

# Инфраструктура как код (IaC)

Обеспечивание, контроль и даже удаление компонентов инфраструктуры являются затратными процессами в плане человеческих ресурсов. К тому же, ручные попытки создания и модификации сред могут содержать риск ошибок. Вне зависимости от того, опирается ли администратор на предыдущий опыт или детальное руководство, возможность ошибки велика.

С помощью Infrastructure as Code, мы имеем возможность автоматизировать процесс создания полнофункциональной среды. Автоматизация может облегчить выполнение рутинных задач и приносить значительную выгоду. С использованием IaC мы можем формализовать нашу инфраструктуру через шаблоны. Один такой шаблон может представлять собой часть или всю среду. Ещё важнее, что такой шаблон может быть повторно использован для создания идентичных сред.

В модели IaC инфраструктура конструируется и управляется через код и CI (Continuous Integration). Эта модель позволяет интерактивно работать с инфраструктурой, минимизировать человеческие ошибки и автоматизировать конфигурирование ресурсов. Так, инфраструктура становится управляемой, как и код, с использованием кодовых инструментов. Благодаря управлению инфраструктурой кодом, приложения могут быть развернуты единообразно, а исправления и версии могут быть обновлены множество раз без ошибок. Ansible, Terraform и AWS CloudFormation — это некоторые из наиболее распространенных инструментов для написания сценариев IaC.

Роль архитектора решений в этом контексте включает в себя проектирование решений для автоматизации и управления инфраструктурой с использованием практик IaC, определение требований и стандартов для шаблонов и скриптов, а также обеспечение совместимости с инструментами и процессами CI/CD. Он также активно участвует в улучшении процессов автоматизации, с целью устранения вручную выполняемых задач и минимизации риска ошибок в инфраструктуре.

## Пример применения IaC

Может быть организован на основе инструмента Terraform, который является популярным инструментом для реализации IaC. Предположим, у нас есть веб-приложение, которое мы хотим развернуть на AWS. В данном случае, мы создаем код для инфраструктуры, который описывает все необходимые ресурсы, такие как виртуальные машины (EC2), базы данных (RDS) и сетевые конфигурации (VPC, Subnets).

# 1. Создание файлов конфигурации Terraform:

Пример файла main.tf может выглядеть следующим образом:

```
provider "aws" {  
  region = "us-west-2"  
}  
  
resource "aws_instance" "my_instance" {  
  ami           = "ami-0c55b159cbfaffe1f0"  
  instance_type = "t2.micro"  
  
  tags = {  
    Name = "MyInstance"  
  }  
}  
  
resource "aws_db_instance" "my_db" {  
  allocated_storage = 20  
  engine             = "mysql"  
  engine_version     = "5.7"  
  instance_class     = "db.t2.micro"  
  name               = "mydb"  
  username           = "admin"  
  password           = "adminpassword"  
  parameter_group_name = "default.mysql5.7"  
}  
  
resource "aws_vpc" "my_vpc" {  
  cidr_block = "10.0.0.0/16"  
  enable_dns_support = true  
  tags = {  
    Name = "my_vpc"  
  }  
}
```

## 2. Инициализация и Применение конфигурации:

После создания файла конфигурации, нужно выполнить следующие команды:

```
terraform init # Инициализация Terraform, загрузка провайдера AWS  
terraform apply # Применение конфигурации, создание ресурсов на AWS
```

### 3. Поддержка и изменения:

В будущем, если потребуется внести изменения в инфраструктуру, можно просто модифицировать файлы конфигурации Terraform и затем применить изменения, используя `terraform apply`.

### 4. Уничтожение ресурсов:

Когда ресурсы больше не нужны, они могут быть удалены с помощью команды:

```
terraform destroy # Удаление всех ресурсов, созданных Terraform
```

Этот пример демонстрирует, как IaC позволяет полностью автоматизировать процесс создания, изменения и уничтожения инфраструктуры, минимизируя возможность человеческой ошибки и ускоряя развертывание ресурсов.