<epam>

# IaC vs AppCode and Terraform Optimisations

Bogdan Dumitrescu

</>DevCon - November 2022


EPAM is **a leading global provider of software product development and digital platform engineering services** to hundreds of Fortune 500 and 1000 clients located around the world, primarily in North America, Europe, Asia and Australia

I’ve worked as a SysAdmin for about 14 years, and for the last 5 years, he acted as a Senior DevOPS Engineer

Main industries like Telecom, Delivery, Financial/Banking, Insurance, Real Estate and HORECA.

During this time, I’ve been working with Cloud Infrastructures (mainly AWS) as well as On-Premise solutions.

# Agenda

**01** Programing Language Categories

**02** HCL

**03** Terraform

**04** Terraform Examples

**05** Terragrunt

**06** Terragrunt Example

**07** Q&A

# Programing Language Categories

Functional

Object-Oriented

Interpreted

Compiled

Imperative

Declarative

Procedural

Scripting

Logic

Front-End

Back-End

High Level

Low Level

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

**3 Functional**

5 Object-Oriented

1 Interpreted

7 Compiled

5 Imperative

3 Declarative

Functional languages focus on the output of mathematical functions and evaluations.

Each function, considered a reusable piece of code, performs a specific task and returns a result

- Scala
- Erlang
- Haskell
- Elixir
- F#

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

3 Functional

**5 Object-Oriented**

1 Interpreted

7 Compiled

5 Imperative

3 Declarative

This type of language treats a program as a group of objects composed of data and program elements, known as attributes and methods.

Objects can be reused within a program or in other programs.

- Java
- Python
- PHP
- C++
- Ruby

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

3 Functional

5 Object-Oriented

**1 Interpreted**

7 Compiled

5 Imperative

3 Declarative

With these languages, code goes through a program called an interpreter, which reads and executes the code line by line.

This tends to make these languages more flexible and platform independent.

- Python
- JavaScript
- PHP
- Ruby

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

3 Functional

5 Object-Oriented

1 Interpreted

**7 Compiled**

5 Imperative

3 Declarative

These languages go through a build step where the entire program is converted into machine code.

This makes it faster to execute, but it also means that you have to compile or "build" the program again anytime you need to make a change.

- C, C++, and C#
- Rust
- Erlang

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

3 Functional

5 Object-Oriented

1 Interpreted

7 Compiled

**5 Imperative**

3 Declarative

**Imperative programming** is a paradigm describing HOW the program should do something by explicitly specifying each instruction (or statement) step by step, which mutate the program's state.

- Fortran
- C, C++
- Cobol

Source: https://www.coursera.org/articles/types-programming-language

# Programing Language Categories

3 Functional

5 Object-Oriented

1 Interpreted

7 Compiled

5 Imperative

**3 Declarative**

**Declarative programming** is a paradigm describing WHAT the program does, without explicitly specifying its control flow.

Functional languages are characterized by a declarative programming style.

- Haskell
- Scheme
- ML

Source: https://www.coursera.org/articles/types-programming-language

# Then, what’s this?

```
resource "aws_vpc" "main" {
  cidr_block = var.base_cidr_block
}

<BLOCK TYPE> "<BLOCK LABEL>" "<BLOCK LABEL>" {
  # Block body
  <IDENTIFIER> = <EXPRESSION> # Argument
}
```

```
resource "aws_vpc" "main" {
  cidr_block = var.base_cidr_block
}

<BLOCK TYPE> "<BLOCK LABEL>" "<BLOCK LABEL>" {
  # Block body
  <IDENTIFIER> = <EXPRESSION> # Argument
}
```

# HCL

HCL is a system for defining configuration languages for applications.

… i[t’]s optimized for human authoring and maintenance, as opposed to machine generation of configuration.

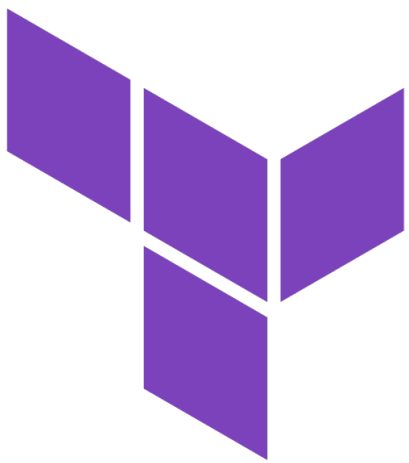
# HCL – Hashicorp Configuration Language

The language consists of **three** integrated sub-languages:

The structural language defines the overall hierarchical configuration structure, and is a serialization of HCL bodies, blocks and attributes.

The expression language is used to express attribute values, either as literals or as derivations of other values.

The template language is used to compose values together into strings, as one of several types of expression in the expression language.



HashiCorp

# Terraform



# Terraform

Terraform codifies cloud APIs into declarative configuration files.

- Infrastructure as code
- Multi-cloud deployment (AWS, GCP, AZURE)
- Enforce policy as code
- Manage Kubernetes
- Manage network infrastructure
- Manage virtual machine images
- Integrate with existing workflows
- Supports secrets injection

Source: https://www.terraform.io

# Terraform

Terraform codifies cloud APIs into declarative configuration files.

**Infrastructure as code**

**Multi-cloud deployment (AWS, GCP, AZURE)**

**Enforce policy as code**

Manage Kubernetes

Manage network infrastructure

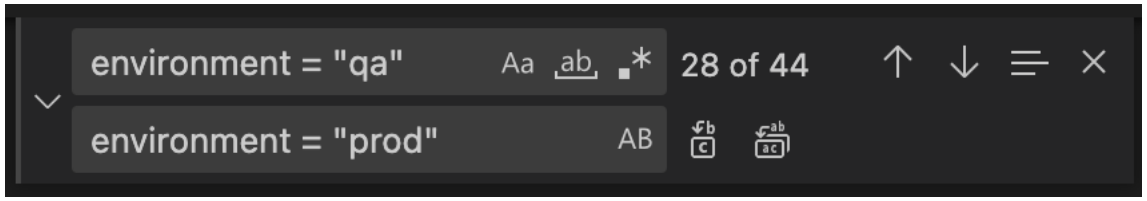
Manage virtual machine images

Integrate with existing workflows

Supports secrets injection

Source: https://www.terraform.io

# Terraform




environment =”qa”

environment = ”Qa”

environment = ”QA”

site_environment = ”qa”

environment_id = ”qa”

environment_name = ”qa”

Gruntwork-io

# Terragrunt

# Terragrunt

Terragrunt is a thin wrapper that provides extra tools for keeping your configurations DRY, working with multiple Terraform modules, and managing remote state.

- Keep your Terraform code DRY
- Keep your remote state configuration DRY
- Keep your Terragrunt Architecture DRY
- Keep your CLI flags DRY
- Execute Terraform commands on multiple modules at once
- Work with multiple AWS accounts
- Before, After, and Error Hooks
- Auto-init
- Auto-retry
- Caching
- AWS Auth
- Debugging
- Lock File Handling

Source: https://terragrunt.gruntwork.io

# Terragrunt

Terragrunt is a thin wrapper that provides extra tools for keeping your configurations DRY, working with multiple Terraform modules, and managing remote state.

**Keep your Terraform code DRY**

**Keep your remote state configuration DRY**

**Keep your Terragrunt Architecture DRY**

**Keep your CLI flags DRY**

Execute Terraform commands on multiple modules at once

Work with multiple AWS accounts

Before, After, and Error Hooks

Auto-init

Auto-retry

Caching

AWS Auth

Debugging

Lock File Handling

Source: https://terragrunt.gruntwork.io

# Q&A!

Demo Repo to play with:

**https://github.com/ebogdum/devcon-demo**

# Thank you!

**Bogdan Dumitrescu**
Team Lead DevOPS

bogdan_dumitrescu@epam.com
www.epam.com