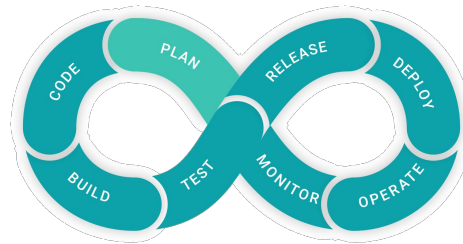



DevOps - IAC



Autores: João Frois e Lucas Padrão

Professor: Johnatan Alves de Oliveira

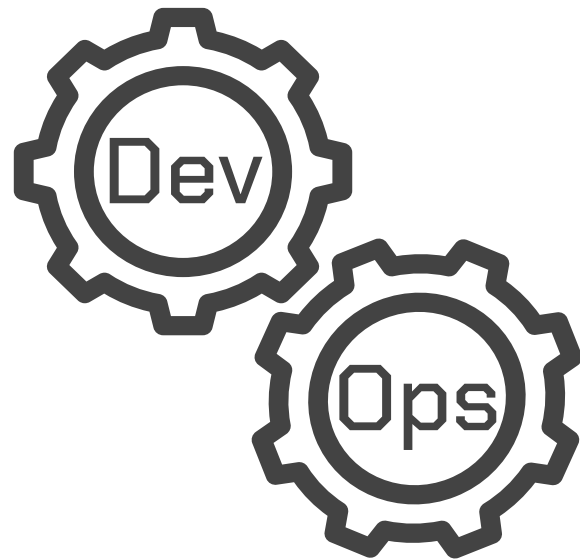


“DevOps não se trata de automação, assim como astronomia não se trata de telescópios” - **Christopher Little**

DevOps

Na sua essência, o DevOps é uma cultura, um movimento, uma filosofia. É um aperto de mão entre desenvolvimento e operações que enfatiza uma mudança de mentalidade, melhor colaboração e integração mais rígida.

Une entrega ágil e contínua para ajudar as equipes de desenvolvimento e operações a serem mais eficientes, inovadoras e fornecerem maior valor a empresas e clientes.



Pilares

Cultura

- Foco nas pessoas;
- Melhorar comportamentos;
- Abraçar mudanças e experimentações;

Integrar Dev + Ops



Automação

- Fazer pequenas e constantes entregas;
- Fazer pequenas e constantes entregas;
- Estruturas centralizadas de Testes e Deploy;

Infraestrutura como código



Fluxo

- Entregar valor para o usuário final;
- Identificar o fluxo de valor;
- Eliminar gastos;

Perseguir a melhoria contínua e fazer o simples





IaC - Infrastructure as Code

Gerenciamento e provisionamento da infraestrutura necessária a um software por meio de ferramentas que possibilitem codificá-la ao invés de depender de processos manuais

Por meio de arquivos de configuração, que especificam os recursos que a compõem, há uma maior facilidade em modificar aspectos da infraestrutura.

Promover maior escalabilidade e controle

Ferramentas





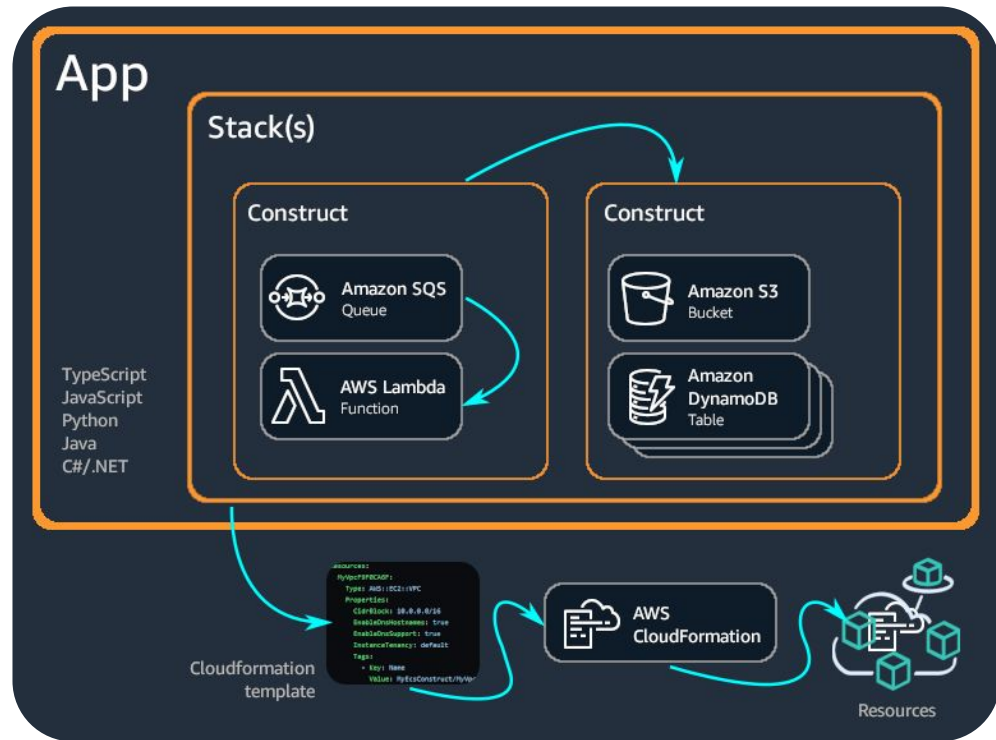
Cloud Development Kit

Autores: João Frois e Lucas Padrão

Professor: Johnatan Alves de Oliveira

O que é ?

É um framework de desenvolvimento de software de código aberto usado para definir a infraestrutura de nuvem como código e provisioná-la por meio do AWS CloudFormation.

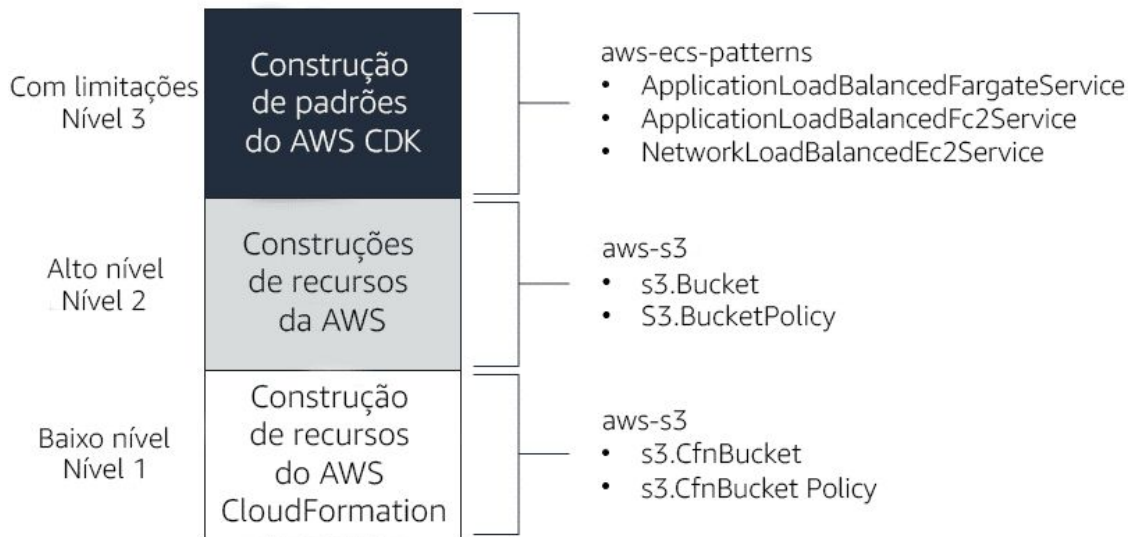


Três componentes básicos

As construções são implementadas em classes que estendem a classe base Construção. Instancie a classe para definir uma construção. Todas as construções usam três parâmetros quando são inicializadas: **scope, id e props**.

```
constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {  
    super(scope, id, props);  
  
    const vpc = new ec2.Vpc(this, "MyVpc", {maxAzs: 2});  
    ;  
}
```

Níveis de construções



Por exemplo, a construção `ecs_patterns.ApplicationLoadBalancedFargateService` representa uma arquitetura com um `cluster de contêiner do Fargate` que usa um `Application Load Balancer`.

```
1 import * as cdk from '@aws-cdk/core';
2 import * as ec2 from '@aws-cdk/aws-ec2';
3 import * as ecs from '@aws-cdk/aws-ecs';
4 import * as ecs_patterns from '@aws-cdk/aws-ecs-patterns';
5
6 export class CdkPrimerStack extends cdk.Stack {
7   constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
8     super(scope, id, props);
9
10    const vpc = new ec2.Vpc(this, "MyVpc", {maxAzs: 2});
11    const cluster = new ecs.Cluster(this, "MyCluster", {vpc: vpc});
12    new ecs_patterns.ApplicationLoadBalancedFargateService(this, "MyFargateService", {
13      cluster: cluster,
14      taskImageOptions: { image: ecs.ContainerImage.fromRegistry("amazon/amazon-ecs-sample") },
15      publicLoadBalancer: true
16    });
17  }
18 }
19
```

Ambientes

O ambiente (**env**) representa a conta e a região da AWS em que uma pilha é implantada.

O AWS CDK seleciona a região e a conta padrão em seu perfil atual da AWS CLI. No entanto, é possível especificar manualmente um conjunto de valores diferente do padrão para modificar o ambiente.

```
const app = new cdk.App();
const env1 = { account: '444455556666', region: 'us-west-1' };
const env2 = { account: '123456789012', region: 'us-west-2' };
new CdkPrimerStack(app, 'CdkPrimerStack', { env: env1 });
new CdkPrimerStack(app, 'CdkPrimerStack2', { env: env2 });
```

Comandos

cdk init

Cria um novo projeto do AWS CDK no diretório atual com base em um modelo especificado

cdk destroy

Destrói as pilhas especificadas

cdk synth

Sintetiza e imprime o modelo do AWS CloudFormation para uma ou mais pilhas especificadas

cdk deploy

Implanta as pilhas especificadas

Análise de código

```
declare const vpc: ec2.IVpc;
const lb = new elb.LoadBalancer(this, 'LB', {
  vpc,
  internetFacing: true,
});

// instance to add as the target for load balancer.
const instance = new ec2.Instance(this, 'targetInstance', {
  vpc: vpc,
  instanceType: ec2.InstanceType.of(ec2.InstanceClass.BURSTABLE2, ec2.InstanceSize.MICRO),
  machineImage: new ec2.AmazonLinuxImage({ generation: ec2.AmazonLinuxGeneration.AMAZON_LINUX_2 }),
});
lb.addTarget(new elb.InstanceTarget(instance));
```



thanks!

