

Docker 勉強会 1回目

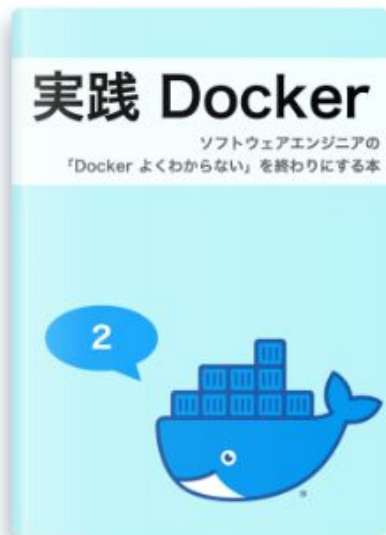
2024/1/31

事前準備

- Dockerを使える状態にしておく
 - [インストール方法 \(Ubuntu20.04\)](#)
- (Docker Hubのアカウント作成)

参考書

実践 Docker – ソフトウェアエンジニアの「Docker よくわからない」を終わりにする本



Chapters

- 1部
 - 導入
- 2部
 - 基礎
- 3部
 - 応用

この資料について

1. 基本的には**参考書に準拠**して解説する
2. 取り扱わないトピックについては、~~このように表記する~~
3. (今回の目的において)特に大事なところは ページ全体を赤枠で覆う



はじめに

- **目的**

- Dockerを普段から使うための簡単な理解をする

- **目標**

- 複数のROSバージョンによる煩わしさから解放される
- ROS2勉強会のための環境をDockerで構築する

結局 何がしたいのか(便利なのか)

- **ROSの複数のワークスペースやバージョンへの対応**
 - 開発環境を統一しやすい(人へ簡単に配布できる)
 - apt installしたものまでバックアップできる(依存関係)
 - apt install ros-<distro>-* や ~msgsなど入れ直す必要がない
- **CUDA 複数のバージョンを同じホストで共存できる**
 - ドライバ, フレームワーク, ライブラリのバージョン管理が楽
 - ホストにCUDA環境を構築する必要がない

→簡単に壊せて, どのマシンでも同じ環境を容易に再現できる.

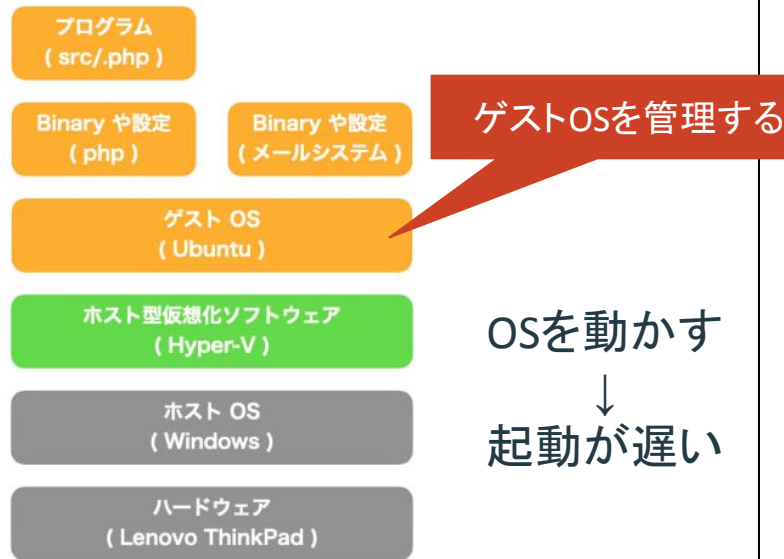
1部のざっくりとした内容

02: 仮想化とは

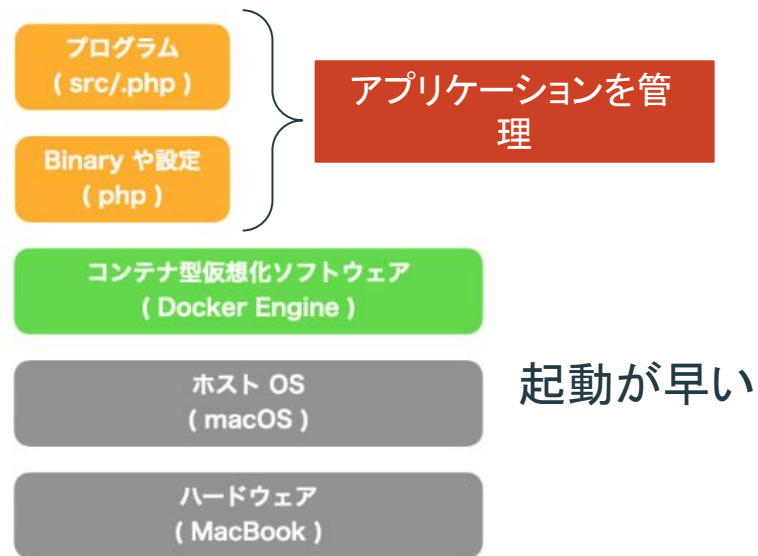
- ホスト型仮想化
- ~~ハイパーバイザー型仮想化~~
- コンテナ型仮想化

02: 仮想化とは

ホスト型仮想化 (VirtualBox, VMware)

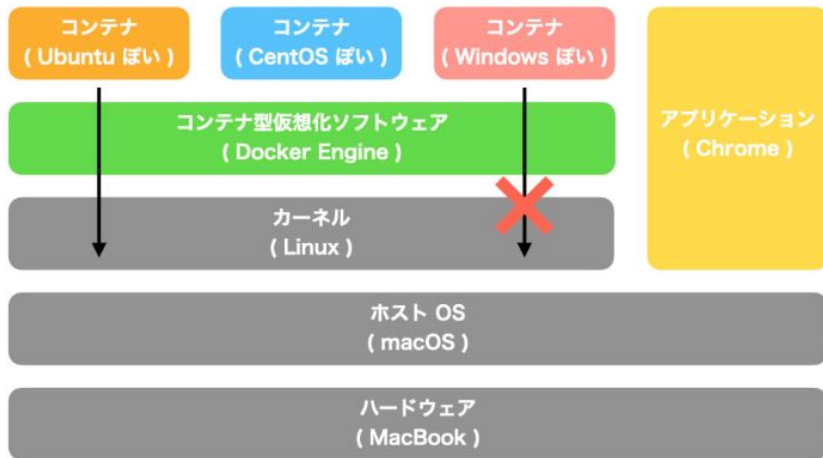


コンテナ型仮想化 (Docker)



02: 仮想化とは

コンテナ型仮想化



- **完全な分離はされない**
 - ホストOSのLinuxカーネルを利用
- **コンテナ型の利点**
 - 起動が早い
 - デプロイがしやすい
- **注意点**
 - OSがあるように見える

1部のざっくりとした内容

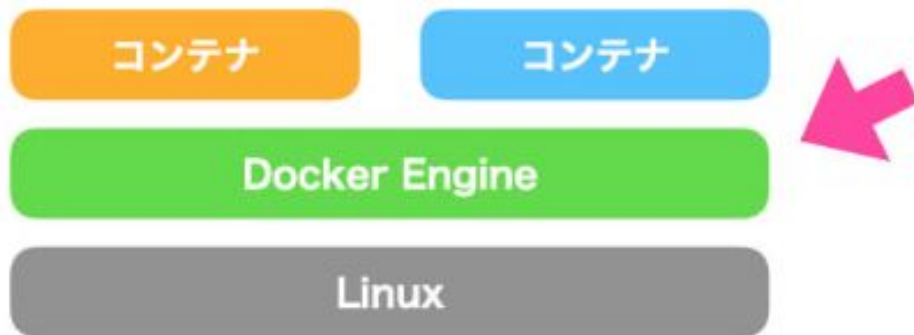
03: Dockerとは

- Docker Engine とは
- Docker CLI とは
- ~~● Docker Desktop とは~~
- ~~● Docker Compose とは~~
- Docker Hub とは
- ~~● ECS / GKE とは~~
- ~~● ECR / GCR とは~~
- ~~● Kubernetes とは~~

03: Dockerとは

Docker Engineとは

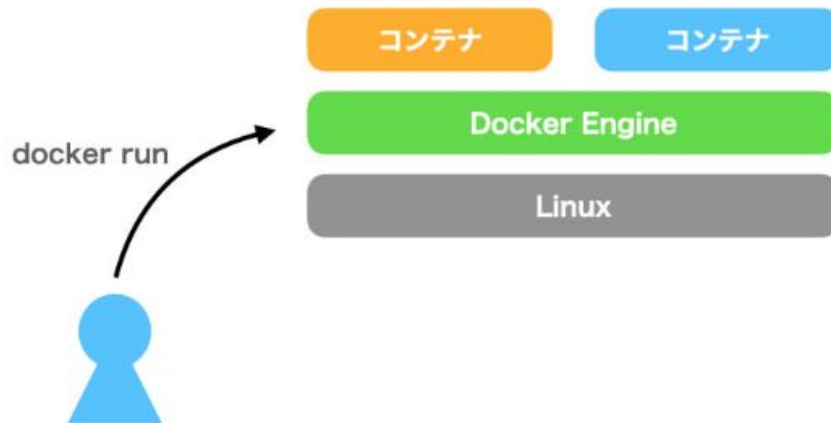
- コンテナを乗せる部分
- Linuxで動くソフトウェア



03: Dockerとは

Docker CLIとは

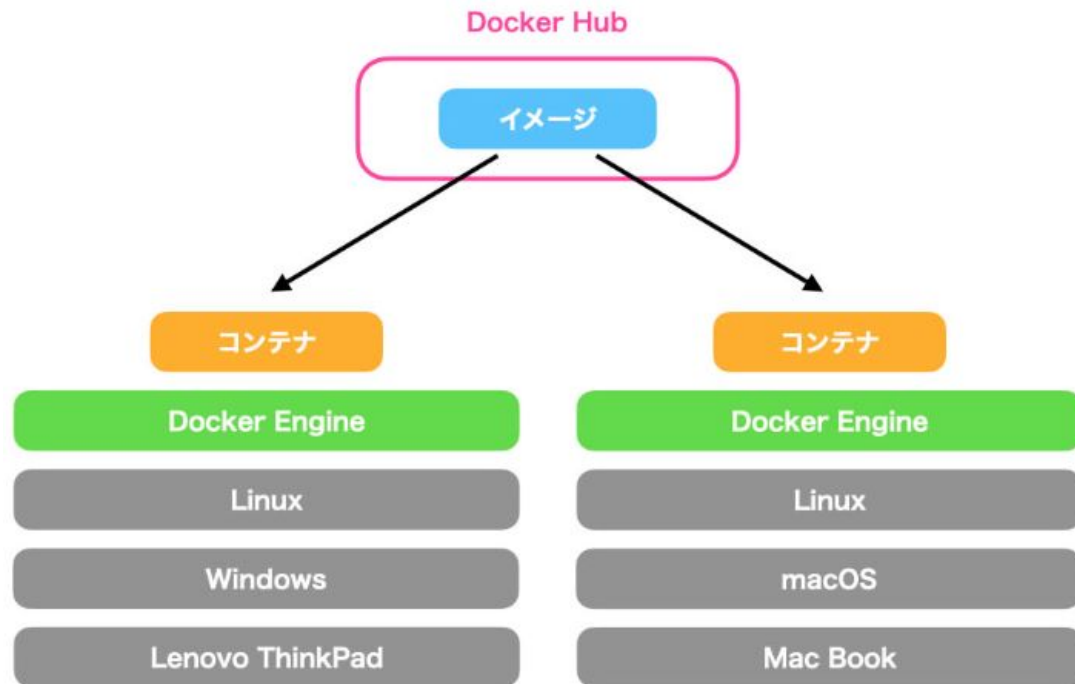
- Dockerのコマンド群のこと



03: Dockerとは

Docker Hubとは

- Docker版の **GitHub**



2部

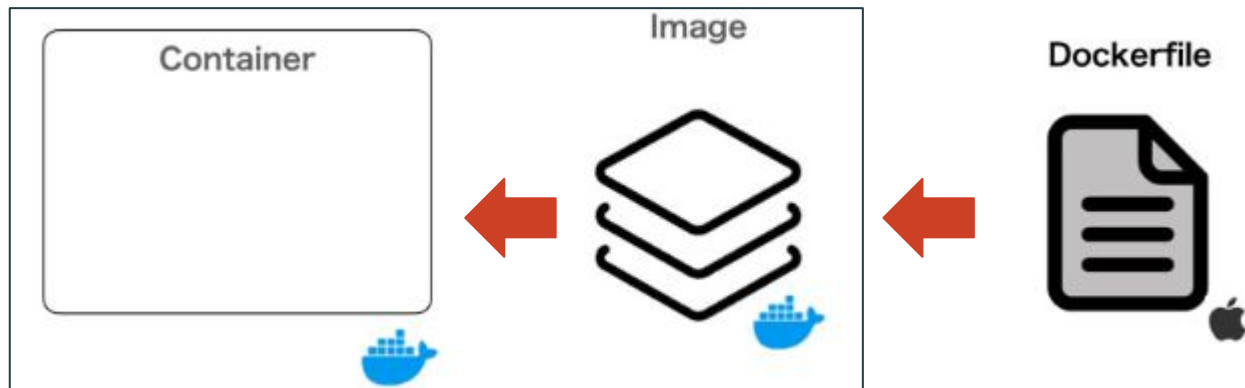
04: Dockerを理解するためのポイント

1. 基本の要素は3つ
2. 基本のコマンドも3つ
3. (コマンドの形を意識する)
4. ~~Dockerと周辺ツールを分ける~~

2部 04: Dockerを理解するためのポイント

1. 基本の要素は3つ

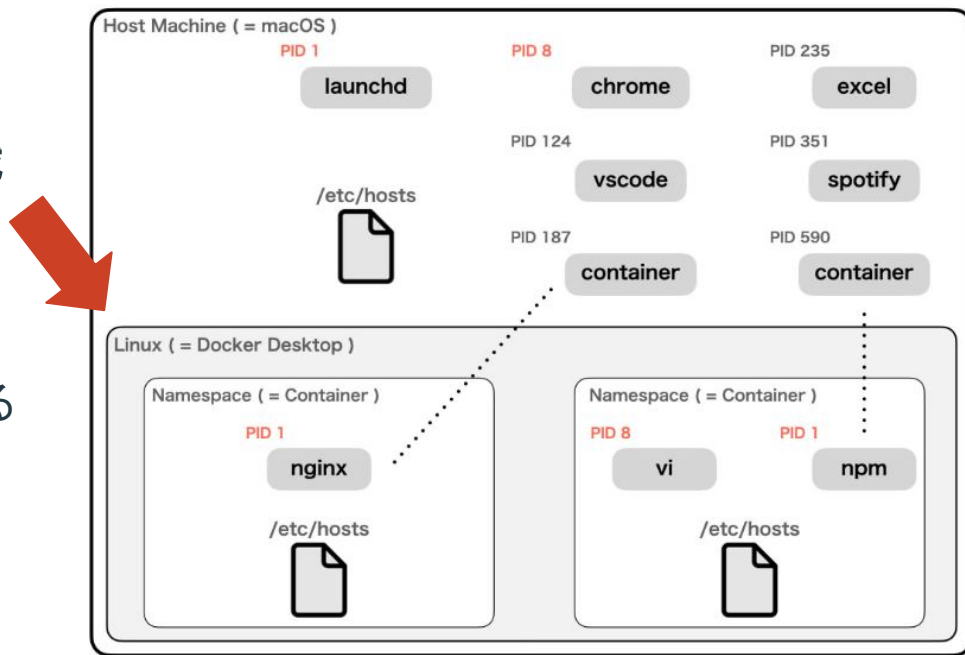
- コンテナ
- イメージ
- Dockerfile



2部 04: Dockerを理解するためのポイント

コンテナとは

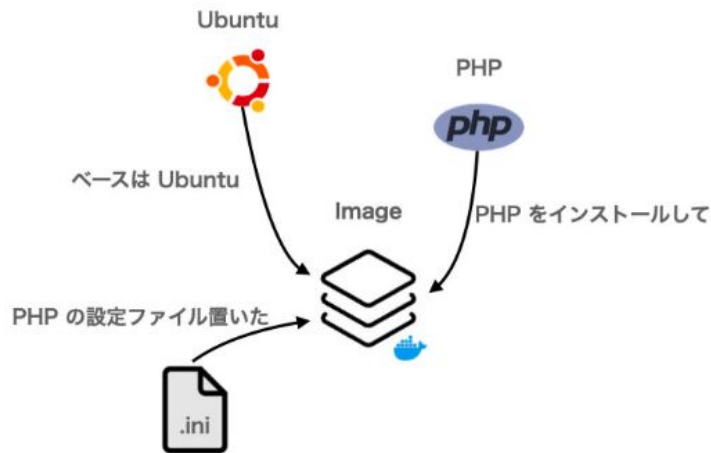
- ホストマシン上の隔離された領域
- namespaceでPIDの衝突回避
 - ROSのnamespaceと似ている
 - 各コンテナの独立が保証



2部 04: Dockerを理解するためのポイント

イメージとは

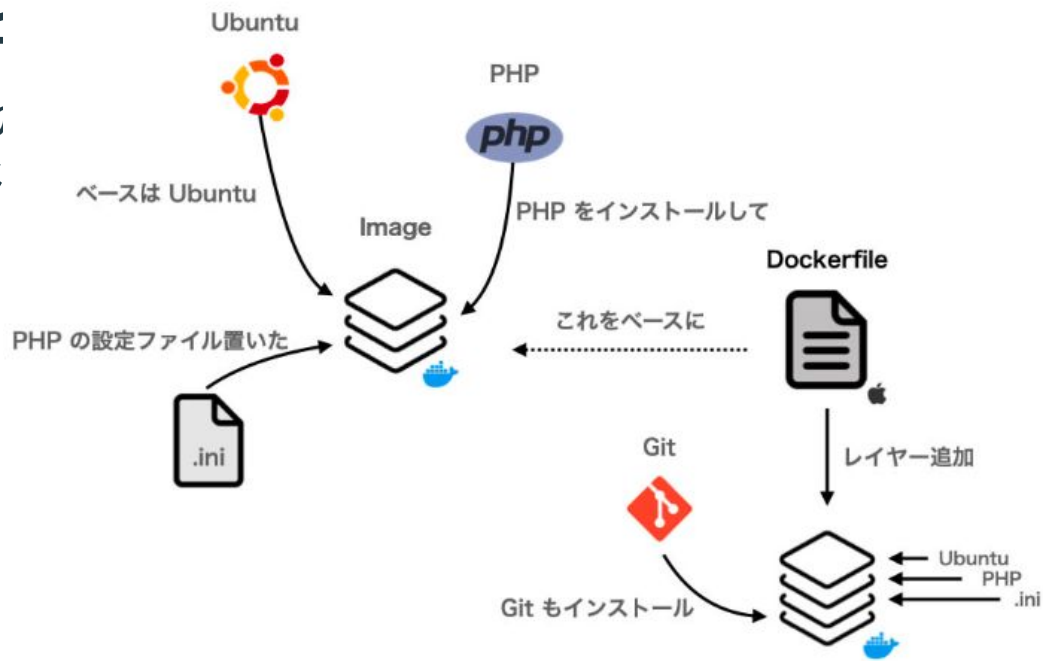
- コンテナを作成するための材料
 - **ファイルやメタ情報を集めたもの**
- イメージの内容
 - ベースはなにか
 - なにをインストールしてあるか
 - 環境変数はどうなっているか
 - どういう設定ファイルを配置しているか
 - ~~デフォルト命令はなにか~~



2部 04: Dockerを理解するためのポイント

Dockerfileと

- イメージの作成
- イメージの管理



2部 04: Dockerを理解するためのポイント

2. 基本のコマンドも 3つ

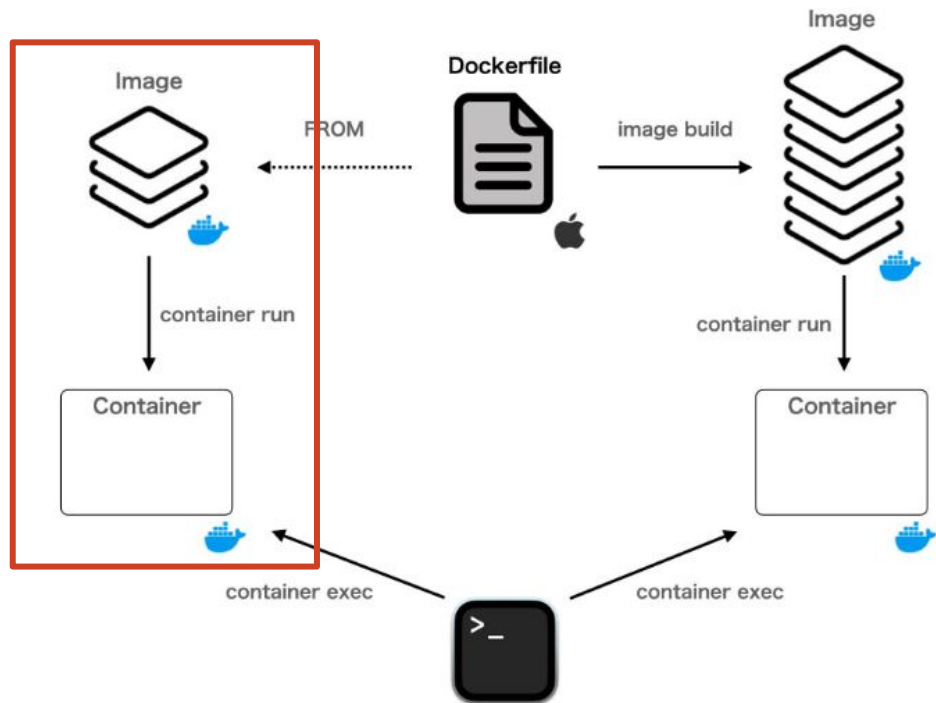
1. コンテナを起動する(docker run)
2. イメージを作る(docker build)
3. コンテナをどうにかする(docker exec)

2部 04: Dockerを理解するためのポイント

2. 基本のコマンドも 3つ

コンテナを起動する

- docker run
- イメージからコンテナを起動

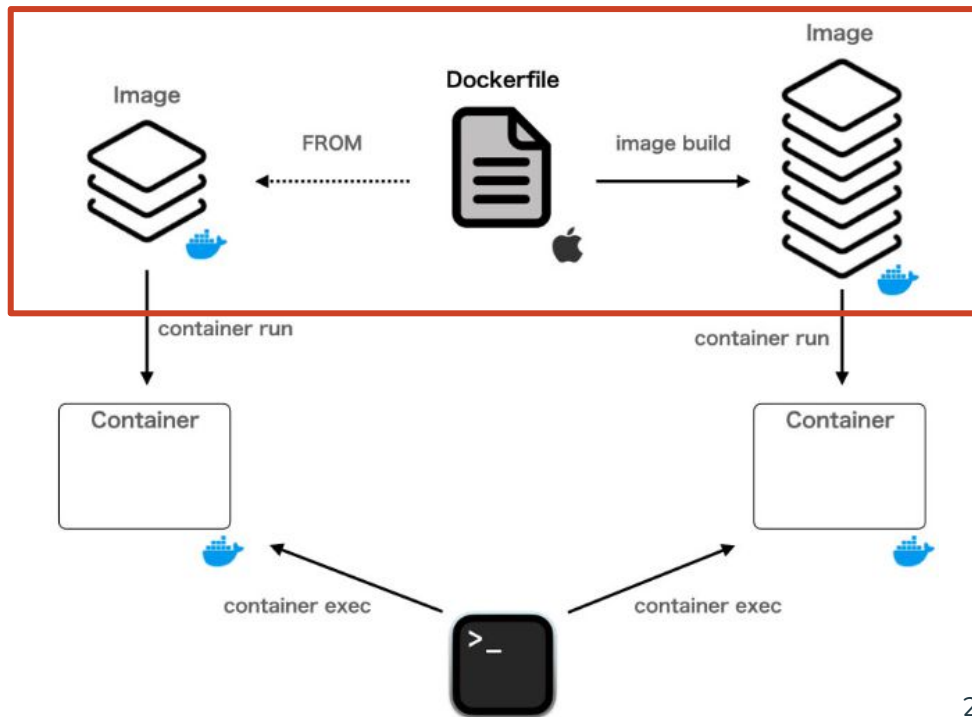


2部 04: Dockerを理解するためのポイント

2. 基本のコマンドも 3つ

イメージを作る

- docker build
- Dockerfileからイメージを作成

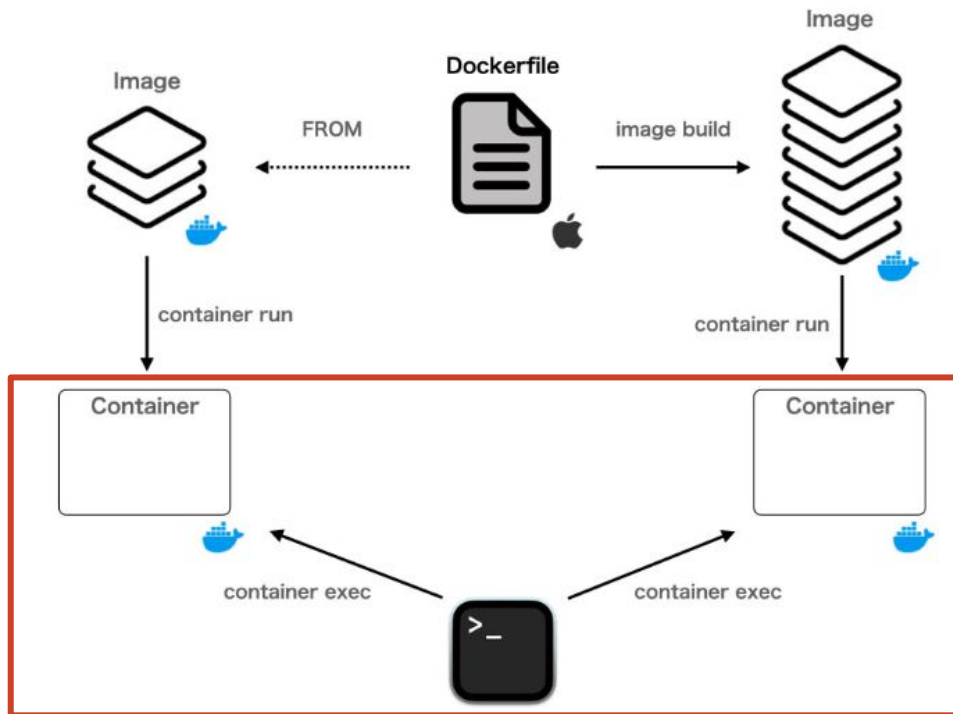


2部 04: Dockerを理解するためのポイント

2. 基本のコマンドも 3つ

コンテナを操作する

- docker exec
- コンテナに命令を送る



2部

まとめ(04: Dockerを理解するためのポイント)

- **基本の要素**

- コンテナ: 隔離されたプロセス
- イメージ: コンテナの情報を集約
- Dockerfile: イメージの設計図

- **基本のコマンド**

- コンテナを起動: `docker run`
- イメージを作成: `docker build`
- コンテナを操作: `docker exec`

2部

05: コンテナの基礎操作

- コンテナを起動する - docker run
- コンテナ一覧を確認 - docker ps
- コンテナを停止 - docker stop
- コンテナを削除 - docker rm



Hands-on !

2部 05: コンテナの基礎操作

コンテナを起動する(docker run)

```
$ docker run [option] <image> [command]
```

Hands-on !



```
$ docker run -it --name ubuntu ubuntu bash
```

2部 05: コンテナの基礎操作

コンテナを確認する(docker ps)

```
$ docker ps [option]
```

Hands-on !



```
$ docker ps -as
```

2部 05: コンテナの基礎操作

コンテナを停止する(`docker stop`)

```
$ docker stop [option] <container>
```

Hands-on !

```
$ docker stop ubuntu
```

2部 05: コンテナの基礎操作

コンテナを削除する(docker rm)

```
$ docker rm [option] <container>
```

Hands-on !

```
$ docker rm ubuntu
```

2部

06: コンテナ起動時の基本の指定

- コンテナを起動する
- ~~● コンテナを対話操作する~~
- ~~● コンテナをバックグラウンドで起動する~~
- コンテナ停止時に自動で削除する
- コンテナに名前をつける
- ~~● コンテナ起動時の挙動を変更する~~
- ~~● (余談) コンテナのOSアーキテクチャを指定する~~

2部 06: コンテナ起動時の基本の設定

コンテナを起動する

オプション	意味	用途
-i (--interactive)	コンテナの標準入力に接続する	コンテナを対話操作
-t (--tty)	疑似ターミナルを割り当てる	コンテナを対話操作
--rm	停止済みコンテナを 自動で削除する	起動時に停止済みコンテナと一意な情報が追突するのを避ける
--name	コンテナに名前をつける	コンテナを指定しやすくなる

2部 06: コンテナ起動時の基本の設定

コンテナを起動する

```
$ docker run -it --name ubuntu ubuntu bash
```

コンテナをターミナルから操作可能に

コンテナの名前を指定

2部 06: コンテナ起動時の基本の設定

コンテナ停止時に自動で削除する

Hands-on !

```
$ docker run -it --rm --name ubuntu ubuntu bash
$ docker ps -a
root@~~~~~: /# exit
$ docker ps -a
```


2部 06: コンテナ起動時の基本の設定

コンテナに名前をつける

Hands-on !

```
$ docker run -it --rm ubuntu bash  
$ docker ps  
root@~~~~~: /# exit
```

2部

07: コンテナの状態遷移(省略)

まとめ

- コンテナはメインプロセス (PID=1) を実行するために起動する
- コンテナが停止する理由
 - コンテナを停止する (docker stop, docker rm など)
 - メインプロセスが終了する (bash -> exit)

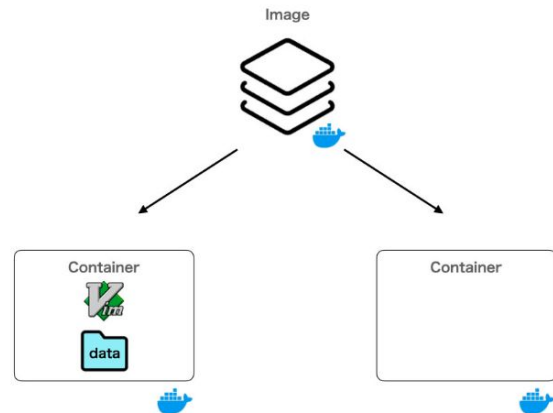
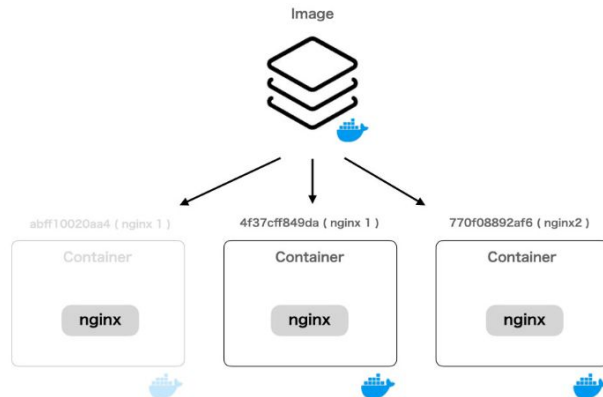
例) `$ docker run -it ubuntu bash`

2部

08: コンテナの状態保持(省略)

まとめ

- コンテナは**起動するたびに違うコンテナ**である
- コンテナの操作は**ほかのコンテナに影響しない**
- 別のコンテナに変更を反映するには...
 - Dockerfile
 - ボリュームやバインドマウント(3部)



2部

09: コンテナに接続する

- コンテナ内でコマンドを実行する (docker exec)
 - 起動中のコンテナ内でコマンドを実行する
 - コンテナに接続する

2部 09: コンテナに接続する

コンテナ内でコマンドを実行する(`docker exec`)

```
$ docker exec [option] <container> command
```

Hands-on !

```
$ docker run -it --rm --name ubuntu ubuntu bash
```

```
$ docker exec ubuntu date
```

```
$ docker exec -it ubuntu bash
```

2部

10: イメージの基礎

- イメージ一覧を確認する(docker images)
- イメージを探すには
- イメージを指定するにはTAGを使う
- ~~● 使うTAGを決めるには~~
- ~~● ローカルにある取得済みのイメージ一覧を確認するには~~
- ~~● どんなイメージか把握するには~~

2部 10: イメージの基礎

イメージ一覧を確認する(docker images)

```
$ docker images [option]
```

Hands-on !

```
$ docker images
```

2部 10: イメージの基礎

イメージを探すには

<https://hub.docker.com/>

The screenshot shows the Docker Hub search results page. The header includes the Docker Hub logo, navigation links (Explore, Repositories, Organizations), a search bar, and user controls. The left sidebar contains filters for Products (Images, Extensions, Plugins), Trusted Content (Docker Official Image, Verified Publisher, Sponsored OSS), Operating Systems (Linux, Windows), and Architectures (ARM, ARM 64, IBM POWER, IBM Z, PowerPC 64 LE, x86, x86-64). The main content area displays search results for '1 - 25 of 10,000 available results.' The results are sorted by 'Suggested'. Four images are shown: 'alpine' (minimal Docker image based on Alpine Linux), 'nginx' (Official build of Nginx), 'busybox' (Busybox base image), and 'ubuntu' (Debian-based Linux operating system). Each result includes the image icon, name, update status, description, supported architectures, pull count, and a 'Learn more' link.

Image Name	Description	Architectures	Pulls
alpine	A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size!	Linux x86-64 ARM ARM 64 386 PowerPC 64 LE IBM Z riscv64	9,366,032
nginx	Official build of Nginx.	Linux unknown ARM ARM 64 386 unknown mips64le PowerPC 64 LE IBM Z x86-64	49,875,825
busybox	Busybox base image.	unknown Linux riscv64 unknown x86-64 ARM ARM 64 386 mips64le PowerPC 64 LE IBM Z	61,797,957
ubuntu	Ubuntu is a Debian-based Linux operating system based on free software.	Linux ARM 64 PowerPC 64 LE IBM Z 386 riscv64 x86-64 ARM	25,793,168

2部 10: イメージの基礎

イメージを指定するには TAGを使う

```
$ docker run [option] <image> [command]
```

ubuntu:latest == ubuntu:22.04

REPOSITORY : TAG

Hands-on !

```
$ docker run --rm ubuntu cat /etc/lsb-release  
$ docker run --rm ubuntu:18.04 cat /etc/lsb-release
```

2部

11: Dockerfile の基礎

省略予定

要望があれば次回解説

(Dockerfileの代わりになるものは次回解説)

次回

範囲: 3部全部

事前準備: DockerHubのアカウント作成

場所: オンライン, 2号館18階会議室

時間: 15:00～(屋外 MTG終わり次第)