Почему Pulumi это охрененно, но при этом полное 💩

Александров Андрей

Ни один тезис не будет раскрыт!

Все детали обсудим после доклада

Cloud Native Infrastructure as Code

Экосистема работы со сложностью

Экосистема работы со сложностью

Unit Testing

Экосистема работы со сложностью

Unit Testing

Property Testing

Экосистема работы со сложностью

Unit Testing

Property Testing

Integration Testing

Экосистема работы со сложностью

Unit Testing

Property Testing

Integration Testing

Policy as Code

Экосистема работы со сложностью

Unit Testing

Property Testing

Integration Testing

Policy as Code

Поддержка любой структуры проекта

Экосистема работы со сложностью

Unit Testing

Property Testing

Integration Testing

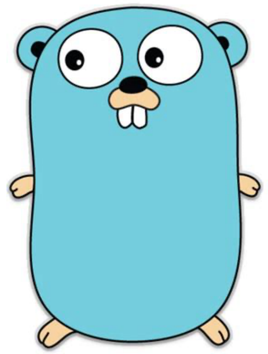
Policy as Code

Поддержка любой структуры проекта

Собственные абстракций!


JS

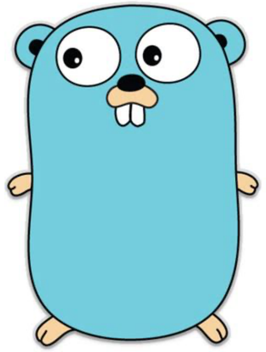

TS




C#


JS


TS




C#

```
const databasesideListener = new k8s.core.v1.Service("database-side-listener", {
    metadata: { labels: databaseDeployment.metadata.labels },
    spec: {
        type: "ClusterIP",
        ports: [{ port: 5432, targetPort: "http" }],
        selector: databaseAppLabels,
        publishNotReadyAddresses: false,
    }}, {
    provider: eksCluster.provider,
    },
);
```

```
const webSg = new aws.ec2.SecurityGroup("webServerSecurityGroup", {
    description: "Enable HTTP and SSH access",
    egress: [
        { protocol: "-1", fromPort: 0, toPort: 0, cidrBlocks: [ "0.0.0.0/0" ] },
    ],
    ingress: [
        { protocol: "-1", fromPort: 0, toPort: 0, cidrBlocks: [ "0.0.0.0/0" ] },
    ],
});
```

https://github.com/pulumi/examples

```
transformations: [
  (obj: any) => {
    if (obj.kind === "PersistentVolumeClaim"
          && obj.metadata.name === "kube-prometheus-stack-grafana") {
      obj.spec.selector = { volume: "grafana" }
    }
  }
]
```

```
const projectNamespaces = [
  "ns1",
  "ns2",
  "ns3",
]
projectNamespaces.map(function(ns) {
  return new k8s.core.v1.Namespace(ns, {
    metadata: { name: ns }
  });
});
```

https://github.com/pulumi/examples

```
new Accounts.Administrators(["user1", "user2", "user3"]);
new Accounts.Developers(["user4", "user5"]);
```

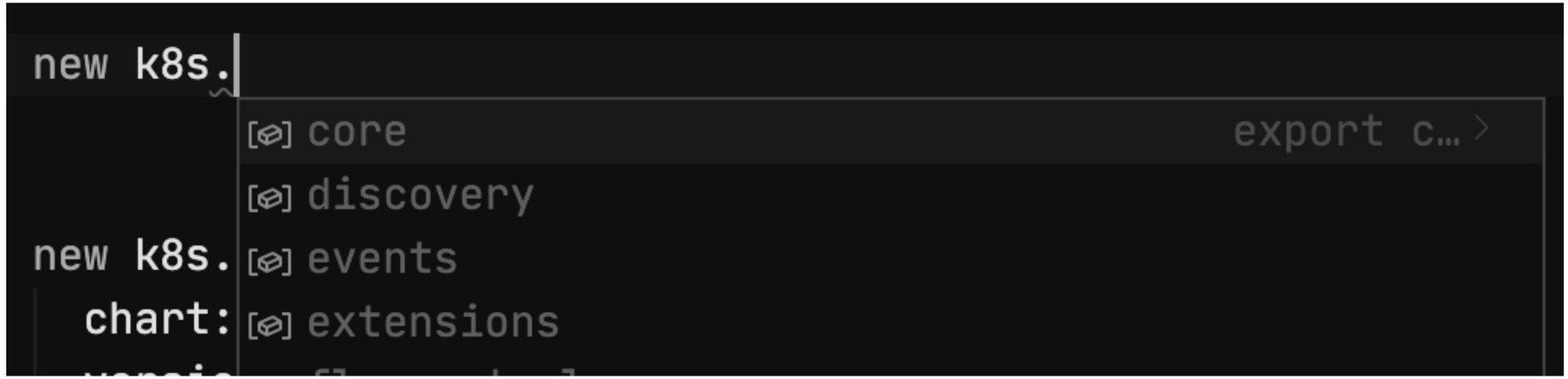
```
export class Administrators extends pulumi.ComponentResource {

  constructor(usernames: pulumi.Input<string>[], opts: pulumi.ComponentResourceO
    super("Accounts:Administrators", "admin-accounts", {}, opts);

    const serviceAccounts = usernames.map((username: pulumi.Input<string>) => {
      return new k8s.core.v1.ServiceAccount(`${ username }-sa`, {
        metadata: {
          name: username,
          namespace: "kube-system",
          annotations: {}
        }
      }, { parent: this });
    });

    new k8s.rbac.v1.ClusterRoleBinding("crb", {
      metadata: {
```

```typescript
// Create a Kubernetes PersistentVolumeClaim.
const pvc = new k8s.core.v1.PersistentVolumeClaim('data', {
  spec: {
    accessModes: ['ReadWriteOnce'],
    resources: { requests: { storage: '1Gi' } }
  }
})
// Create a Kubernetes ConfigMap.
const cm = new k8s.core.v1.ConfigMap('cm', {
  data: { config: 'very important data' }
})
// Create a Kubernetes Secret.
const secret = new k8s.core.v1.Secret('password', {
  stringData: {
    password: new random.RandomPassword('pw', {
      length: 12
    }).result
  }
})
// Create a Kubernetes Deployment.
const appLabels = { app: 'nginx' }
const deployment = new k8s.apps.v1.Deployment('nginx', {
  spec: {
    selector: { matchLabels: appLabels },
    replicas: 1,
    template: {
      metadata: { labels: appLabels },
      spec: {
        containers: [
          {
            name: 'nginx',
            image: 'nginx',
            env: [
              {
                name: 'CONFIG',
                valueFrom: {
                  configMapKeyRef: {
                    key: 'config',
                    name: cm.metadata.name
                  }
                }
              },
              {
                name: 'PASSWORD',
                valueFrom: {
                  secretKeyRef: {
                    key: 'password',
                    name: secret.metadata.name
                  }
                }
              }
            ],
            ports: [{ name: 'http', containerPort: 80 }],
            volumeMounts: [
              {
                name: 'data',
                mountPath: '/data'
              }
            ]
          }
        ],
        volumes: [
          {
            name: 'data',
            persistentVolumeClaim: {
              claimName: pvc.metadata.name
            }
          }
        ]
      }
    }
  }
})
// Create a Kubernetes Service.
const service = new k8s.core.v1.Service('nginx', {
  spec: {
    ports: [{ name: 'http', port: 80 }],
    selector: appLabels,
    type: 'LoadBalancer'
  }
})
```

```typescript
// Create a Kubernetes PersistentVolumeClaim.
const pvc = new kx.PersistentVolumeClaim('data', {
  spec: {
    accessModes: ['ReadWriteOnce'],
    resources: { requests: { storage: '1Gi' } }
  }
})
// Create a Kubernetes ConfigMap.
const cm = new kx.ConfigMap('cm', {
  data: { config: 'very important data' }
})
// Create a Kubernetes Secret.
const secret = new kx.Secret('secret', {
  stringData: {
    password: new random.RandomPassword('pw', {
      length: 12
    }).result
  }
})
// Define a Pod.
const pb = new kx.PodBuilder({
  containers: [
    {
      env: {
        CONFIG: cm.asEnvValue('config'),
        PASSWORD: secret.asEnvValue('password')
      },
      image: 'nginx',
      ports: { http: 8080 },
      volumeMounts: [pvc.mount('/data')]
    }
  ]
})
// Create a Kubernetes Deployment.
const deployment = new kx.Deployment('nginx', {
  spec: pb.asDeploymentSpec()
})
// Create a Kubernetes Service.
const service = deployment.createService({
  type: kx.types.ServiceType.LoadBalancer
})
```

https://github.com/pulumi/pulumi-kubernetesx

Подсказки



Type Checking

```
      Types of property 'version' are incompatible.
        Type 'number' is not assignable to type 'string | Promise<string> | OutputInstance<string> | undefined'.
```

crd2pulumi

```
// Register the CronTab CRD.
const cronTabDefinition = new crontabs.stable.CronTabDefinition("my-crontab-definition")

// Instantiate a CronTab resource.
const myCronTab = new crontabs.stable.v1.CronTab("my-new-cron-object",
{
    metadata: {
        name: "my-new-cron-object",
    },
    spec: {
        cronSpec: "* * * * */5",
        image: "my-awesome-cron-image",
    }
})
```

Реальный Diff

Реальный Diff

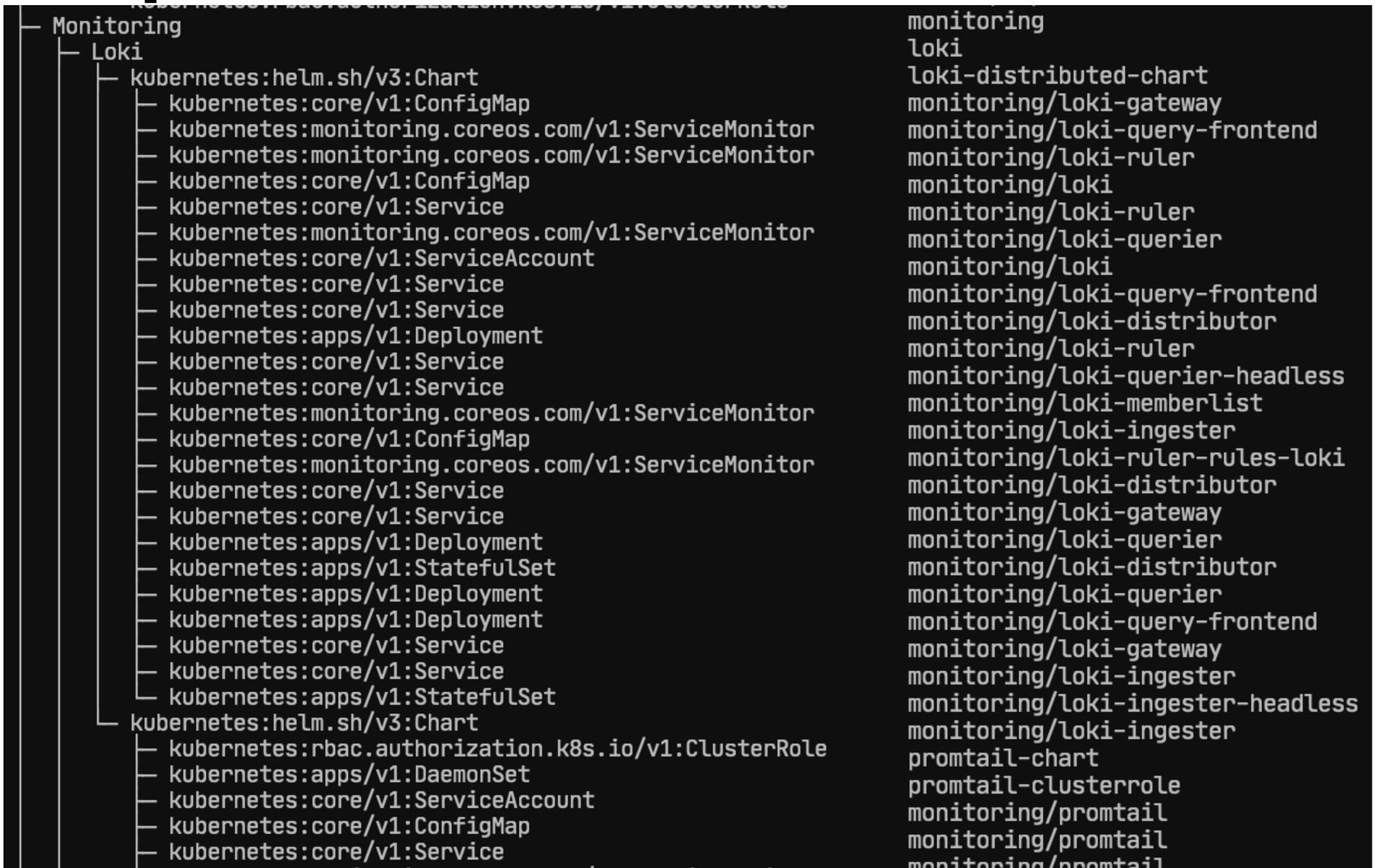
```
~ spec: {
    ~ ports: [
        ~ [0]: {
              ~ port      : 10259 => 10258
              ~ targetPort: 10259 => 10258
            }
      ]
  }
```

Реальный Diff

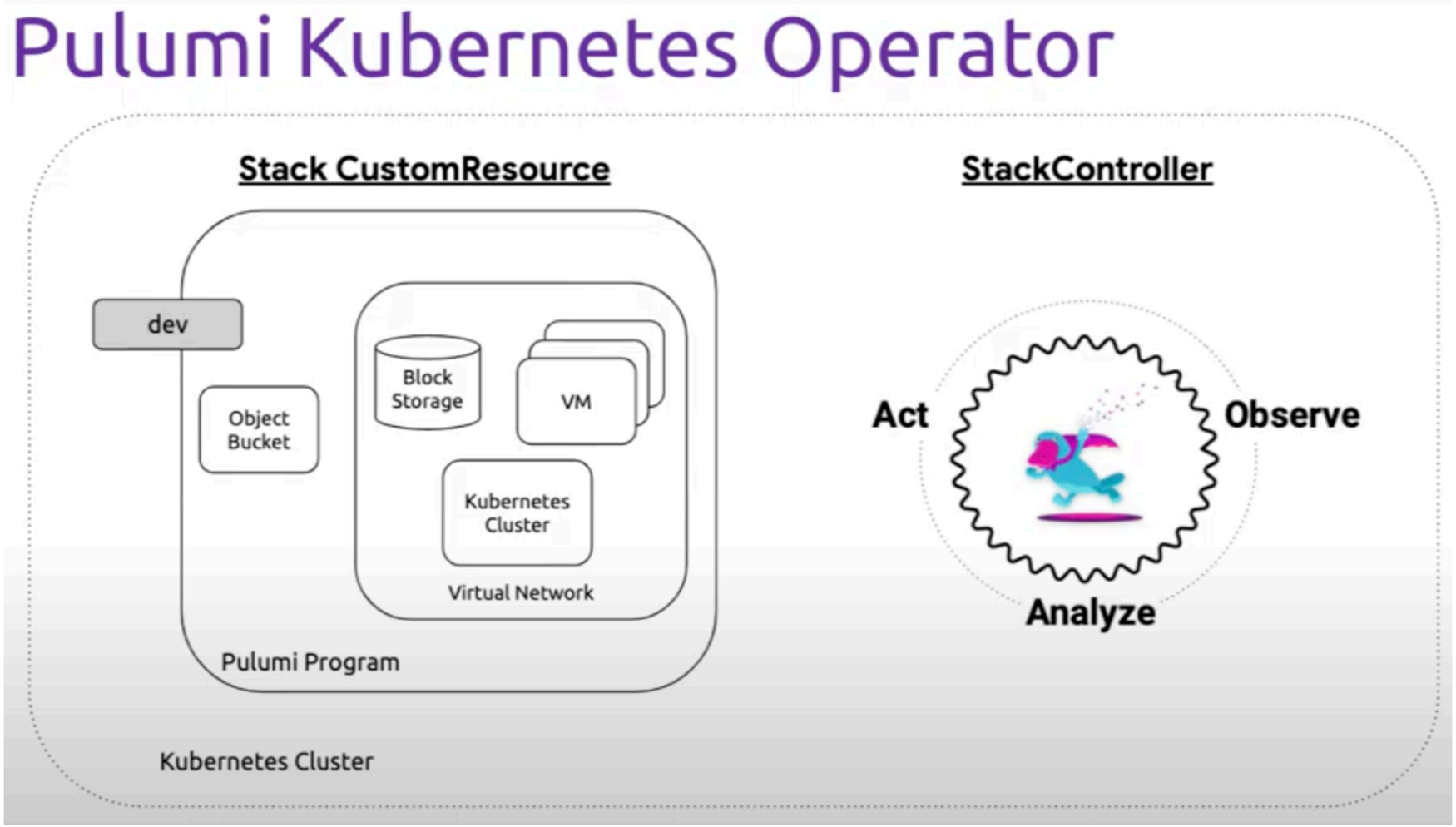
```
~ spec: {
    ~ ports: [
        ~ [0]: {
                ~ port      : 10259 => 10258
                ~ targetPort: 10259 => 10258
              }
    ]
  }
```

```
- kubernetes:monitoring.coreos.com/v1:PrometheusRule: (delete)
    [id=monitoring/kube-prometheus-stack-kubernetes-system-scheduler]
    [urn=                                                          :core/v1:Names
etheusRule::monitoring/kube-prometheus-stack-kubernetes-system-schedul
    [provider=                                                                rnetes::d
- kubernetes:monitoring.coreos.com/v1:ServiceMonitor: (delete)
    [id=monitoring/kube-prometheus-stack-kube-scheduler]
    [urn=                                                          :core/v1:Names
iceMonitor::monitoring/kube-prometheus-stack-kube-scheduler]
    [provider=                                                                rnetes::d
- kubernetes:monitoring.coreos.com/v1:PrometheusRule: (delete)
    [id=monitoring/kube-prometheus-stack-kube-scheduler.rules]
    [urn=                                                          :core/v1:Names
etheusRule::monitoring/kube-prometheus-stack-kube-scheduler.rules]
    [provider=                                                                rnetes::d
- kubernetes:core/v1:ConfigMap: (delete)
    [id=monitoring/kube-prometheus-stack-scheduler]
    [urn=                                                          :core/v1:Names
/kube-prometheus-stack-scheduler]
    [provider=                                                                rnetes::d
- kubernetes:core/v1:Service: (delete)
    [id=kube-system/kube-prometheus-stack-kube-scheduler]
    [urn=                                                          :core/v1:Names
kube-prometheus-stack-kube-scheduler]
    [provider=                                                                rnetes::d
```

Graph


├─ Monitoring
│ ├─ Loki
│ │ ├─ kubernetes:helm.sh/v3:Chart
│ │ │ ├─ kubernetes:core/v1:ConfigMap
│ │ │ ├─ kubernetes:monitoring.coreos.com/v1:ServiceMonitor
│ │ │ ├─ kubernetes:monitoring.coreos.com/v1:ServiceMonitor
│ │ │ ├─ kubernetes:core/v1:ConfigMap
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:monitoring.coreos.com/v1:ServiceMonitor
│ │ │ ├─ kubernetes:core/v1:ServiceAccount
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:apps/v1:Deployment
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:monitoring.coreos.com/v1:ServiceMonitor
│ │ │ ├─ kubernetes:core/v1:ConfigMap
│ │ │ ├─ kubernetes:monitoring.coreos.com/v1:ServiceMonitor
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:apps/v1:Deployment
│ │ │ ├─ kubernetes:apps/v1:StatefulSet
│ │ │ ├─ kubernetes:apps/v1:Deployment
│ │ │ ├─ kubernetes:apps/v1:Deployment
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ ├─ kubernetes:core/v1:Service
│ │ │ └─ kubernetes:apps/v1:StatefulSet
│ │ └─ kubernetes:helm.sh/v3:Chart
│ │   ├─ kubernetes:rbac.authorization.k8s.io/v1:ClusterRole
│ │   ├─ kubernetes:apps/v1:DaemonSet
│ │   ├─ kubernetes:core/v1:ServiceAccount
│ │   ├─ kubernetes:core/v1:ConfigMap
│ │   ├─ kubernetes:core/v1:Service
monitoring
loki
loki-distributed-chart
monitoring/loki-gateway
monitoring/loki-query-frontend
monitoring/loki-ruler
monitoring/loki
monitoring/loki-ruler
monitoring/loki-querier
monitoring/loki
monitoring/loki-query-frontend
monitoring/loki-distributor
monitoring/loki-ruler
monitoring/loki-querier-headless
monitoring/loki-memberlist
monitoring/loki-ingester
monitoring/loki-ruler-rules-loki
monitoring/loki-distributor
monitoring/loki-gateway
monitoring/loki-querier
monitoring/loki-distributor
monitoring/loki-querier
monitoring/loki-query-frontend
monitoring/loki-gateway
monitoring/loki-ingester
monitoring/loki-ingester-headless
monitoring/loki-ingester
promtail-chart
promtail-clusterrole
monitoring/promtail
monitoring/promtail

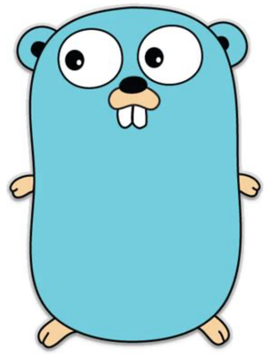
Для наркоманов завезли оператор!



https://www.pulumi.com/blog/tag/operator/


JS


TS




C#


JS


TS

HashiCorp

Terraform

Под редкое облако может не быть провайдера

Состояние! 💩

Diff врет!

-Потому что состояние, надо делать refresh

-Потому что k8s врет :(

Два параллельных запуска могут разломать стейт 💩

Два параллельных запуска могут разломать стейт 💩

- Локов как в terraform нет, надо разруливать самому

Два параллельных запуска могут разломать стейт 💩

- Локов как в terraform нет, надо разруливать самому

- В Enterprise версии не ломает, оно там как-то конкурентно это разруливает ¯_(ツ)_/¯

Большая свобода, большая ответственность

Думать о структуре сложна 💩

Рефакторинг сложна! 💩

Часто проще перелопатить SED’ом state, чем работать с alias 💩

Pulumi это охреенно, но при этом полное 💩

Александров Андрей

https://t.me/aladmit_world