

# kubernetes勉強会

## (基本編)

---

### kubernetes(k8s)とは？

- Kubernetes は**コンテナ化されたアプリケーション**のデプロイ、スケーリングなどの管理を自動化するためのプラットフォーム（コンテナオーケストレーションエンジン）です。
- 

### コンテナとは？

- OS上に「独立したサーバーと同様の振る舞いをする区画」のこと。  
カーネルなどの機能を使って、OSのプロセスとして起動する。

@boxbg-orange text-white rounded demo-box-pad

---

### Dockerコンテナ

 Dockerコンテナ <http://image.itmedia.co.jp/ait/articles/1701/30/wi-docker01002.png>

---

### コンテナ ≡ 仮想ホスト (VM)

+++

### 参考

#### DockreFileの書き方を学ぶ

- 改めてDockerfileのベストプラクティスを振り返ろう  
<https://www.slideshare.net/ssuser1f3c12/introduce-that-best-practices-for-writing-dockerfiles>
- 

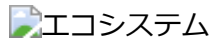
### k8sで何ができるか？

- 複数のKubernetes Node の管理
- コンテナのスケジューリング
- ローリングアップデート
- スケーリング／オートスケーリング
- コンテナの死活監視
- 障害時のセルフヒーリング
- サービスディスカバリ
- ロードバランシング
- データの管理
- ワークロードの管理
- ログの管理

- Infrastructure as Code
- その他エコシステムとの連携や拡張 (下ページ)

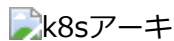
+++

## エコシステム



---

## アーキテクチャ



---

## kubernetes リソース

- **Workloads** リソースコンテナの実行に関するリソース |
- **Discovery & LB** リソースコンテナを外部公開するようなエンドポイントを提供するリソース |
- **Config & Storage** リソース設定/機密情報/永続化ボリュームなどに関するリソース |
- **Cluster** リソースセキュリティやクォータなどに関するリソース |
- **Metadata** リソースクラスタ内の他のリソースを操作するためのリソース |

---

## Workloads リソース

- Pod
- ~~ReplicationController~~ (廃止)
- ReplicaSet
- Deployment
- DaemonSet
- StatefulSet
- Job
- CronJob

---

## Workloadsの階層構造

- Pod ---> ReplicaSet ---> Deployment
- Pod ---> DemonSet
- Pod ---> StatefulSet
- Pod ---> Job ---> CronJob

@box[bg-orange text-white rounded demo-box-pad](Pod が 1 サーバに相当し、コンテナは 1 プロセスに相当する。)

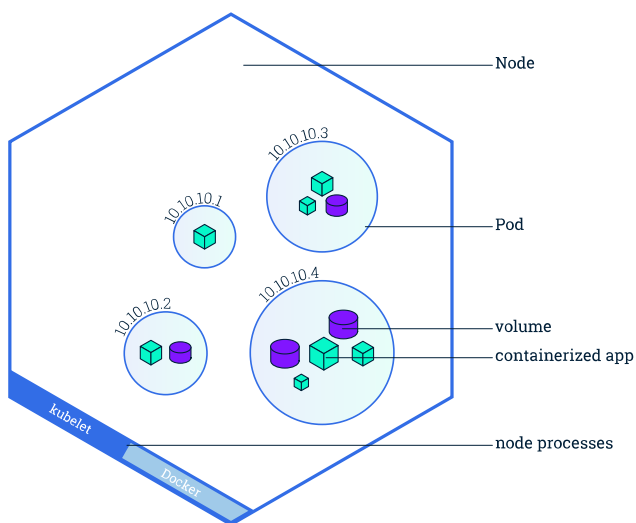
---

## Pods-overview



- <https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/#pods-overview>

## Node-overview



<https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/#node-overview>

## Discovery & LB リソース

- Service
  - ClusterIP ★
  - ExternalIP (ClusterIP の一種)
  - NodePort ★
  - LoadBalancer
  - Headless (None)
  - ExternalName
  - None-Selector
- Ingress ★

## Service の役割

- L4 LoadBalancing
  - クラスタ内DNSによる名前解決
  - ラベルを利用したPodのサービスディスカバリ
- 

## Ingress の役割

- L7 LoadBalancing
  - HTTPS終端
  - パスベースルーティング
- 

## ClusterIP



kind: Service の type: ClusterIP

```
apiVersion: v1
kind: Service
metadata:
  name: sample-clusterip
spec:
  type: ClusterIP
  ports:
    - name: "http-port"
      protocol: "TCP"
      port: 8080
      targetPort: 80
  selector:
    app: sample-app

# ClusterIP Serviceを作成
$ kubectl apply -f clusterip_sample.yml
```

## NodePort



kind: Service の type: NodePort

```
apiVersion: v1
kind: Service
metadata:
```

```
name: sample-nodeport
spec:
  type: NodePort
  ports:
    - name: "http-port"
      protocol: "TCP"
      port: 8080
      targetPort: 80
      nodePort: 30080
  selector:
    app: sample-app

# NodePort Serviceの作成
kubectl apply -f nodeport_sample.yml
```

---

## Ingress

@size[xx-small](LBは一旦Nginx Podまで転送し、NginxがL7相当の処理を行い対象のPodへ転送します。このとき、Nginx Podから対象のPodまではNodePortは通らず、直接PodのIP宛に送られます。)



---

## 小ネタ

- kubectlコマンドのパラメタ補完（必須！） <https://kubernetes.io/docs/tasks/tools/install-kubectl/#enabling-shell-autocompletion>

---

終わり