Terraform Cheat Sheet

Key Concepts

- Infrastructure as Code (IaC): Terraform allows defining infrastructure using a declarative language.
- Providers: Plugins that interact with APIs to manage infrastructure (e.g., AWS, Azure).
- Resources: Components of your infrastructure (e.g., EC2 instances, S3 buckets).
- Modules: Reusable units of Terraform configurations.
- State: Terraform maintains a state file to track infrastructure deployed.
- Execution Plan: A preview of changes Terraform will apply.
- Workspaces: Allows you to manage different states (e.g., for different environments).

Main Files

1. Main Configuration (main.tf): Core file where resources are defined.
2. Variables (variables.tf): Declare input variables.
3. Output (outputs.tf): Define outputs after execution.
4. Terraform State (terraform.tfstate): Stores information about the infrastructure.
5. Provider Configuration: In provider.tf to configure cloud providers.

Syntax Overview

- Resources:

```
  resource "aws_instance" "example" {
    ami = "ami-12345678"
    instance_type = "t2.micro"
  }
```

- Variables:

```
  variable "instance_type" {
    description = "Type of instance"
    default = "t2.micro"
  }
```

- Outputs:

```
  output "instance_ip" {
    value = aws_instance.example.public_ip
  }
```

- Providers:

```
  provider "aws" {
    region = "us-west-2"
  }
```

Basic Commands

- Initialization: terraform init
- Format Code: terraform fmt
- Validate Configuration: terraform validate
- Plan: terraform plan
- Apply: terraform apply
- Destroy Resources: terraform destroy
- Show State: terraform show

Variables and Data Types

- Basic Types: string, number, bool
- Declaring a Variable:

```
  variable "example" {
    type = string

    description = "Example variable"

    default = "default_value"

  }
```

- Using Variables:

```
  resource "aws_instance" "example" {

    instance_type = var.instance_type

  }
```

State Management

- View Current State: terraform state list

- Remove Resource from State (without destroying it): terraform state rm <resource_name>

Terraform Cloud/Backend

- Configure Backend (e.g., S3):

```
  terraform {

    backend "s3" {

      bucket = "my-terraform-state"

      key = "path/to/my/key"

      region = "us-west-2"

    }

  }
```

Best Practices

- Use Modules: Reuse infrastructure configurations.

- Store state remotely: Helps with collaboration.

- Version Control: Use Git to track Terraform files.

- Keep State Secure: Protect terraform.tfstate as it contains sensitive info.