Terraform - Concepts

# Introduction

Terraform is a tool for building, changing & versioning infrastructure

It uses **configuration files**.

# Terraform features

## Infrastructure as Code

Blueprint of infrastructure can be treat as code

## Execution plan

Terraform generates execution plan, for apply them.

## Resource Graph

Terraform builds resource graph

## Change automation

--

# Compare Terraform with other Configuration Management Tools

|  |  |  |
| --- | --- | --- |
|  | Terraform | Configuration Management Tools (Chef/Puppet) |
| Infrastructure automation | Yes | Limited |
| VM & Cloud Provisioning | Yes | - |
| Declarative like | Yes | Yes |
| OS Configuration Management | Limited | Yes |
| Application Installation | - | Yes |

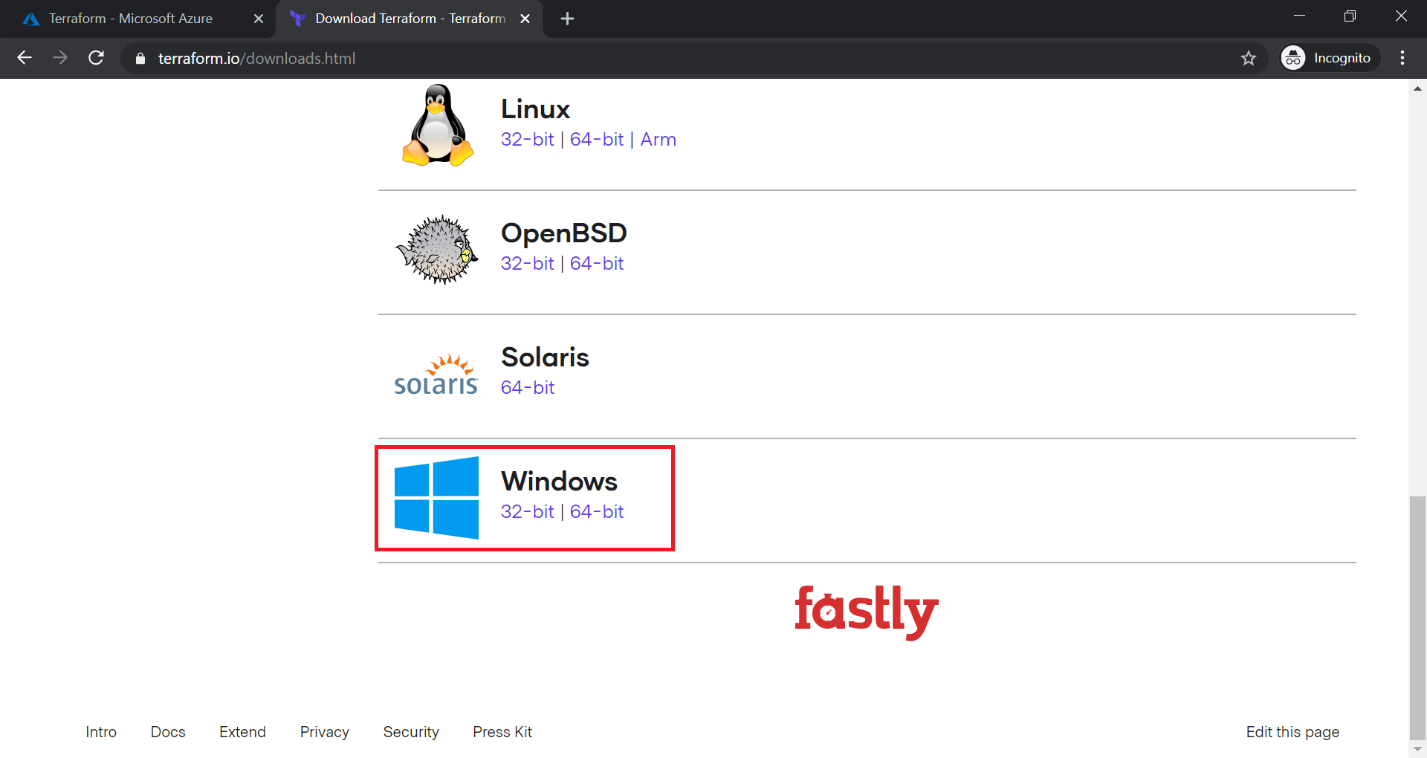
Terraform creates infrastructure like network/storage/virtual machines and hand off to Chef/puppet for application installation or IIS configuration.

# Terraform use cases

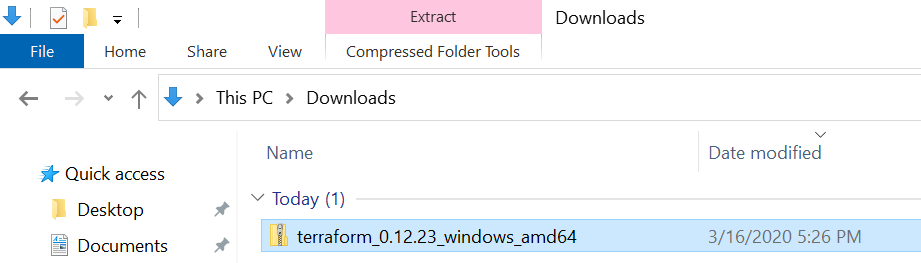
Multi cloud deployment

# Download, Install & Configure Terraform

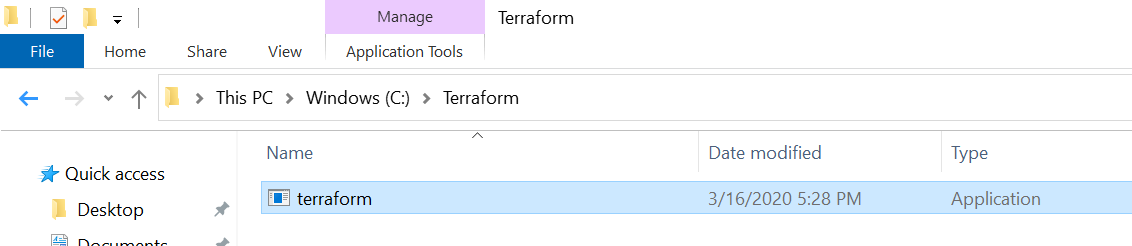
1. Download CLI from <https://www.terraform.io/>



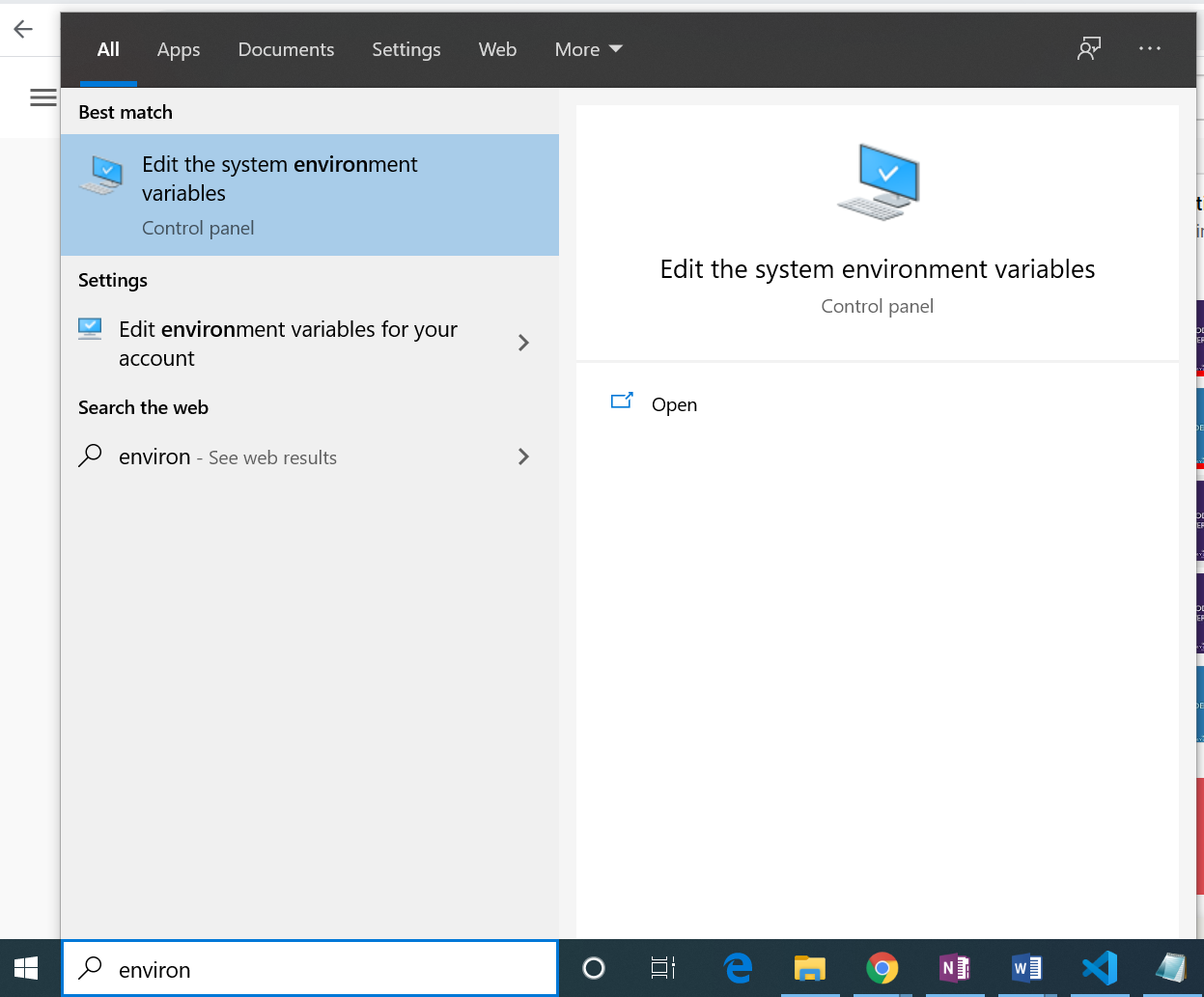
1. Select 64-Bit, this downloads a zip file.

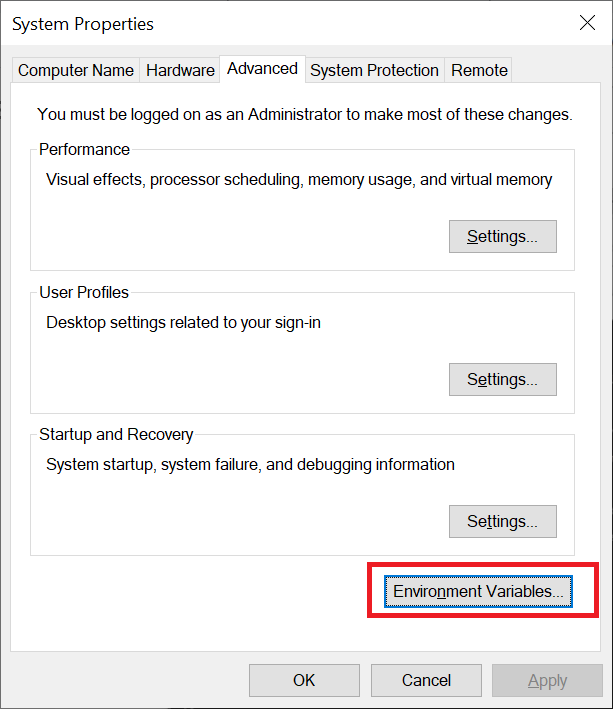


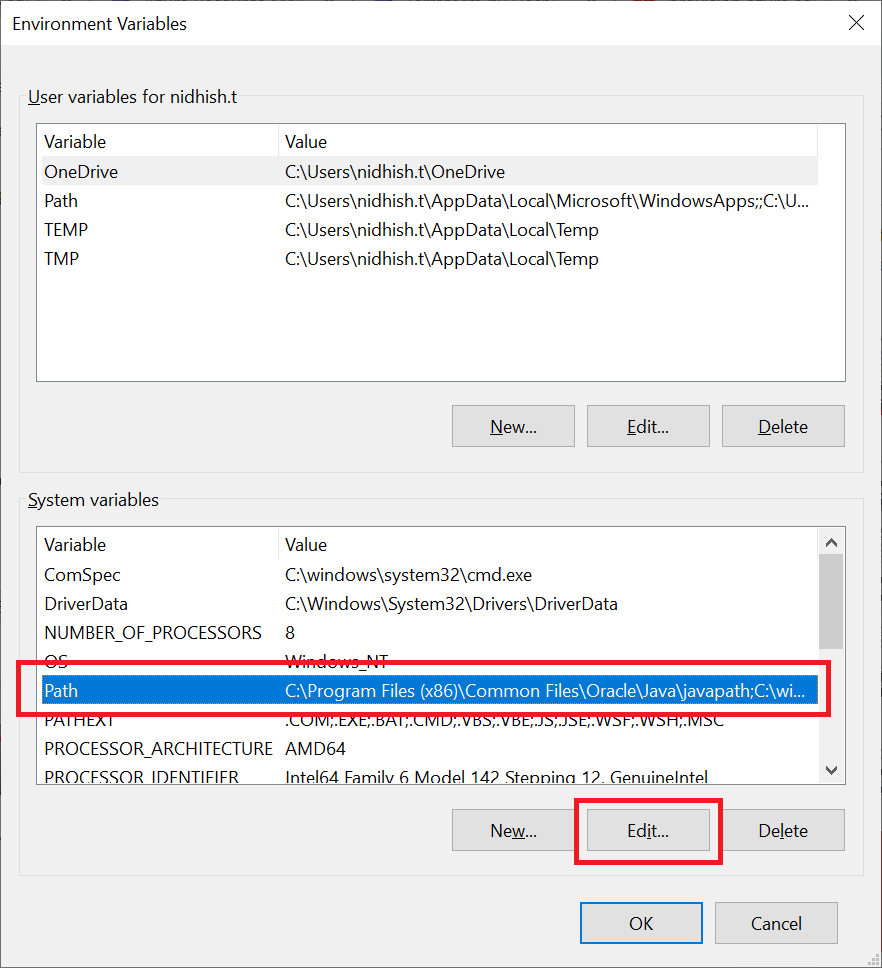
1. Open zip & copy exe to c:\terraform folder

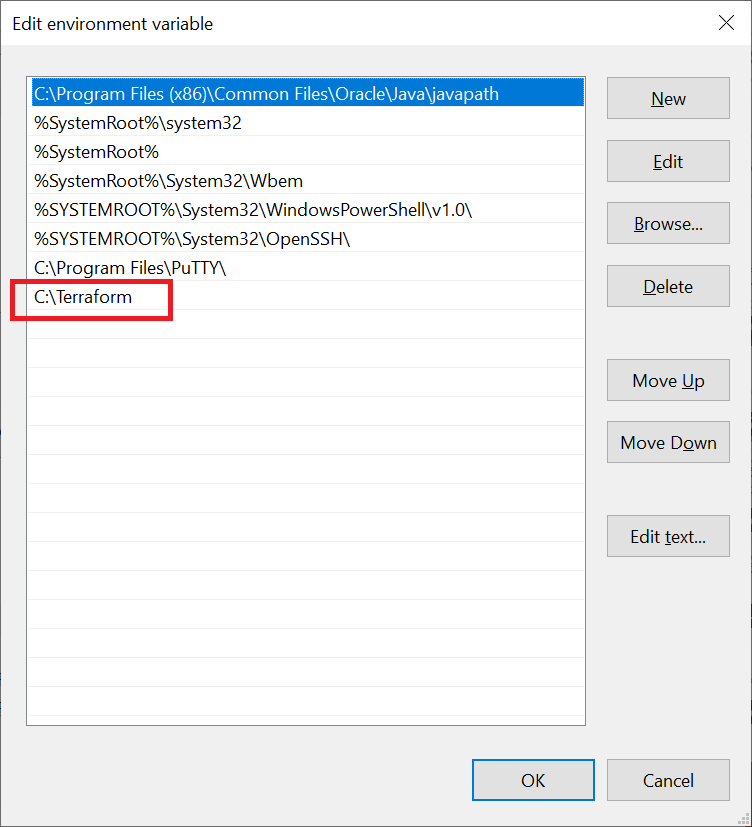


1. Configure **Environment variables**









1. Done

# Terraform constructs and execution

1. Providers
2. Resources
3. Provisioners

## Providers

**Example**: Azure, AWS, Docker, Github etc

<https://www.terraform.io/docs/providers/index.html>

## Resources

Example:

resource "azurerm\_resource\_group" "example"

resource = **component**

azurerm = **provider**

resource\_group = **Type**

“example” = **name**

## Provisioners

Provisioners can execute to initialize the resource

# Terraform Execution

1. Plan
2. Apply
3. Destroy

# Reference:

[Skyline youtube](https://www.youtube.com/watch?v=cDjYRmKYOCc&list=PLD7svyKaquTlE9dErhMazFhWbSSCfMP_4&index=2)