You can set TF_LOG to one of the log levels TRACE, DEBUG, INFO, WARN or ERROR to change the verbosity of the logs

```
terraform workspace new stage
```

```
terraform workspace select
```

```
terraform force-unlock     =>  remove the lock on the state
```

private registry

```
aws_instance.bryan-demo["vault"]
```

Terraform is available for macOS, FreeBSD, OpenBSD, Linux, Solaris, and Windows.

```
terraform import
```

```
.terraform/providers
```

```
terraform apply -replace
```

Terraform Cloud supports the following VCS providers as of March 2024:

- GitHub.com
- GitHub App for TFE
- GitHub.com (OAuth)
- GitHub Enterprise
- GitLab.com
- GitLab EE and CE
- Bitbucket Cloud
- Bitbucket Data Center
- Azure DevOps Server
- Azure DevOps Services

```
terraform plan
```

```
terraform apply -lock=false   => will instruct Terraform to not hold a state lock during the operation
```

```
terraform state mv  => don't want Terraform to replace the object after changing your configuration files.
```

```
terraform apply - refresh-only
```

```
terraform state rm
```

alias

```
terraform workspace select dev
```

```
terraform validate
```

terraform state show

terraform state pull

terraform state push

terraform state import

 terraform.tfstate.d

terraform plan -refresh-only => used in Terraform to update the state of your infrastructure in memory without making any actual changes to the infrastructure.

TF_VAR => prefix string for setting input variables using environment variables in Terraform is TF_VAR.(export TF_VAR_instructor_name="bryan")

terraform-<PROVIDER>-<NAME>, The requirement for release tag names to follow the x.y.z format and optionally be prefixed with a 'v' is valid

depends_on

Terraform supports variable types such as `string`, `bool`, and `number`, list (or tuple), map (or object)

terraform console

Names that are the same as Terraform reserved words, such as source, version, providers, count, for_each, lifecycle, depends_on, locals.

terraform get => used to download modules from the module registry or a version control system, making them available for use in the configuration

It's important to note that terraform init is typically run automatically when running other Terraform commands, so you may not need to run terraform get separately. However, if you need to update specific modules, running terraform get can be useful.

Vault provider => allows Terraform to read from, write to, and configure Hashicorp Vault.

export TF_LOG=TRACE

terraform show => used to inspect a plan to ensure that the planned operations are expected, or to inspect the current state as Terraform sees it.

dynamic block

The prefix "-/+" means that Terraform will destroy and recreate the resource. Some attributes and resources can be updated in-place and are shown with the "~" prefix.

required_providers

terraform apply -target=aws => apply changes to the AWS provider

terraform.tfstate

Sentinel

variables.tf

outputs.tf

locals

If the state has drifted from the last time Terraform ran,terraform plan -refresh-only or terraform apply -refresh-only allows drift to be detected.

terraform plan -out=myplan.tfplan

terraform fmt

.tfvars

terraform init -upgrade

terraform login

terraform fmt -recursive

Providers can be installed using multiple methods, including downloading from a Terraform public or private registry, the oƯicial HashiCorp releases page, a local plugins directory, or even from a plugin cache

terraform apply -destroy

terraform init -migrate-state

terraform workspace list

each.value.ip

terraform apply -refresh=false

terraform destroy, terraform apply -destroy

terraform state list

terraform version

```
terraform output

terraform destroy -auto-approve

TF_LOG=TRACE

TF_LOG_PATH="<file_path>"
```