Packer

Machine image builder

CONCEPTS

A template is HCL files: a source says what to boot, a build runs provisioners on it and saves an image.

Template *.pkr.hcl
↓ `packer init`
Plugins builders, provisioners
↓ `packer build`
Temp machine source + provisioners
↓ `post-processors`
Image AMI, box, template

Point commands at a directory (.) to load every *.pkr.hcl in it, not just one file.

INIT & PLUGINS

`packer init` *<dir>*
Install plugins the template requires

`packer init -upgrade` *<dir>*
Upgrade plugins within constraints

`packer plugins installed`
List installed plugin binaries

CHECK & FORMAT

`packer validate` *<dir>*
Check syntax and configuration

`packer validate -syntax-only` *<dir>*
Check syntax only

`packer fmt` *<dir>*
Rewrite files in canonical style

`packer fmt -check` *<dir>*
Fail if unformatted (CI)

`packer fmt -diff` *<dir>*
Show formatting changes

`packer inspect` *<dir>*
List variables, sources, provisioners

BUILD

`packer build` *<dir>*
Build all sources

`packer build` *<file.pkr.hcl>*
Build from a single file

`packer build -only=`*<a>*`,`*<b>* *<dir>*
Run only the named builds

VARIABLES

`packer build -var` *<key>*`=`*<value>* *<dir>*
Set one variable

`packer build -var-file=`*<file>* *<dir>*
Load variables from a file

`PKR_VAR_`*<name>*`=`*<value>*
Set a variable via environment

`*.auto.pkrvars.hcl`
Variable file loaded automatically

Later wins: environment, then auto files, then -var and -var-file in the order given.

DEBUG & ERRORS

`packer build -on-error=ask` *<dir>*
Prompt to clean up, abort or retry

`packer build -debug` *<dir>*
Step by step, pauses between steps

`PACKER_LOG=1`
Verbose logs on stderr

TEMPLATE SKELETON

Plugins and a variable

```
packer {
  required_plugins {
    docker = {
      version = "≥ 1.0.0"
      source  = "github.com/hashicorp/
docker"
    }
  }
}

variable "tag" {
  type    = string
  default = "latest"
}
```

Source and build

```
source "docker" "app" {
  image  = "ubuntu:24.04"
  commit = true
}

build {
  sources = ["source.docker.app"]
  provisioner "shell" {
    inline = ["apt-get update"]
  }
}
```