参考

https://cloud-ace.jp/column/detail229/
https://qiita.com/yuichi1992_west/items/571016084c110d15320e
https://amateur-engineer-blog.com/getting-started-minikube/
https://amateur-engineer-blog.com/create-multi-cluster-by-minikube/
https://zenn.dev/nameless_gyoza/articles/f22466cadaf60bf50405
https://developer.mamezou-tech.com/blogs/2022/04/18/use-minikube-profile/
https://qiita.com/yuichi1992_west/items/49470a7a347f5a932e98

概要

コンテナ化されたアプリケーションをどこにでもデプロイ、スケール、管理できるオープンソースのシステム。
Docker は単一のコンテナを実行するコンテナエンジンで、Kubernetes は複数のコンテナを管理するツールのイメージ。
Kubernetes はコンテナエンジンとして､Docker 以外にも Containerd などを利用することができる。

ローカルで使ってみる

Minikube

kubarnetes をローカルで簡単に試したりするためのツールに Minikube がある。

■ minikube のダウンロード

https://minikube.sigs.k8s.io/docs/start/
上記 URL にアクセスして自分の環境を選択すると必要なコマンドが表示される。
Linux(x86) にインストールするときは以下

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

```
minikube status
```

でなにか表示されればインストールされている。

■ kubectl のダウンロード

https://kubernetes.io/ja/docs/tasks/tools/install-kubectl/

```
curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"
sudo install kubectl
```

■ docker

コンテナエンジンとして Docker を使うので Docker を使えるようにしておく

Hello-World

minikube を起動

kubernetes 環境を使えるようにするために minikube を起動する

```
minikube start
```

pod 作成

```
kubectl run hello-world --image hello-world --restart=Never
```

hello-world の pod を作成

```
pod/hello-world created
```

と表示されれば成功。

pod 確認

```
kubectl get pod
```

で以下が表示される

```
NAME          READY   STATUS      RESTARTS   AGE
hello-world   0/1     Completed   0          2m30s
```

実行ログを確認

hello-world コンテナは起動すると「Hello from Docker!」と出力するので、そのログを確認する

```
kubectl logs pod/hello-world
```

minikube の使い方

クラスタ開始

```
minikube start
```

クラスタ停止

```
minikube stop
```

クラスタ削除

```
minikube delete
```

ダッシュボードにアクセス

```
minikube dashboard
```

Web ブラウザで状況の確認ができる

クラスタに ssh

```
minikube ssh
```

プロファイルを使う

minikubeでもクラスタを複数扱うことができる。デフォルトではminikubeというプロファイルでクラスタが起動される。

■クラスタの一覧

```
minikube profile list
```

■クラスタ (プロファイル) 追加

start に -p で任意のプロファイル名を指定することで新しいクラスタを追加できる

```
minikube start -p hoge
minikube profile list
```

クラスタの停止や削除も「-p プロファイル名」で操作対象のプロファイル指定ができる

■全てのクラスタ削除

```
minikube delete --all
```

kubectl の使い方

pod 追加

```
ubectl run hello-world --image=hello-world --restart=Never
```

pod ログ確認

```
kubectl logs pod/hello-world
```

pod 確認

```
kubectl get pod
```

pod の削除

```
kubectl delete pod/hello-world
```

デプロイ

https://qiita.com/yuichi1992_west/items/49470a7a347f5a932e98

■デプロイ用 YAML(nginx.yaml)

```
apiVersion: v1 # 使用する Kubernetes API バージョンを定義
kind: Service # Pod のネットワークである Service リソースを定義
metadata: # 名前やラベルを指定
  labels:
    app: nginx-app
```

```
      name: nginx
    name: nginx-service
  spec: # spec 内にリソースの仕様を記述する
    selector: # 同一のラベルを定義しているリソースを参照
      app: nginx-app
      name: nginx
    type: NodePort # Kubenetes 外からアクセスできるように NodePort を指定 ( ホスト OS から Nginx の画面が確
認できるようにする )
    ports:
    - name: nginx-port # ポート番号を指定
      port: 80
      protocol: TCP
      targetPort: nginx-port
  ---
  apiVersion: apps/v1
  kind: Deployment # Pod の自己修復機能を持つ Deployment リソースを定義
  metadata:
    labels:
      app: nginx-app
      name: nginx
    name: nginx
  spec:
    replicas: 1 # 作成する Pod 数を定義
    selector:
      matchLabels:
        app: nginx-app
        name: nginx
    template:
      metadata:
        labels:
          app: nginx-app
          name: nginx
      spec:
        containers:
        - name: nginx
          image: nginx:latest # Nginx の DockerHub に記載されている DockerImage の名称とバージョンを指定

          ports:
          - containerPort: 80 # Pod から解放するポート番号を定義
            name: nginx-port
            protocol: TCP
```

■デプロイ

```
kubectl apply -f nginx.yaml
```

■確認

```
kubectl get pod
kubectl get service
minikube service nginx-service
```