General Usage

`$terraform --version`
Shows the terraform version

`$terraform init`
The first command to initialize a working directory and will install backend, modules and plugins

Format

Is used to rewrite Terraform configuration files to a canonical format and style.

`$terraform fmt`

Plan

The terraform plan command creates an execution plan.

`$terraform plan`
Creates an execution plan, useful to check infrastructure changes.

`$terraform plan --out infra-plan.out`
Creates an execution plan in a separate file, useful to check infrastructure changes.

`$terraform plan -target=azurerm_virtual_network.demo`
Only plan 1 target resource.

Apply

The apply command executes the actions proposed in a Terraform plan.

`$terraform apply`
Applies the infrastructure changes.

`$terraform apply infra-plan.out`
Applies the infrastructure changes set in the infra-plan.out file.

`$terraform apply -auto-approve`
Applies the infrastructure changes without prompting for 'YES'. **!use with caution!**

`$terraform apply -var myvar=test 123`
Define and use the variable "myvar".

`$terraform apply -target=azurerm_virtual_network.demo`
Only apply to 1 target resource.

Destroy

The terraform destroy command is a convenient way to destroy all remote objects managed by a particular Terraform configuration.

`$terraform plan -destroy`
Shows a deletion plan.

`$terraform destroy`
Deletes all resources defined in the terraform files. **!use with caution!**

`$terraform destroy -target=azurerm_virtual_network.demo`
Deletes only the targeted resource. **!use with caution!**

Validate

The validate command validates the configuration files and does not access any remote services such as remote state, provider APIs, etc.

`$terraform validate`

State

The terraform state command is used for advanced state management.

`$terraform state show azurerm_virtual_network.demo`
Shows the state of this defined resource.

`$terraform state pull > terraform.tfstate`
Pull's remote state into local file

`$terraform state mv azurerm_public_ip.demo-instance module.instances`
Moves existing tracted resources via state to a module.

`$terraform state list`
Shows all resources tracked in the state file.

`$terraform state rm`
Removes the tracking of a resource from the state file

`$terraform force-unlock LOCK_ID`
Removes lock from state file, the LOCK_ID will be provided beforehand

Graph

Generates a visual representation of either a configuration or execution plan

`$ terraform graph | dot - Tpng > tf_graph.png`

(Un)Taint

Informs Terraform that a particular object has become degraded or damaged. On the next plan/apply the resource will be recreated.

```
$terraform taint azurerm_virtual_network.demo
```
Taints the resource

```
$terraform untaint azurerm_virtual_network.demo
```
Removes the taint from the resource

Workspace

Separate instances of state data that can be used from the same working directory.

```
$terraform workspace new myudemyworkspacedev
```
Creates new workspace

```
$terraform workspace select myotherworkspace
```
Changes to defined workspace

```
$terraform workspace list
```
Show all workspaces

Autocomplete

Terraform can provide tab-completion support for bash or zsh

```
$terraform -install-autocomplete
```
Creates new workspace

Console

Provides an interactive console for evaluating expressions.

```
$echo '1+2'| terraform console
```
Executes 1+2 in the console, useful for testing expressions.

```
$echo 'join(",",["tf","udemy"])' | terraform console
```
Joins a list of strings into 1 string

Import

Import existing resources into your Terraform state.

```
$terraform import azurerm_virtual_network.demo
/subscriptions/00000000-0000-0000-0000-000000000000/reso
urceGroups/mygroup1/providers/Microsoft.Network/virtualN
etworks/demovnet1
```
Imports a resource into the defined resource azurerm_virtual_network.

Output

Used to extract the value of an output variable from the state file.

```
$terraform output vm_ips
```
Shows the state of this defined resource.

```
$terraform output -json
```
Output all resources defined in the root module in json format.

Typical Terraform lifecycle: