



CDK FOR KUBERNETES DEEP DIVE

Go From k8s YAML Engineer To Software Engineer With cdk8s

Robert Hoffmann

Senior Solutions Architect
Amazon Web Services

Twitter: [@robhoffmax](https://twitter.com/robhoffmax)





YAML:

- Y: Yelling
- A: At
- M: My
- L: Laptop

Let's engineer some YAML



Http Echo Server – with YAML

```

• • •
apiVersion: apps/v1
kind: Deployment
metadata:
  name: echo-deployment
spec:
  selector:
    matchLabels:
      app: echo
  template:
    metadata:
      labels:
        app: echo
    spec:
      containers:
        - args:
            - -text
            - hello
          image: hashicorp/http-echo
          name: main
          ports:
            - containerPort: 5678

```

```

• • •
apiVersion: v1
kind: Service
metadata:
  name: echo-service
spec:
  ports:
    - port: 5678
      targetPort: 5678
  selector:
    app: echo

```

```

• • •
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: echo-ingress
spec:
  rules:
    - http:
        paths:
          - backend:
              service:
                name: echo-service
                port:
                  number: 5678
            path: /hello
            pathType: Prefix

```

~ 45 lines of YAML

Kubernetes YAML

- Using plain YAML results in a lot of copy / paste
- Often you have to make multiple changes at different locations
- Customizations are manual or require special tools
- Hard to share or you need a special package manager

This creates high cognitive load for engineers.

The cdk8s approach: infrastructure as actual code



cdk8s in a nutshell

```

export class MyChart extends Chart {

  constructor(scope: Construct, id: string) {
    super(scope, id)

    new kplus.Deployment(this, 'Deployment', {
      containers: [
        {
          image: 'hashicorp/http-echo',
          args: [ '-text', 'hello' ],
          port: 5678
        }
      ]
    })
  }
}

```

Code

cdk8s synth



```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: cdk8splus-deployment-c8771008
spec:
  minReadySeconds: 0
  progressDeadlineSeconds: 600
  replicas: 1
  selector:
    matchLabels:
      cdk8s.io/metadata.addr: cdk8splus-Deployment-c81d4eba
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      labels:
        cdk8s.io/metadata.addr: cdk8splus-Deployment-c81d4eba
    spec:
      automountServiceAccountToken: true
      containers:
        - args:
            - -text
            - hello
          image: hashicorp/http-echo
          imagePullPolicy: Always
          name: main
          ports:
            - containerPort: 5678
          securityContext:
            privileged: false
            readOnlyRootFilesystem: false
            runAsNonRoot: false
          dnsPolicy: ClusterFirst
          securityContext:
            fsGroupChangePolicy: Always
            runAsNonRoot: false
            setHostnameAsFQDN: false

```

Config

CDK For Kubernetes (cdk8s)

An open source multi-language software development framework for modeling Kubernetes resources as reusable components



Go from Code to Config

Define Kubernetes applications and architectures using familiar programming languages.



Cut Copy & Paste

Turn best practices into code libraries and share easily.



Run Everywhere

cdk8s runs locally and generates YAML you can deploy to any cluster, anywhere.



Open source CDK, for Everyone!

AWS

Terraform



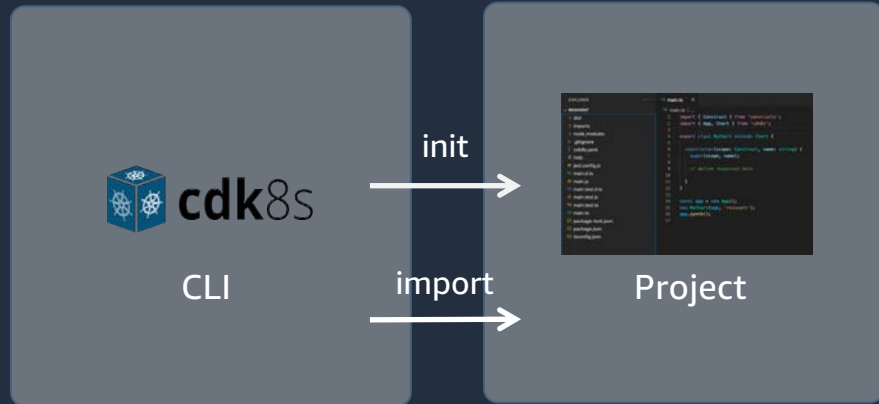
Construct Hub
<https://constructs.dev/>

Kubernetes

Working with cdk8s



Initialize your project



```
npm install -g cdk8s-cli
```

```
cdk8s init <TYPE>
```

```
cdk8s import <SPEC>
```

init – scaffold a project

```
cdk8s init typescript-app  
go-app  
java-app  
python-app
```

Create a new cdk8s project from a template.

> __snapshots__

> dist

> imports

◆ .gitignore

! cdk8s.yaml

☰ help

JS jest.config.js

TS main.d.ts

JS main.js

TS main.test.d.ts

JS main.test.js

TS main.test.ts

TS main.ts

{ } package-lock.json

{ } package.json

TS tsconfig.json

👤 yarn.lock

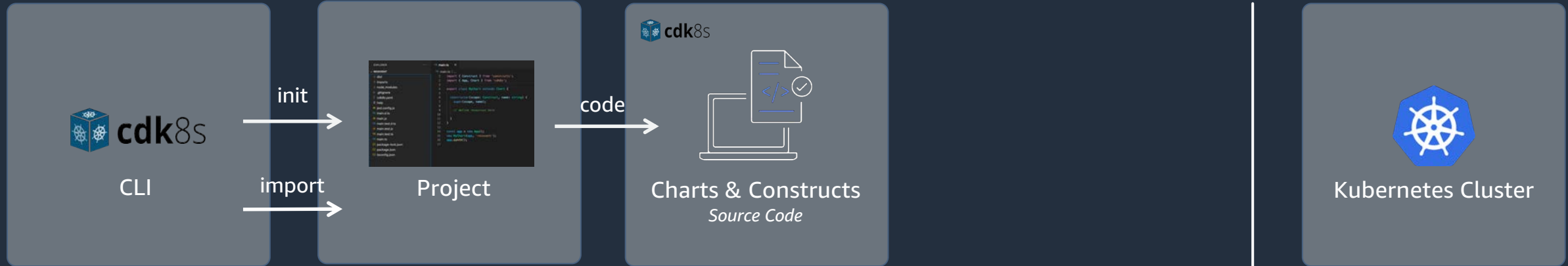
import – get k8s resources as code

```
cdk8s import k8s@1.20.0  
            jenkins.io_jenkins_crd.yaml
```

Imports API objects to your app by generating constructs.

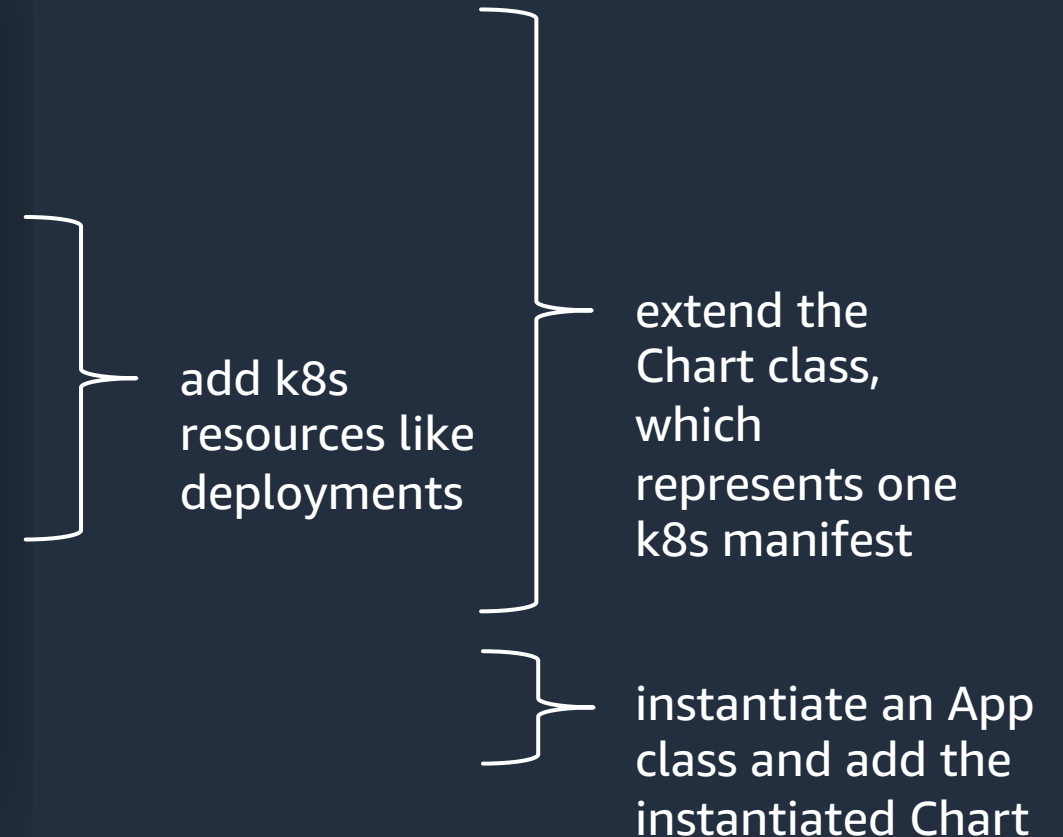


Model your k8s manifests as code



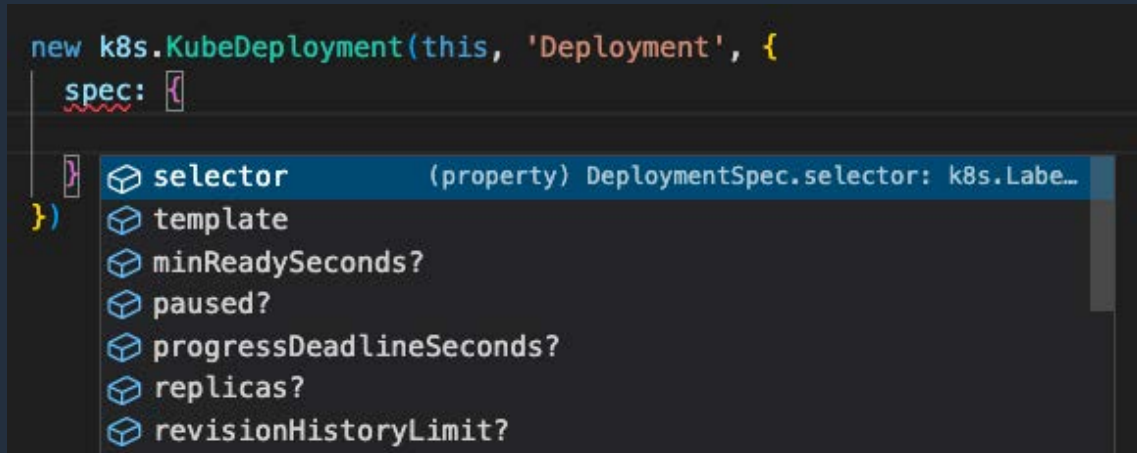
The anatomy of the main code

```
export class MyChart extends Chart {  
  
  constructor(scope: Construct, id: string) {  
    super(scope, id)  
  
    new kplus.Deployment(this, 'Deployment', {  
      containers: [  
        {  
          image: 'hashicorp/http-echo',  
          args: [ '-text', 'hello' ],  
          port: 5678  
        }  
      ]  
    })  
  }  
}  
  
const app = new App()  
new MyChart(app, 'my-chart')
```



Code completion and inline documentation

```
new k8s.KubeDeployment(this, 'Deployment', {  
  spec: {  
    selector  
  }  
})
```



Your config is
typed now! 🥳

```
(property) DeploymentSpec.selector: k8s.LabelSelector  
  
Label selector for pods. Existing ReplicaSets whose pods are selected by this will be the  
ones affected by this deployment. It must match the pod template's labels.  
  
@schema — io.k8s.api.apps.v1.DeploymentSpec#selector  
  
new  
sp  
selector: {}  
})
```



HTTP Echo Server – with cdk8s

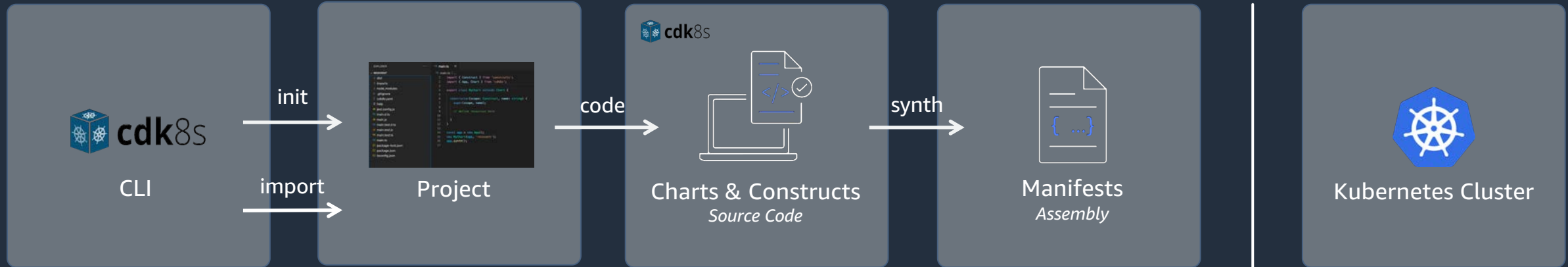
```
const port = 5678
const labels = { app: 'echo' }
```

```
new k8s.KubeDeployment(this, 'Deployment', {
  spec: {
    selector : {matchLabels: labels},
    template: {
      metadata: {labels : labels},
      spec: {
        containers: [{
          name: 'main',
          image: 'hashicorp/http-echo',
          args: ['-text', 'hello'],
          ports: [{containerPort: port}],
        }]
      }
    }
  }
})
```

```
const service = new k8s.KubeService(this, 'Service', {
  spec: {
    ports: [{
      port: port,
      targetPort: k8s.IntOrString.fromNumber(port)
    }],
    selector: labels
  }
})
```

```
new k8s.KubeIngress(this, 'Ingress', {
  spec: {
    rules: [{
      http: {
        paths: [{
          pathType: 'Prefix',
          path: '/hello',
          backend: {
            service: {
              name: service.name,
              port: {
                number: port
              }
            }
          }
        }]
      }
    }
  }
})
```

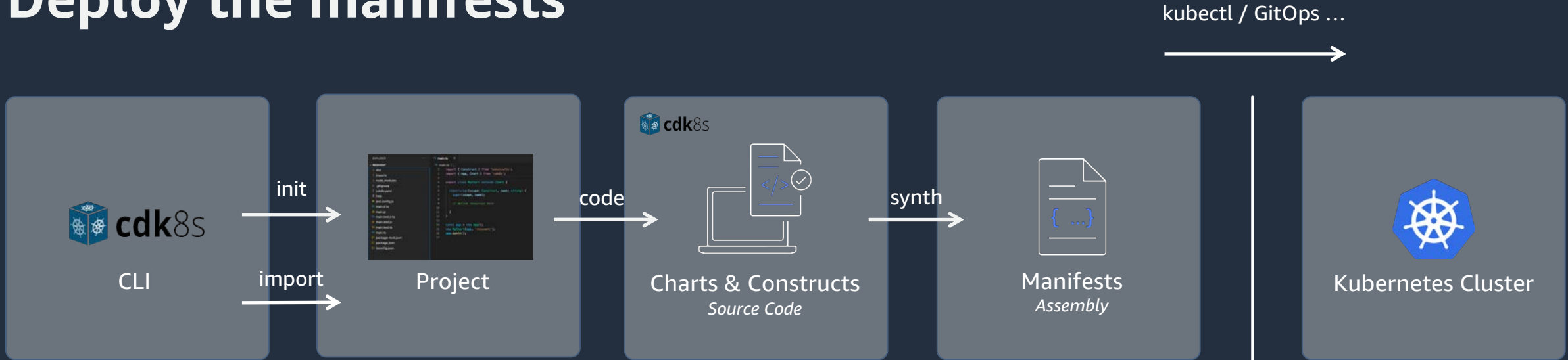
Synthesize the k8s manifests



`cdk8s synth`

Synthesizes Kubernetes manifests for all charts in your app.

Deploy the manifests

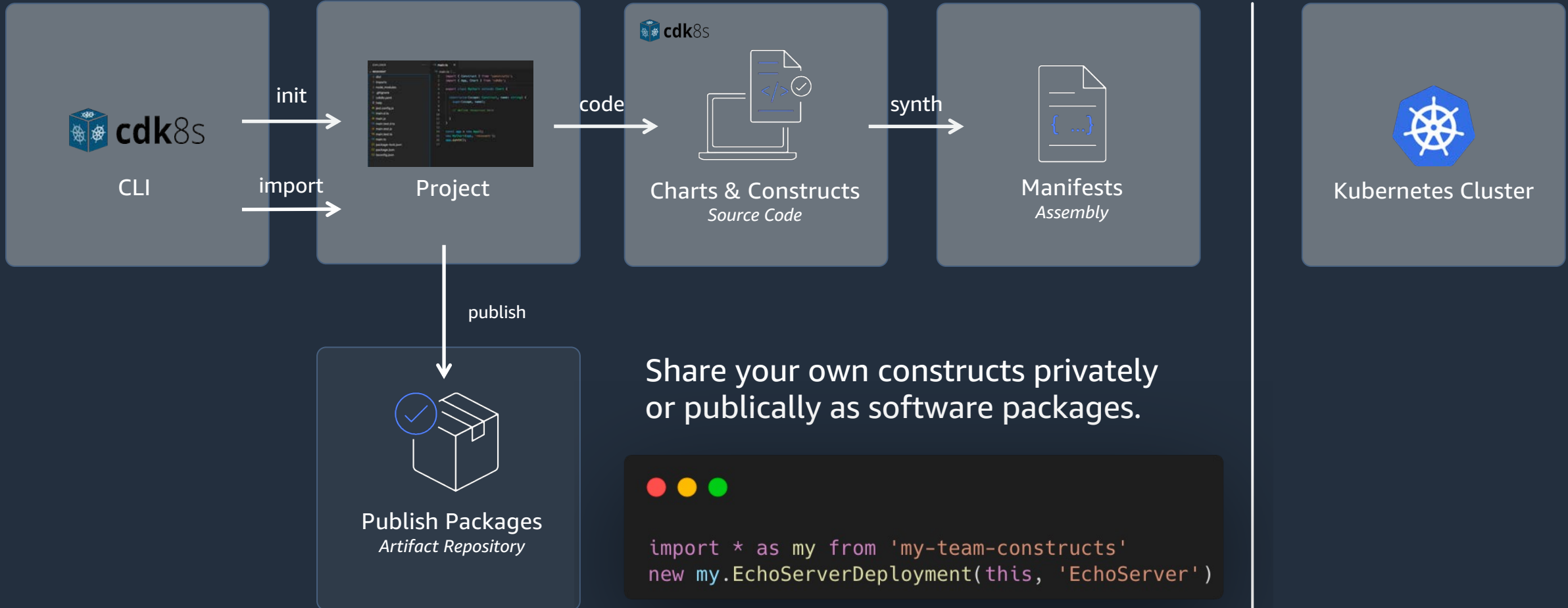


```
kubectl apply -f /dist/manifest.yaml
```

Deploy the Kubernetes manifests with your favorite tools.

Publish your code as a re-usable package

kubectl / GitOps ...



How cdk8s+ reduces cognitive load with abstractions



Construct Levels

L2

`kplus.Deployment`

Hand crafted high level APIs

L1

`k8s.KubeDeployment`

Automatically generated

L0

`cdk8s.ApiObject`

Common Functionality

Http Echo Server – with cdk8s+

```
const port = 5678

const deployment = new kplus.Deployment(this, 'Deployment', {
  containers: [
    {
      image: 'hashicorp/http-echo',
      args: [ '-text', 'hello' ],
      port: port
    }
  ]
})

deployment.exposeViaIngress('/hello', {
  pathType: kplus.HttpIngressPathType.PREFIX
})
```

~ 13 lines of code

Declare your intent!
Write for humans, not machines.

Clean Code

Code is clean if it can be understood easily – by everyone on the team.

Clean code can be read and enhanced by a developer other than its original author.

With understandability comes readability, changeability, extensibility and maintainability.

- Robert C. Martin, and others

```

const port = 5678

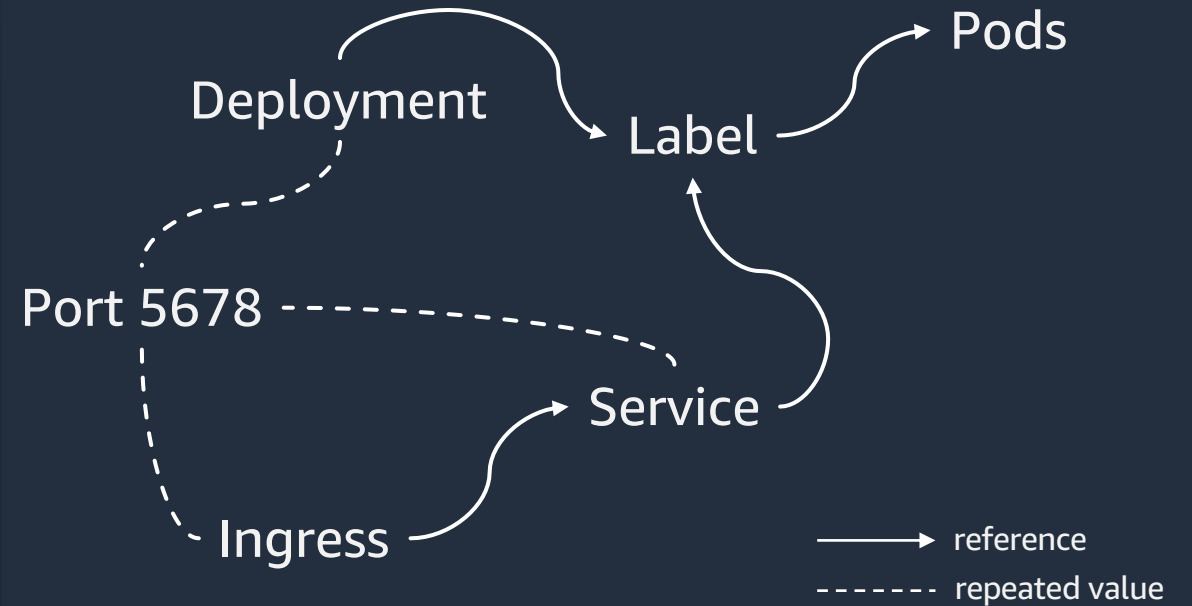
const deployment = new kplus.Deployment(this, 'Deployment', {
  containers: [
    {
      image: 'hashicorp/http-echo',
      args: [ '-text', 'hello' ],
      port: port
    }
  ]
})

deployment.exposeViaIngress('/hello', {
  pathType : kplus.HttpIngressPathType.PREFIX
})

```

The story:

“I want to create a deployment and expose it via ingress.”



Under the hood:

cdk8s creates and wires multiple resources for you, taking care of technical details if possible.

Why should I code my infrastructure?

- Software engineering practices: Clean Code
- Familiar programming languages
- Great IDE / tool support
- Powerful abstractions with cdk8s+
- Developer platform: Build your own abstractions and share them

Why not?

- You want to keep your existing tools (you could mix & match)
- You don't like coding
- You are the author of a Helm chart

Resources

Getting started with cdk8s

- Project website: <https://cdk8s.io/>
- Find constructs published by the open-source community: <https://constructs.dev/>

Interact with the community

- CNCF Sandbox project website: <https://www.cncf.io/projects/cdk-for-kubernetes-cdk8s/>
- Slack: <https://cdk.dev/>
- The community event for the whole CDK ecosystem: <https://www.cdkday.com/>



**Give us your feedback on
the session and potential
new cdk8s features.**

Robert Hoffmann

Twitter: @robhoffmax





Thank you!

Robert Hoffmann

Twitter: @robhoffmax