OpenTofu Open-source infrastructure as code

CONCEPTS

OpenTofu is the open-source fork of Terraform: same HCL and workflow, command is tofu. Configuration and state stay compatible for typical setups.

Config .tf files
↓ `init`
Working dir providers, modules
↓ `plan`
Plan diff vs. state
↓ `apply`
Infrastructure state updated

Always read the plan before applying. Saving it with -out guarantees apply does exactly what you reviewed.

State may hold secrets. Keep it out of git, use a remote backend, and consider OpenTofu's built-in state encryption.

WORKFLOW

`tofu init`
Prepare the working directory

`tofu init -upgrade`
Update providers and modules

`tofu init -reconfigure`
Ignore saved backend setup

`tofu validate`
Check configuration syntax

`tofu plan`
Preview changes

`tofu plan -out=<file>`
Save the plan to a file

`tofu apply`
Create or update infrastructure

`tofu apply <planfile>`
Apply a saved plan

`tofu apply -auto-approve`
Skip the confirmation prompt

`tofu destroy`
Remove all managed resources

`tofu plan -destroy`
Preview a destroy

`tofu plan -refresh-only`
Preview drift against reality

WORKFLOW (cont.)

`tofu fmt`
Format files in place

`tofu fmt -recursive -check`
Check formatting in CI

TARGETING & PLANS

`tofu plan -target=<addr>`
Limit to one resource

`tofu apply -replace=<addr>`
Force recreating a resource

`tofu plan -detailed-exitcode`
Exit 2 when changes exist

`tofu show <planfile>`
Read a saved plan

VARIABLES & OUTPUTS

`tofu plan -var '<name>=<value>'`
Set one variable

`tofu plan -var-file=<file>`
Load variables from a file

`TF_VAR_<name>=<value>`
Set a variable via environment

`terraform.tfvars`
Auto-loaded variable file

`tofu output`
Show all outputs

`tofu output <name>`
Show one output

`tofu output -raw <name>`
Plain value for scripts

`tofu output -json`
All outputs as JSON

`tofu console`
Try expressions interactively

STATE

`tofu state list`
List tracked resources

`tofu state show <addr>`
Show one resource's attributes

`tofu state mv <src> <dst>`
Rename or move in state

`tofu state rm <addr>`
Forget a resource, keep the real one

STATE (cont.)

`tofu import <addr> <id>`
Adopt an existing resource

WORKSPACES

`tofu workspace list`
List workspaces

`tofu workspace show`
Show the current workspace

`tofu workspace new <name>`
Create and switch

`tofu workspace select <name>`
Switch workspace

Each workspace has its own state under the same configuration; it is not a strong isolation boundary.

TESTING & MODULES

`tofu test`
Run *.tftest.hcl test files

`tofu test -filter=<file>`
Run one test file

`tofu providers`
Show required providers

`tofu version`
Show the OpenTofu version

`tofu -chdir=<dir> <cmd>`
Run in another directory

RECIPES

Reviewed deploy

```
tofu init
tofu fmt -check && tofu validate
tofu plan -out=tfplan
tofu show tfplan
tofu apply tfplan
```

Per-environment variables

```
tofu workspace select prod
tofu plan -var-file=prod.tfvars -
out=prod.plan
tofu apply prod.plan
```

Rename without recreating

```
tofu state mv aws_instance.old
aws_instance.new
tofu plan   # expect no changes
```

Replace one broken resource

```
tofu plan -
replace=aws_instance.web -
out=fix.plan
tofu apply fix.plan
```