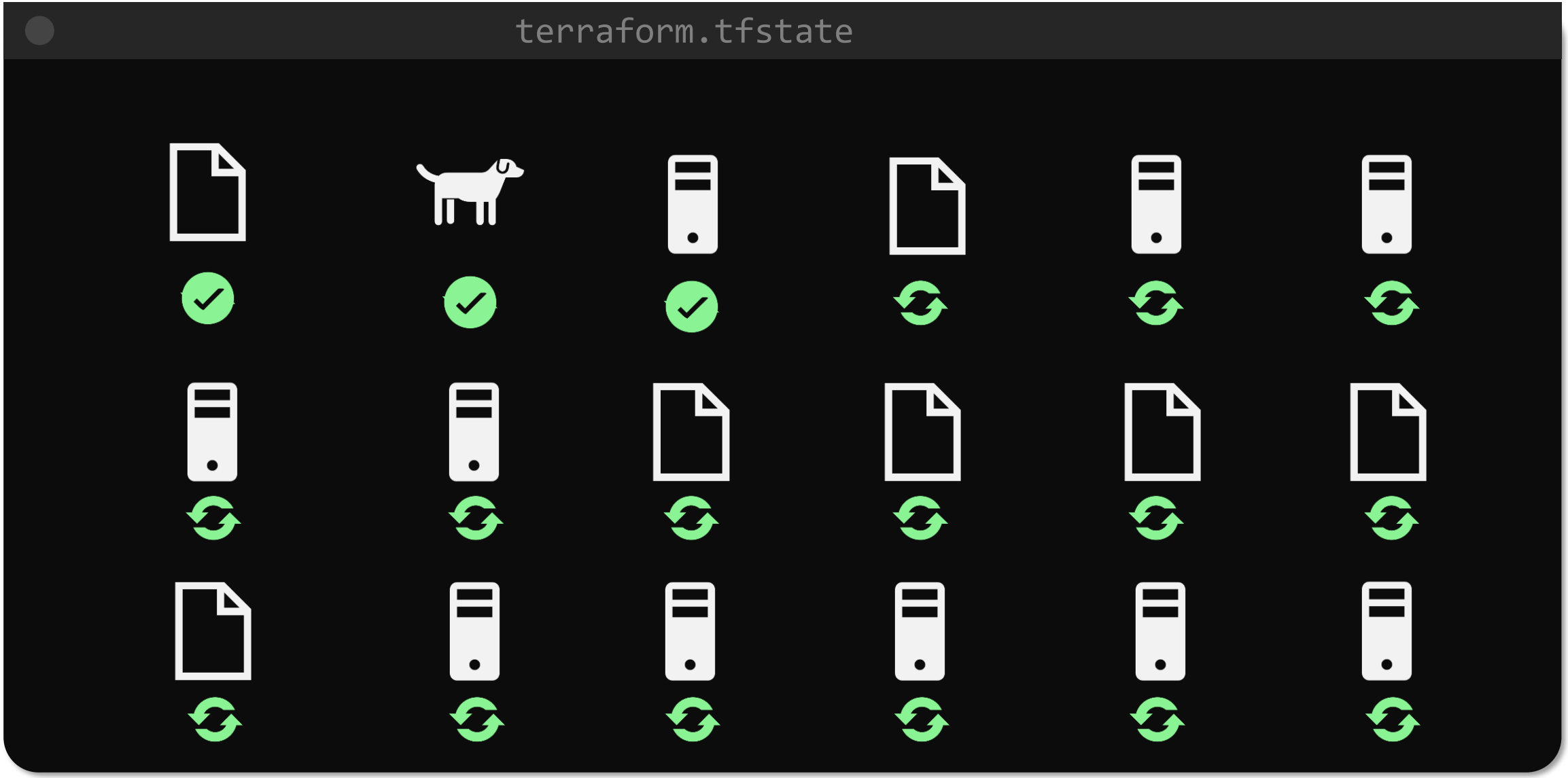
```
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```

>_

```
$ terraform apply

Plan: 0 to add, 0 to change, 2 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: yes

local_file.pet: Destroying...
[id=28b373c6c1fa3fce132a518eadd0175c98f37f20]
local_file.pet: Destruction complete after 0s


random_pet.my-pet: Destroying... [id=yak]
```



Performance



Performance

terraform.tfstate

```
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
...
```

>_

```
$ terraform plan --refresh=false

An execution plan has been generated and is shown
below.
Resource actions are indicated with the following
symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # local_file.cat must be replaced
-/+ resource "local_file" "pet" {
      ~ content              = "I like cats too!" ->
"Dogs are awesome!" # forces replacement
        directory_permission = "0777"
        file_permission      = "0777"
        filename             = "/root/pets.txt"
      ~ id                   =
"cba595b7d9f94ba1107a46f3f731912d95fb3d2c" -> (known
after apply)
    }

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

------------------------------------------------
```

Collaboration

terraform.tfstate

```
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
...
```

```
>_

$ ls

main.tf variables.tf terraform.tfstate
```

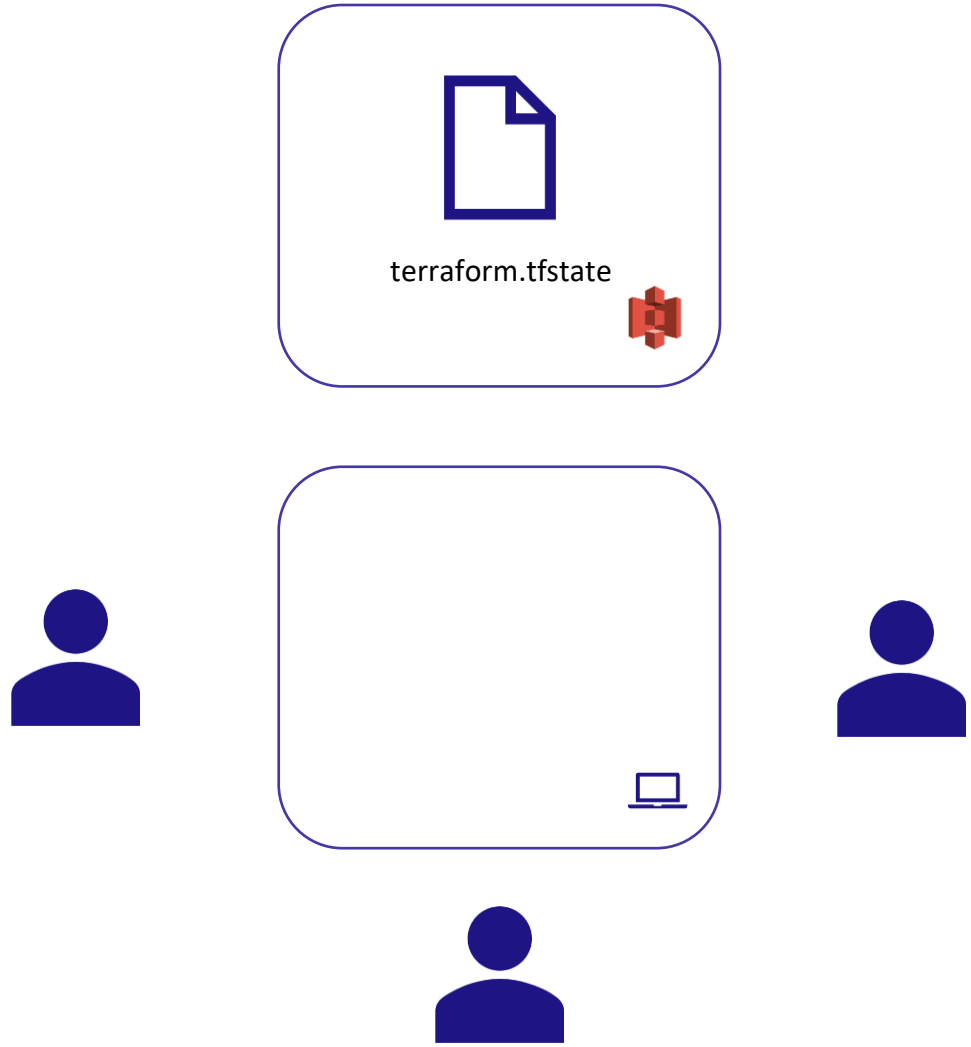
Collaboration

AWS S3 | HashiCorp Consul

Google Cloud Storage | Terraform Cloud

terraform.tfstate

```
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 4,
  "lineage": "e35dde72-a943-de50-3c8b-1df8986e5a31",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "local_file",
      "name": "pet",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "We love pets!",
            "content_base64": null,
            "directory_permission": "0777",
...
```



Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

Terraform State Considerations

Sensitive Data

terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": ""
```

Terraform State Considerations

Remote State Backends

Version Control

terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "volume_id": "vol-070720a3636979c22",
                "volume_size": 8,
```

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pet.txt"
  content  = "My favorite pet is Mr.Whiskers!"
}
resource "random_pet" "my-pet" {
  length = 1
}
resource "local_file" "cat" {
  filename = "/root/cat.txt"
  content  = "I like cats too!"
}
```

our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

No Manual Edits

terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": ""
```

Terraform Commands

terraform validate

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permissions = "0700"
}
```

>_

```
$ terraform validate
Success! The configuration is valid.

$ terraform validate

Error: Unsupported argument

  on main.tf line 4, in resource "local_file" "pet":
   4:     file_permissions = "0777"

An argument named "file_permissions" is not expected
here. Did you mean "file_permission"?
```

terraform fmt

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"
}
```

>_

```
$ terraform fmt
main.tf
```

terraform fmt

main.tf

```
resource "local_file" "pet" {
    filename        = "/root/pets.txt"
    content         = "We love pets!"
    file_permission = "0700"
}
```

>_

```
$ terraform fmt
main.tf
```

terraform show

```
$ terraform show
# local_file.pet:
resource "local_file" "pet" {
    content              = "We love pets!"
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
    id                   =
"cba595b7d9f94ba1107a46f3f731912d95fb3d2c"
}
```

```
$ terraform show -json
{"format_version":"0.1","terraform_version":"0.13.0
","values":{"root_module":{"resources":[{"address":
"local_file.pet","mode":"managed","type":"local_fil
e","name":"pet","provider_name":"registry.terraform
.io/hashicorp/local","schema_version":0,"values":{"
content":"We love
pets!","content_base64":null,"directory_permission"
:"0777","file_permission":"0777","filename":"/root/
pets.txt","id":"cba595b7d9f94ba1107a46f3f731912d95f
b3d2c","sensitive_content":null}}]}}}
```

terraform providers

main.tf

```
resource "local_file" "pet" {
    filename        = "/root/pets.txt"
    content         = "We love pets!"
    file_permission = "0700"
}
```

>_

```
$ terraform providers
Providers required by configuration:
.
└── provider[registry.terraform.io/hashicorp/local]

Providers required by state:

    provider[registry.terraform.io/hashicorp/local]



$ terraform providers mirror /root/terraform/new_local_file
- Mirroring hashicorp/local...
  - Selected v1.4.0 with no constraints
  - Downloading package for windows_amd64...
  - Package authenticated: signed by HashiCorp
```

terraform output

main.tf

```
resource "local_file" "pet" {
  filename        = "/root/pets.txt"
  content         = "We love pets!"
  file_permission = "0777"
}
resource "random_pet" "cat" {
  length    = "2"
  separator = "-"
}
output content {
  value       = local_file.pet.content
  sensitive   = false
  description = "Print the content of the file"

}
output pet-name {
  value       = random_pet.cat.id
  sensitive   = false
  description = "Print the name of the pet"

}
```

>_

```
$ terraform output
content = We love pets!
pet-name = huge-owl
```

```
$ terraform output pet-name
pet-name = huge-owl
```

terraform refresh

main.tf

```
resource "local_file" "pet" {
  filename        = "/root/pets.txt"
  content         = "We love pets!"
  file_permission = "0777"
}
resource "random_pet" "cat" {
  length    = "2"
  separator = "-"
}
```

>_

```
$ terraform refresh
random_pet.cat: Refreshing state... [id=huge-owl]
local_file.pet: Refreshing state...
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]



$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this
plan, but will not be
persisted to local or remote state storage.

random_pet.cat: Refreshing state... [id=huge-owl]
local_file.pet: Refreshing state...
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
------------------------------------------------------------

No changes. Infrastructure is up-to-date.
```

terraform graph

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content  = "My favorite pet is ${random_pet.my-pet.id}"
}
resource "random_pet" "my-pet" {
  prefix = "Mr"
  separator = "."
  length = "1"
}
```

>_

```
 $ terraform graph
digraph {
        compound = "true"
        newrank = "true"
        subgraph "root" {
                "[root] local_file.pet (expand)" [label = "local_file.pet", shape = "box"]
                "[root] provider[\"registry.terraform.io/hashicorp/local\"]" [label = "provider[\"registry.terraform.io/hashicorp/local\"]", shape = "diamond"]
                "[root] provider[\"registry.terraform.io/hashicorp/random\"]" [label = "provider[\"registry.terraform.io/hashicorp/random\"]", shape = "diamond"]
                "[root] random_pet.my-pet (expand)" [label = "random_pet.my-pet", shape = "box"]
                "[root] local_file.pet (expand)" -> "[root] provider[\"registry.terraform.io/hashicorp/local\"]"
                "[root] local_file.pet (expand)" -> "[root] random_pet.my-pet (expand)"
                "[root] meta.count-boundary (EachMode fixup)" -> "[root] local_file.pet (expand)"
                "[root] provider[\"registry.terraform.io/hashicorp/local\"] (close)" ->
```

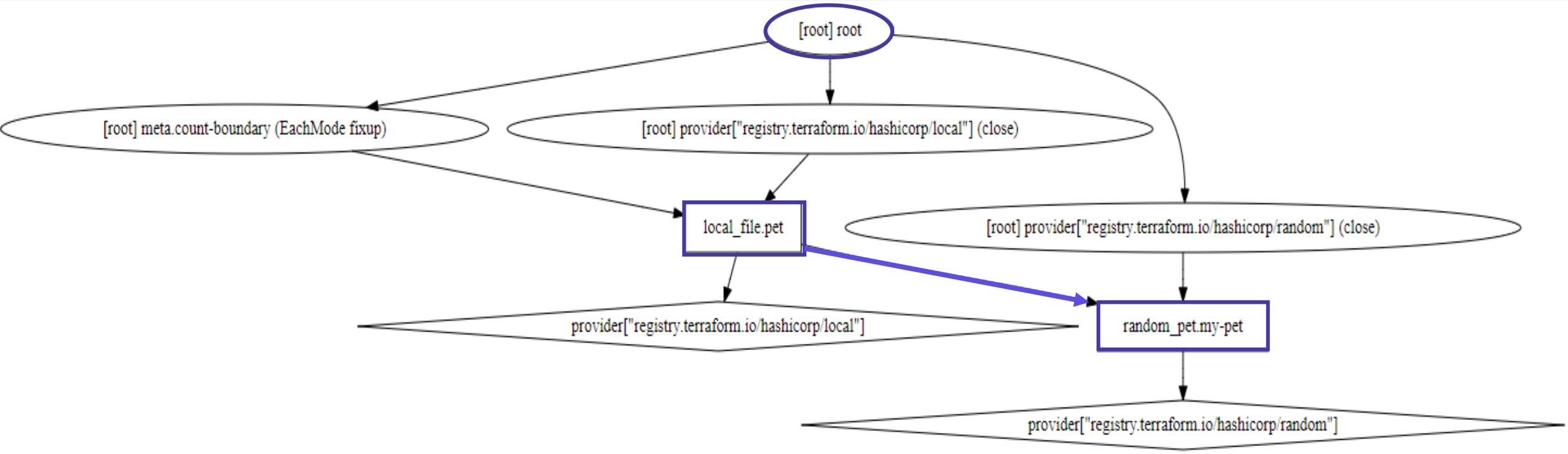
terraform graph

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content  = "My favorite pet is ${random_pet.my-pet.id}"
}
```

>_

```
$ apt update
$ apt install graphviz -y
$ terraform graph | dot -Tsvg > graph.svg
```



HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

Mutable vs Immutable Infrastructure

terraform validate

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"
}
```

>_

```
 $ terraform apply

# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces
replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after
apply)
    }

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: yes

local_file.pet: Destroying...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s
```

upgrade-nginx.sh

ANSIBLE

Mutable Infrastructure

Configuration Drift

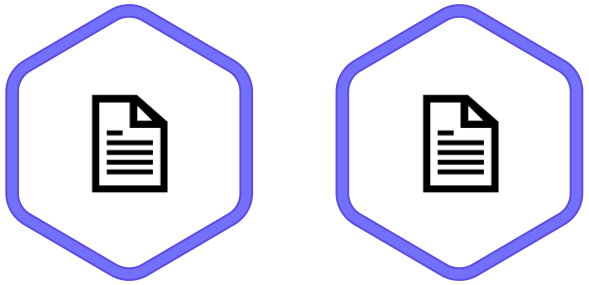
Immutable Infrastructure

Immutable Infrastructure

Immutable Infrastructure

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"
}
```



>_

```
$ terraform apply

# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces
replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after
apply)
    }

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

local_file.pet: Destroying...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```


Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

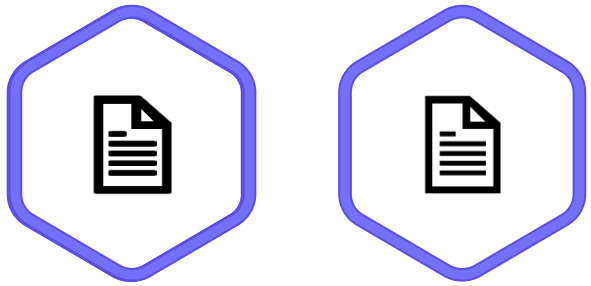
KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Lifecycle Rules

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"
}
```



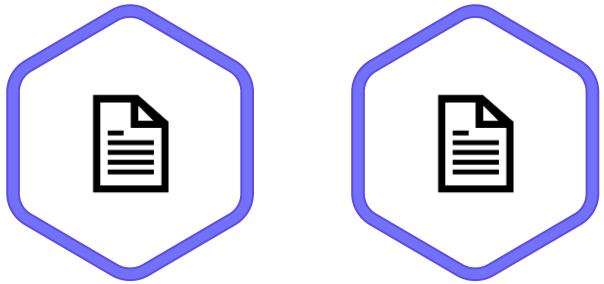
>_

```
$ terraform apply

# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces
replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after
apply)
    }

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

local_file.pet: Destroying...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

create_before_destroy

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"

    lifecycle  {
        create_before_destroy  = true
    }


}
```



>_

```
$ terraform apply

# local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0755" # forces repl
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after ap
    }

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

...

local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

local_file.pet: Destroying...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

prevent_destroy

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
    file_permission = "0700"

    lifecycle  {
        prevent_destroy  = true
    }


}
```

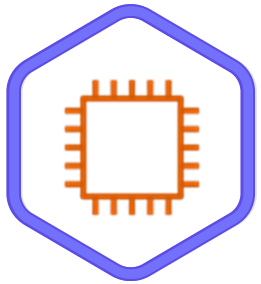
>_

```
$ terraform apply
local_file.my-pet: Refreshing state...
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]

Error: Instance cannot be destroyed

  on main.tf line 1:
   1: resource "local_file" "my-pet" {

Resource local_file.my-pet has
lifecycle.prevent_destroy set, but the plan calls
for this resource to be destroyed. To avoid this error
and continue with the plan, either disable
lifecycle.prevent_destroy or reduce the scope of the
plan using the -target flag.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
      Name = “ProjectA-Webserver"
  }
 }
```



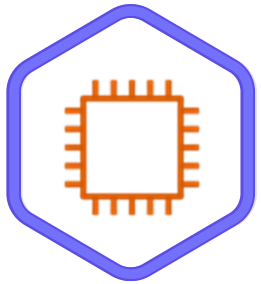
>_

```
$ terraform apply

...
Terraform will perform the following actions:

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                          = "ami-0edab43b6fa892279"
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t2.micro“
      + tags                         {
          + "Name"       = "ProjectA-WebServer"
        }
.
aws_instance.webserver: Creation complete after 33s [id=i-
05cd83b221911acd5]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
      Name = "ProjectA-Webserver"
  }
 }
```



>_

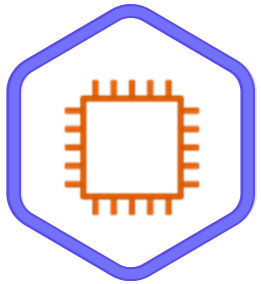
```
$ terraform apply
aws_instance.webserver: Refreshing state... [id=i-
05cd83b221911acd5]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_instance.webserver will be updated in-place
  ~ resource "aws_instance" "webserver" {
.
.
~ tags                          = {
  ~ "Name"       = "ProjectB-WebServer" -> "ProjectA-WebServer"
        }
.
.
Apply complete! Resources: 0 added, 1 changed, 0 destroyed.
```

☑	ProjectB-WebServer	i-05cd83b221911acd5	Running	t2.micro	2/2 checks ...

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
      Name = “ProjectA-Webserver"
  }
  lifecycle {
        ignore_changes = [
          tags

        ]
    }
}
```



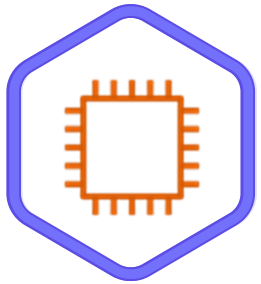
>_

```
$ terraform apply

aws_instance.webserver: Refreshing state... [id=i-
05cd83b221911acd5]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

ignore_changes

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
      Name = “ProjectA-Webserver"
  }
  lifecycle {
        ignore_changes = all



    }
}
```



>_

```
$ terraform apply

aws_instance.webserver: Refreshing state... [id=i-
05cd83b221911acd5]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

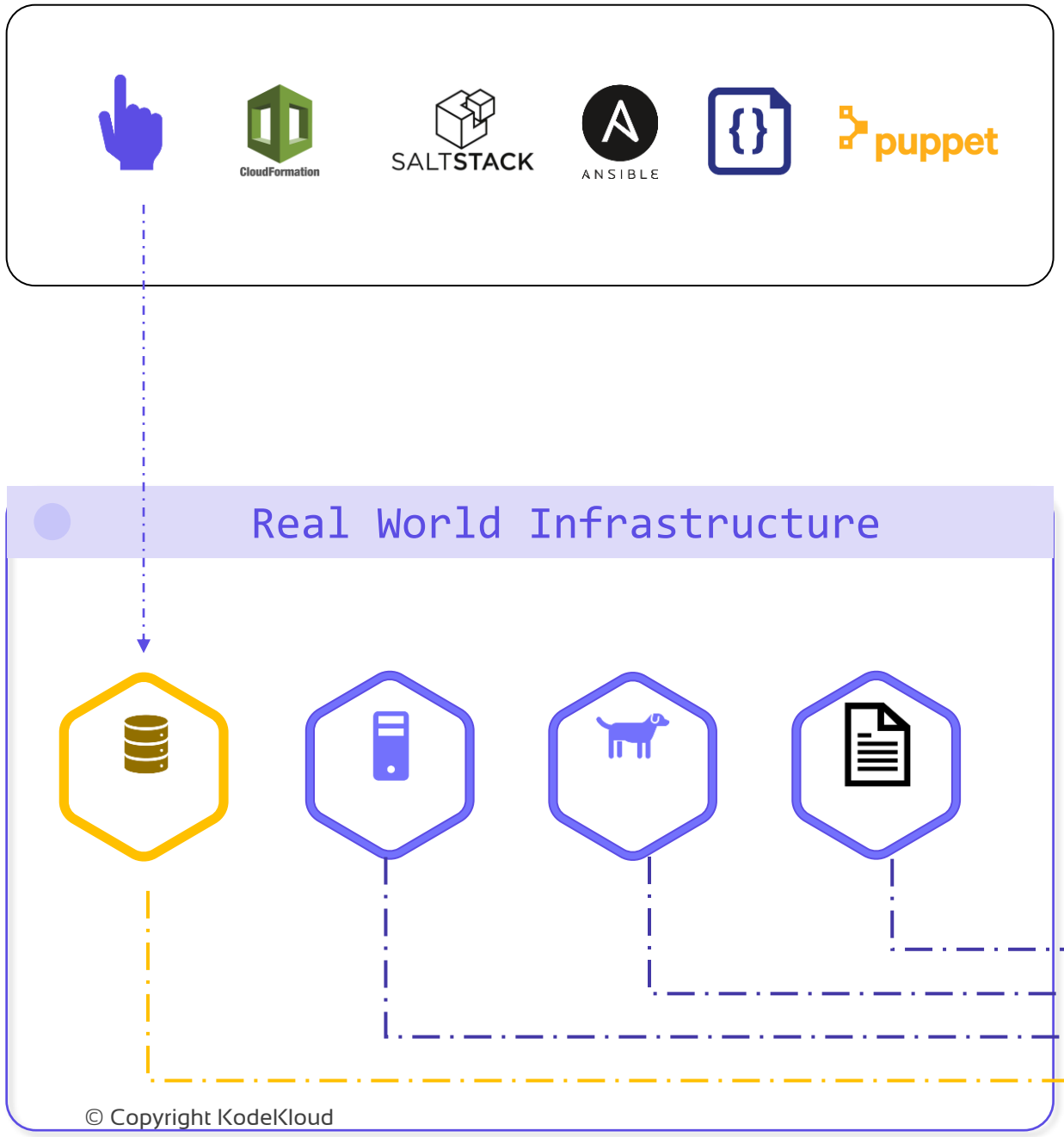
Order	Option	
1	create_before_destroy	Create the resource first and then destroy older
2	prevent_destroy	Prevents destroy of a resource
3	ignore_changes	Ignore Changes to Resource Attributes (specific/all)

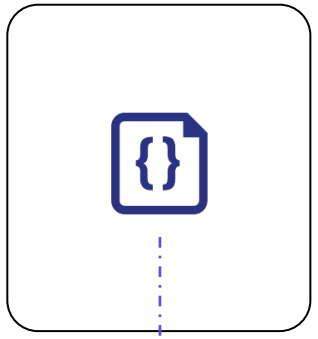
HANDS-ON LABS

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Data Sources


CloudFormation
SALTSTACK
ANSIBLE
puppet
HashiCorp
Terraform
Real World Infrastructure
terraform.tfstate
© Copyright KodeKloud



main.tf

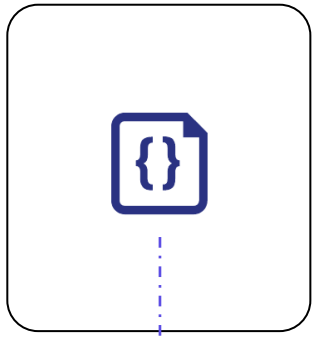
```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}
```

```
>_
$ cat /root/dog.txt
Dogs are awesome!
```

Real World Infrastructure

terraform.tfstate

Data Sources

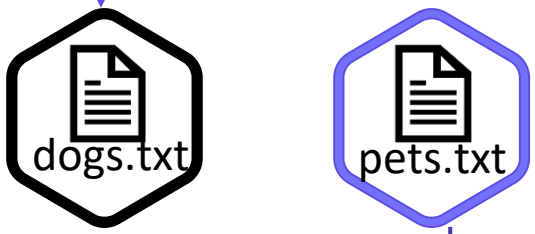


```
>_
$ cat /root/dog.txt
Dogs are awesome!
```

main.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = data.local_file.dog.content
}

data "local_file" "dog" {
    filename = "/root/dog.txt"
}
```

Real World Infrastructure



terraform.tfstate

LOCAL DOCUMENTATION

Filter

local provider

Resources

local_file

Data Sources

local_file

Argument Reference

The following argument is required:

- `filename` - (Required) The path to the file that will be read. The data source will return an error if the file does not exist.

Attributes Exported

The following attribute is exported:

- `content` - The raw content of the file that was read.
- `content_base64` - The base64 encoded version of the file content (use this when dealing with binary data).

Resource	Data Source
Keyword: **resource**	Keyword: **data**
Creates, Updates, Destroys Infrastructure	Only **Reads** Infrastructure
Also called **Managed Resources**	Also called **Data Resources**

HANDS-ON LABS

course on Terraform for the solute Beginners: htt ScZd

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Meta Arguments

main.tf

```
resource "local_file" "pet" {
    filename = var.filename
    content = var.content

}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
variable "content" {
  default = "I love pets!"
}
```

Shell Scripts

create_files.sh

```
#!/bin/bash

for i in {1..3}
  do
    touch /root/pet${i}
  done
```

>_

```
$ ls -ltr /root/
-rw-r--r-- 1 root root    0 Sep  9 02:04 pet2
-rw-r--r-- 1 root root    0 Sep  9 02:04 pet1
-rw-r--r-- 1 root root    0 Sep  9 02:04 pet3
```

Iteration	filename
1	/root/pet1
2	/root/pet2
3	/root/pet3

Meta Arguments

depends_on

main.tf

```
resource "local_file" "pet" {
  filename = var.filename
  content  = var.content
  depends_on = [
    random_pet.my-pet
  ]
}
resource "random_pet" "my-pet" {
  prefix    = var.prefix
  separator = var.separator
  length    = var.length
}
```

lifecycle

main.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
   file_permission = "0700
   lifecycle  {
     create_before_destroy  = true
   }
}
```

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Count

count

main.tf

```
resource "local_file" "pet" {
    filename = var.filename

    count    = 3

}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
```

>_

```
$ terraform plan

[Output Truncated]
Terraform will perform the following actions:
...
# local_file.pet[2] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

Plan: 3 to add, 0 to change, 0 to destroy.
```

count

main.tf

```
resource "local_file" "pet" {
    filename = var.filename

    count    = 3

}
```

variables.tf

```
variable "filename" {
  default = "/root/pets.txt"
}
```

>_

```
$ ls /root
pet.txt
```

count

main.tf

```
resource "local_file" "pet" {
    filename = var.filename[count.index]
    count    = 3
}
```

variables.tf

```
variable "filename" {
  default = [
    "/root/pets.txt",
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

>_

```
$ ls /root
pets.txt
dogs.txt
cats.txt
```

Length Function

main.tf

```
resource "local_file" "pet" {
    filename = var.filename[count.index]
    count    = length(var.filename)
}
```

variables.tf

```
variable "filename" {
  default = [
    "/root/pets.txt",
    "/root/dogs.txt",
    "/root/cats.txt",
    "/root/cows.txt",
    "/root/ducks.txt"
  ]
}
```


```
>_
$ ls /root
pets.txt
dogs.txt
cats.txt
```

Length Function

variable	function	value
fruits = ["apple", "banana", "orange"]	length(fruits)	3
cars = [“honda”, “bmw”, “nissan”, “kia”]	length(cars)	4
colors = [“red”, “purple”]	length(colors)	2

LengthFunction

main.tf

```
resource "local_file" "pet" {
    filename = var.filename[count.index]
    count    = length(var.filename)
}
```

variables.tf

```
variable "filename" {
  default = [
    "/root/pets.txt",
    "/root/dogs.txt",
    "/root/cats.txt",
    "/root/cows.txt",
    "/root/ducks.txt"
  ]
}
```


```
>_
$ ls /root
pets.txt
dogs.txt
cats.txt
```

```
$ terraform apply

.
.
Terraform will perform the following actions:

  # local_file.pet[0] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

  # local_file.pet[1] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/dogs.txt"
      + id                   = (known after apply)
    }

  # local_file.pet[2] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/cats.txt"
      + id                   = (known after apply)
    }
```

```
$ ls /root
pet.txt
dogs.txt
cats.txt
```

main.tf

```
resource "local_file" "pet" {
    filename = var.filename[count.index]

    count    = length(var.filename)
}
```

variables.tf

```
variable "filename" {
    default = [

      "/root/dogs.txt",
      "/root/cats.txt"
    ]

}
```

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

main.tf

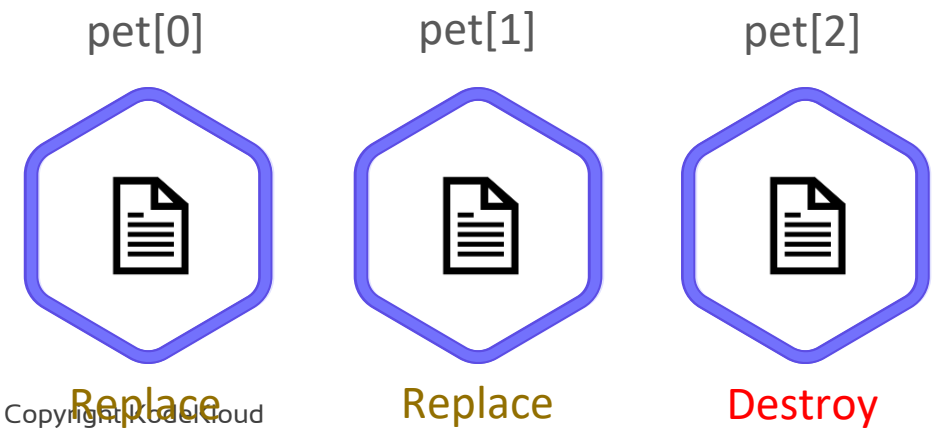
```
resource "local_file" "pet" {
    filename = var.filename[count.index]

    count    = length(var.filename)

}
```

variables.tf

```
variable "filename" {
    default = [

      "/root/dogs.txt",
      "/root/cats.txt"
    ]

}
```

>_

```
$ terraform plan

...
  # local_file.pet[0] must be replaced
-/+ resource "local_file" "pet" {
        directory_permission = "0777"
        file_permission      = "0777"
      ~ filename             = "/root/pets.txt" -> "/root/dogs.txt" #
forces replacement
    }
  # local_file.pet[1] must be replaced
-/+ resource "local_file" "pet" {
        directory_permission = "0777"
        file_permission      = "0777"
      ~ filename             = "/root/dogs.txt" -> "/root/cats.txt" #
forces replacement
    }
  # local_file.pet[2] will be destroyed
  - resource "local_file" "pet" {
      - directory_permission = "0777" -> null
      - file_permission      = "0777" -> null
```

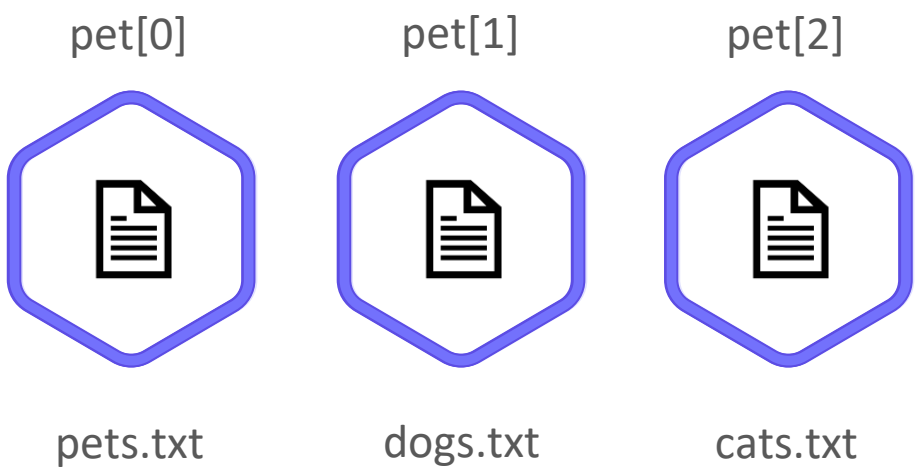


main.tf

```
resource "local_file" "pet" {
    filename = var.filename[count.index]

    count    = length(var.filename)
}

output "pets" {
  value = local_file.pet
}
```

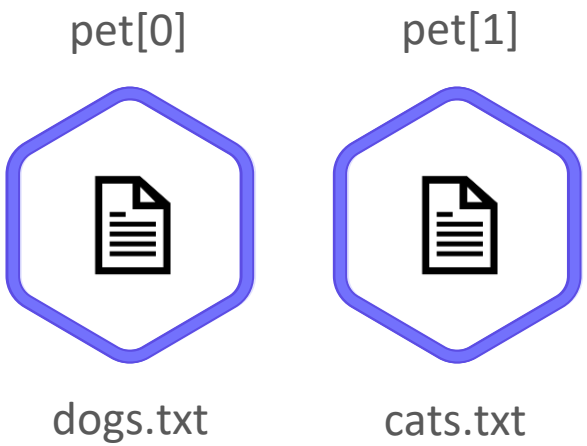
>_

```
$ terraform output
Outputs:

pets = [
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/pets.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/dogs.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/cats.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
]
```



variables.tf

```
variable "filename" {
  default = [

    "/root/dogs.txt",
    "/root/cats.txt"
  ]

}
```



variables.tf

```
variable "filename" {
  default = [

    "/root/dogs.txt",
    "/root/cats.txt"
  ]

}
```

Resource	Resource Updates	Action
pet[0]	/root/pets.txt" -> "/root/dogs.txt"	**Destroy and Replace**
pet[1]	"/root/dogs.txt" -> "/root/cats.txt"	**Destroy and Replace**
pet[2]	Does not Exist	**Destroy**

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

for_each

for_each

main.tf

```
resource "local_file" "pet" {
    filename = each.value
    for_each = var.filename
}
```

variables.tf

```
variable "filename" {
  type=list(string)
    default = [
      "/root/pets.txt",
      "/root/dogs.txt",
      "/root/cats.txt"
    ]
}
```

>_

```
$ terraform plan
Error: Invalid for_each argument

  on main.tf line 2, in resource "local_file" "pet":
   2:   for_each = var.filename

The given "for_each" argument value is unsuitable: the "for_each"
argument must be a map, or set of strings, and you have provided a value
of type list of string.
```

for_each

main.tf

```
resource "local_file" "pet" {
    filename = each.value

    for_each = var.filename

}
```

variables.tf

```
variable "filename" {
    type=set(string)
    default = [
      "/root/pets.txt",
      "/root/dogs.txt",
      "/root/cats.txt"
    ]

}
```

>_

```
$ terraform plan
Terraform will perform the following actions:
  # local_file.pet["/root/cats.txt"] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/cats.txt"
    }
... <output trimmed>
Plan: 3 to add, 0 to change, 0 to destroy.
```

for_each

main.tf

```
resource "local_file" "pet" {
    filename = each.value
    for_each = toset(var.filename)
}
```

variables.tf

```
variable "filename" {
  type=set(string)
  default = [
    "/root/pets.txt",
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

>_

```
$ terraform plan
Terraform will perform the following actions:
  # local_file.pet["/root/cats.txt"] will be created
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/cats.txt"
    }
... <output trimmed>
Plan: 3 to add, 0 to change, 0 to destroy.
```

for_each

main.tf

```
resource "local_file" "pet" {
    filename = each.value

    for_each = toset(var.filename)
}

output "pets" {
  value = local_file.pet
}
```

variables.tf

```
variable "filename" {
  type=list(string)

  default = [
    "/root/dogs.txt",
    "/root/cats.txt"
  ]
}
```

>_

```
$ terraform plan
Terraform will perform the following actions:
  # local_file.pet["/root/pets.txt"] will be destroyed
  + resource "local_file" "pet" {
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
    }
... <output trimmed>
Plan: 0 to add, 0 to change, 1 to destroy.
```

for_each

main.tf

```
resource "local_file" "pet" {
    filename = each.value

    for_each = toset(var.filename)
}

output "pets" {
  value = local_file.pet
}
```

>_

```
$ terraform output
pets = {
  "/root/cats.txt" = {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/cats.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  }


  "/root/dogs.txt" = {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/dogs.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  }
}
```

count

```
>_
$ terraform output
pets = [
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/pets.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/dogs.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
  {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/cats.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  },
]
```

for_each

```
>_
$ terraform output
pets = {
  "/root/cats.txt" = {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/cats.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  }


  "/root/dogs.txt" = {
    "directory_permission" = "0777"
    "file_permission" = "0777"
    "filename" = "/root/dogs.txt"
    "id" = "da39a3ee5e6b4b0d3255bfef95601890afd80709"
  }
}
```

HANDS-ON LABS

course on Terraform for the Absolute Beginners: htt ScZd

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

Version Constraints

main.tf

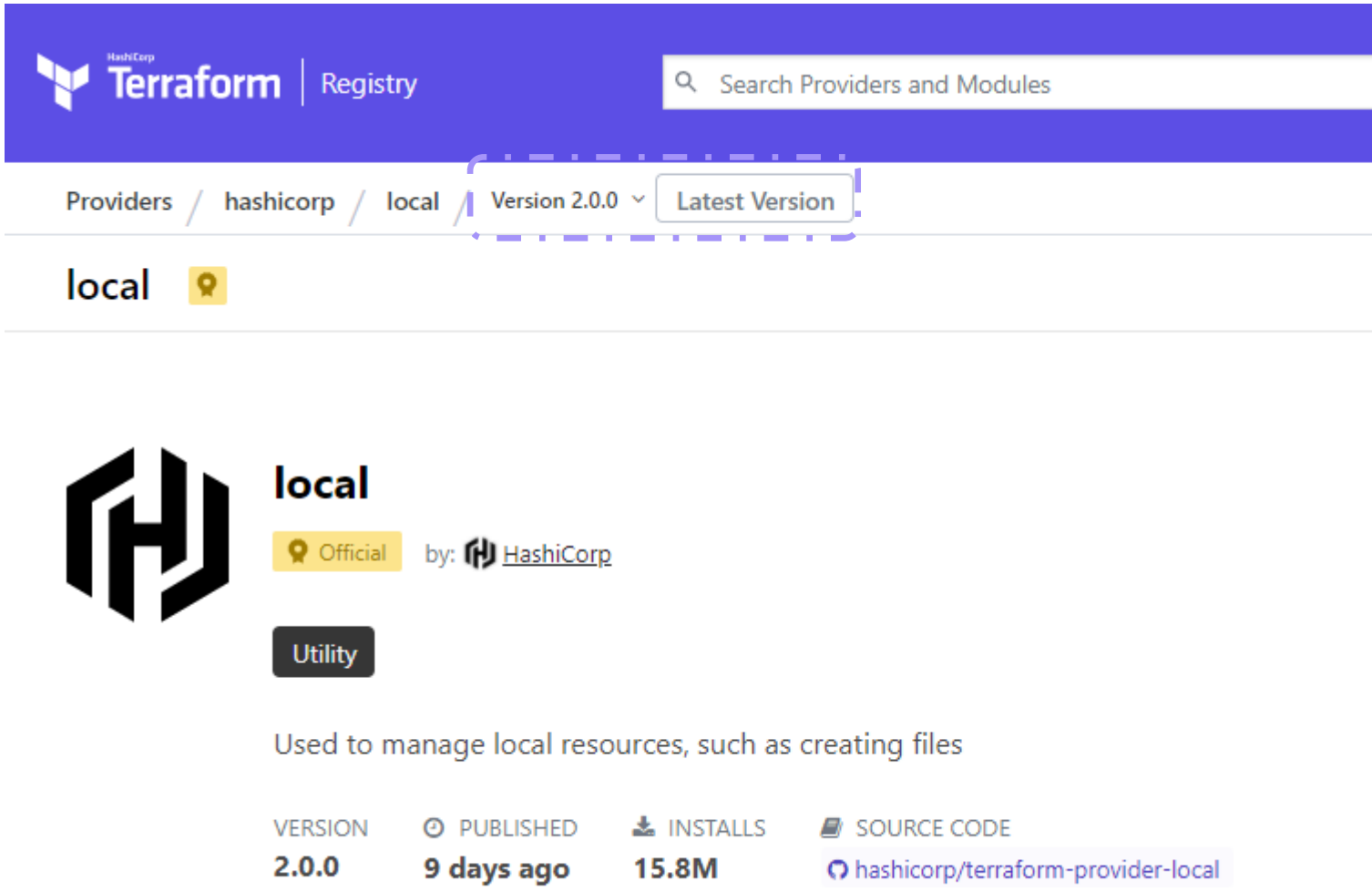
```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```

>_

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

The following providers do not have any version constraints
in configuration, so the latest version was installed.

To prevent automatic upgrades to new major versions that may
contain breaking
changes, we recommend adding version constraints in a
required_providers block
in your configuration, with the constraint strings suggested
below.

* hashicorp/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```

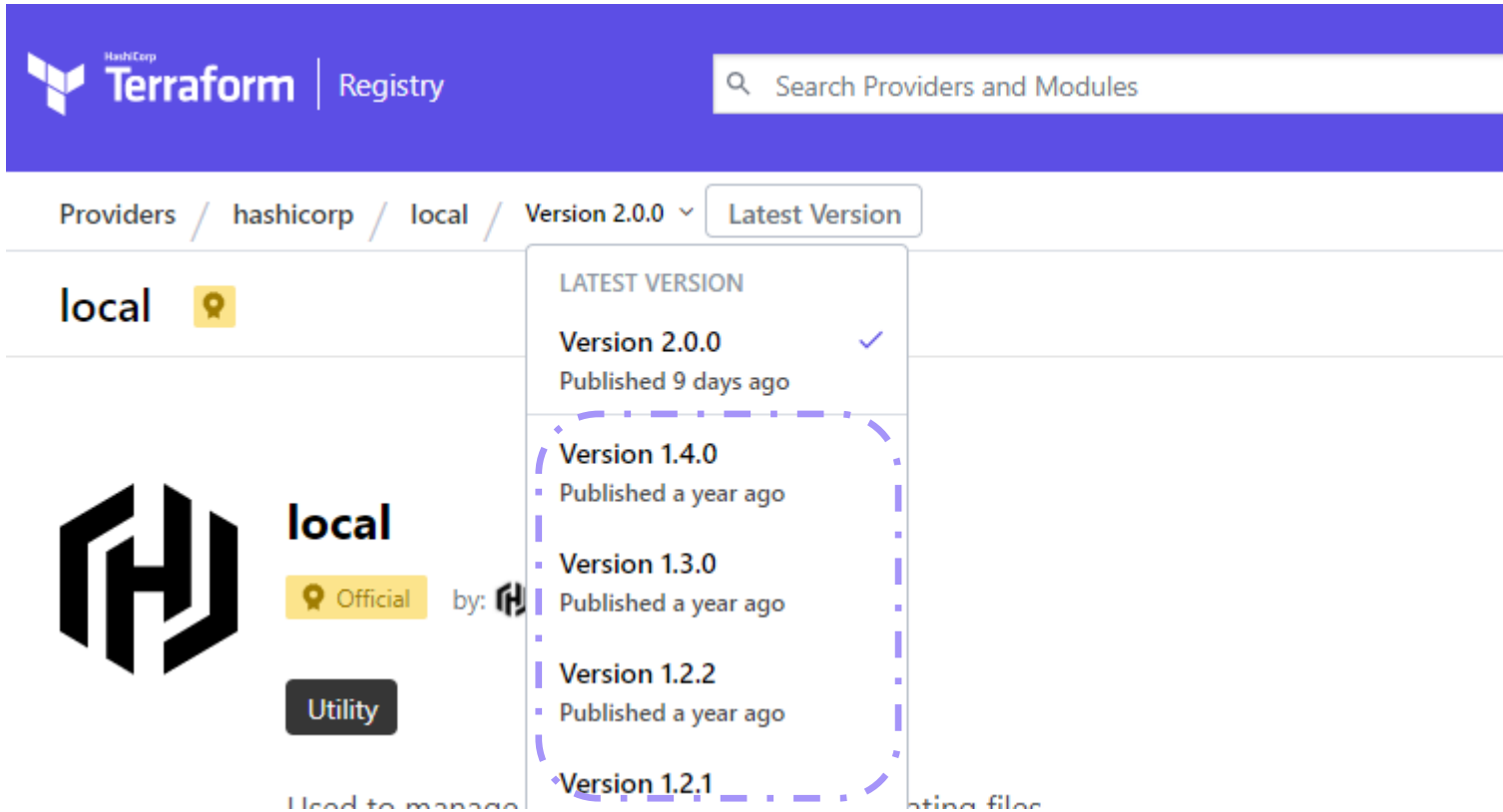
main.tf

```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



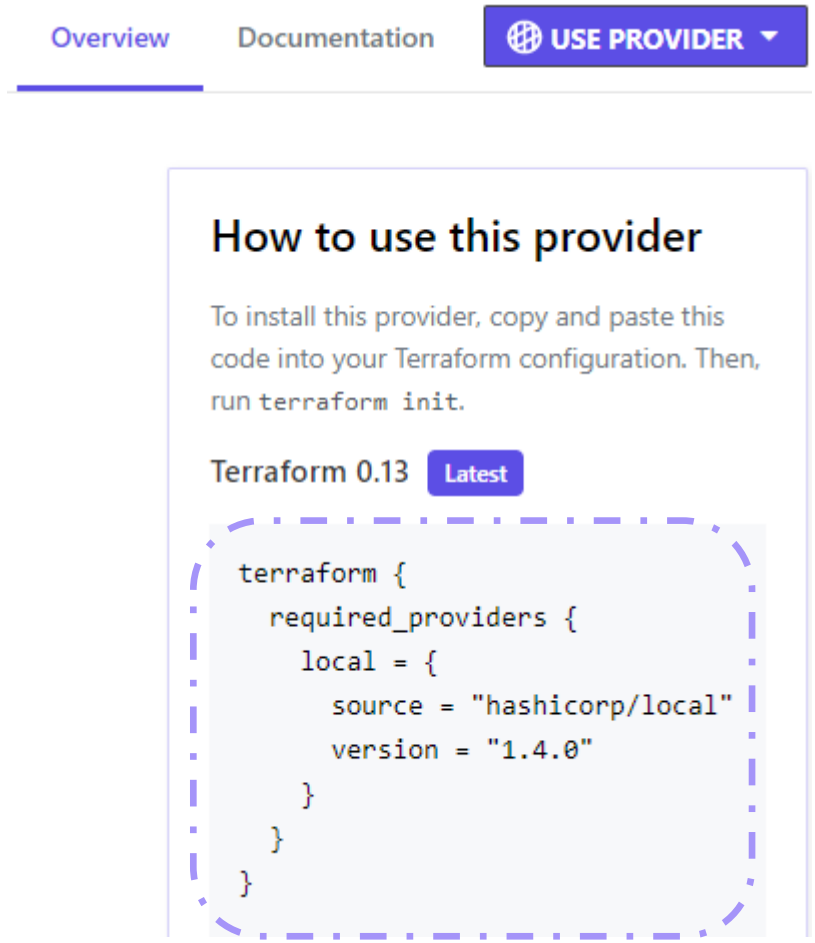
main.tf

```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



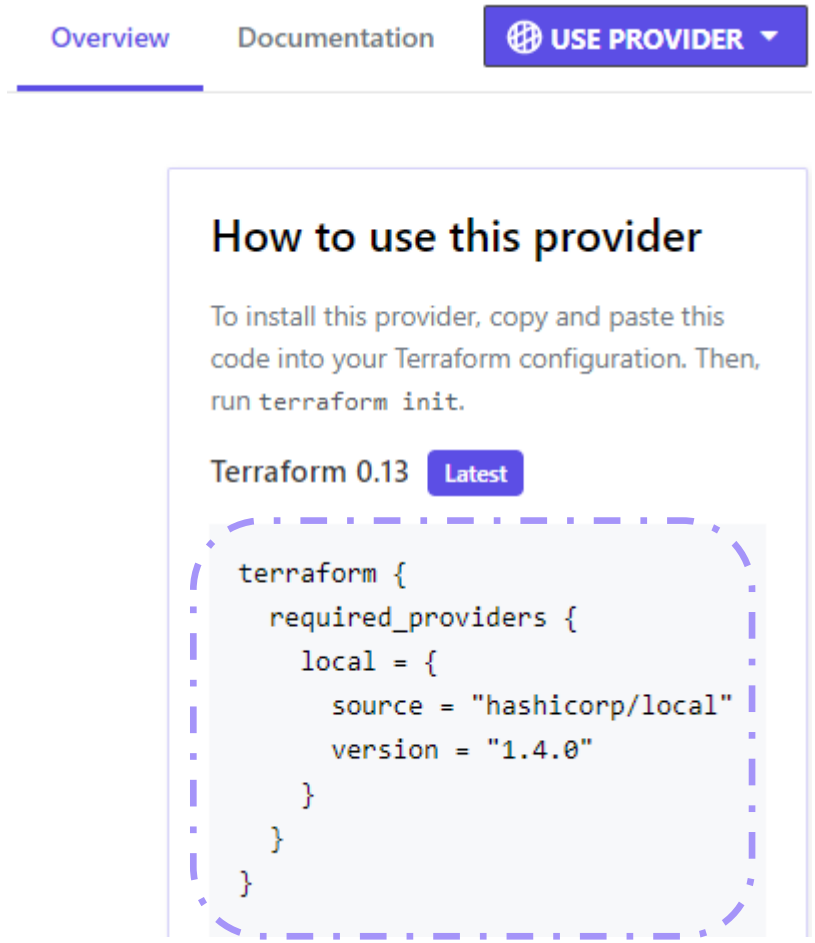
main.tf

```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```

>_

```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/local versions matching "1.4.0"...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running
"terraform plan" to see
any changes that are required for your infrastructure. All
Terraform commands
should now work.

If you ever set or change modules or backend configuration for
Terraform,
rerun this command to reinitialize your working directory. If
you forget, other
commands will detect it and remind you to do so if necessary.
```

main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "> 1.2.0, < 2.0.0, != 1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```

>_

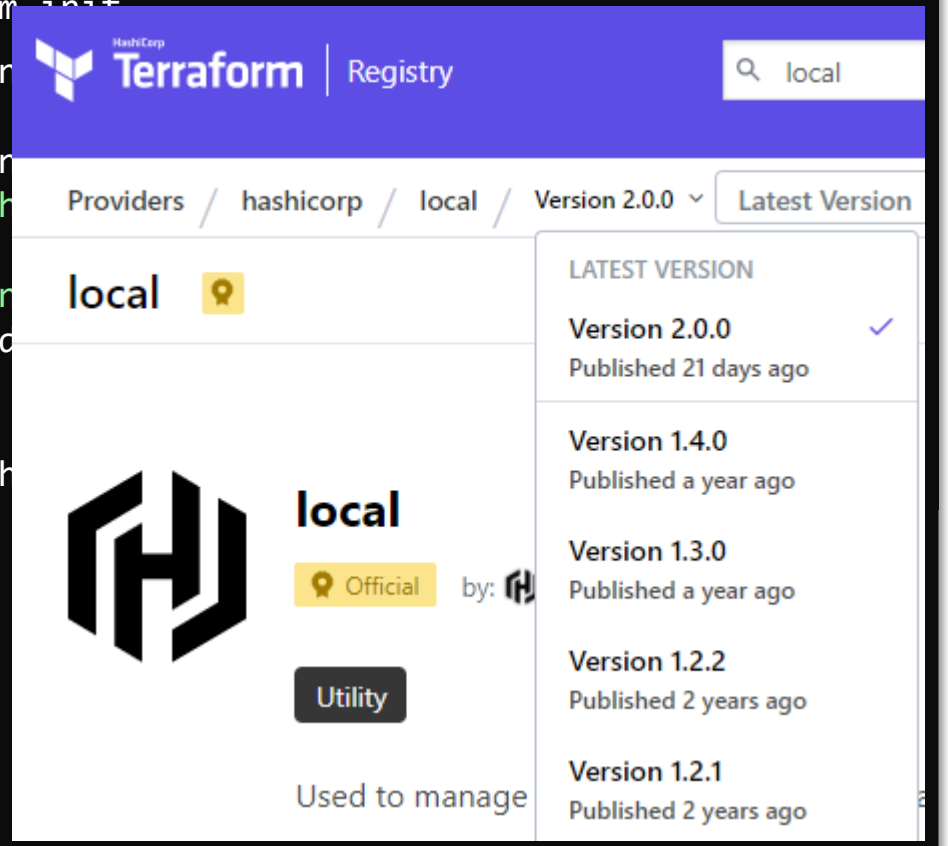
```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/local versions matching "> 1.2.0, < 2.0.0, != 1.4.0"...
- Installing hashicorp/local v1.3.0...
- Installed hashicorp/local v1.3.0 (signed by HashiCorp)

Terraform has been successfully initialized!
```

main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "~> 1.2.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```

>_

```
$ terraform init

Initializin

Initializin
- Finding h
1.2.0"...
- Installin
- Installed
HashiCorp)

Terraform h
```



Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd

KodeKloud

Check out our full course on Terraform for the Absolute Beginners: https://kode.wiki/3PoScZd