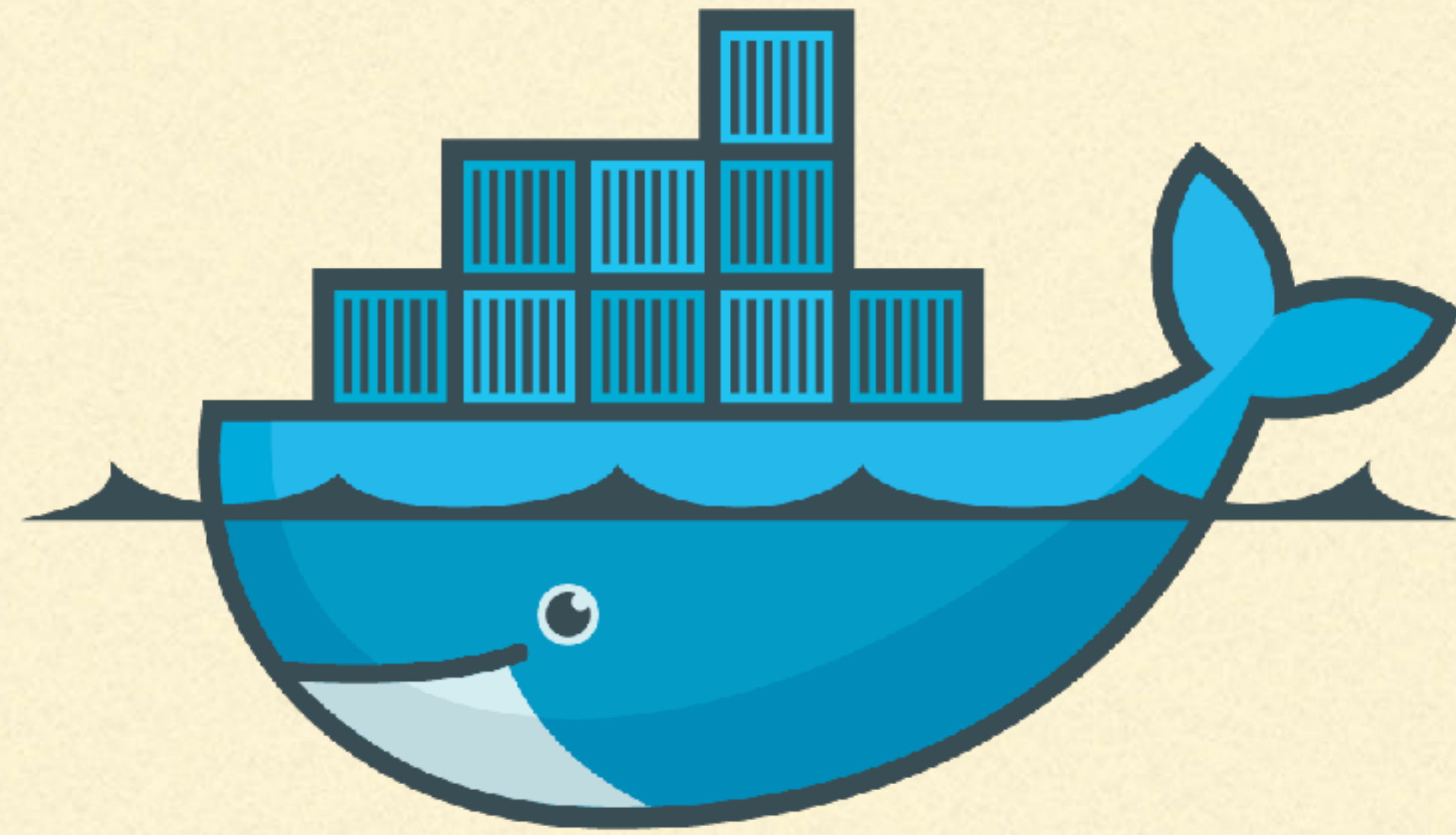

どかどか**Docker**



docker

今日の内容

今日の内容

Dockerって何？

Dockerを使ったことある方

参加者の10割くらいを想定

Dockerとは

Docker社が開発・提供している、Linux用コンテナ管理ソフトウェア（OSS）。

⇒ とりあえず、**Linux用の仮想化ソフト**という認識でOKです。

Demo

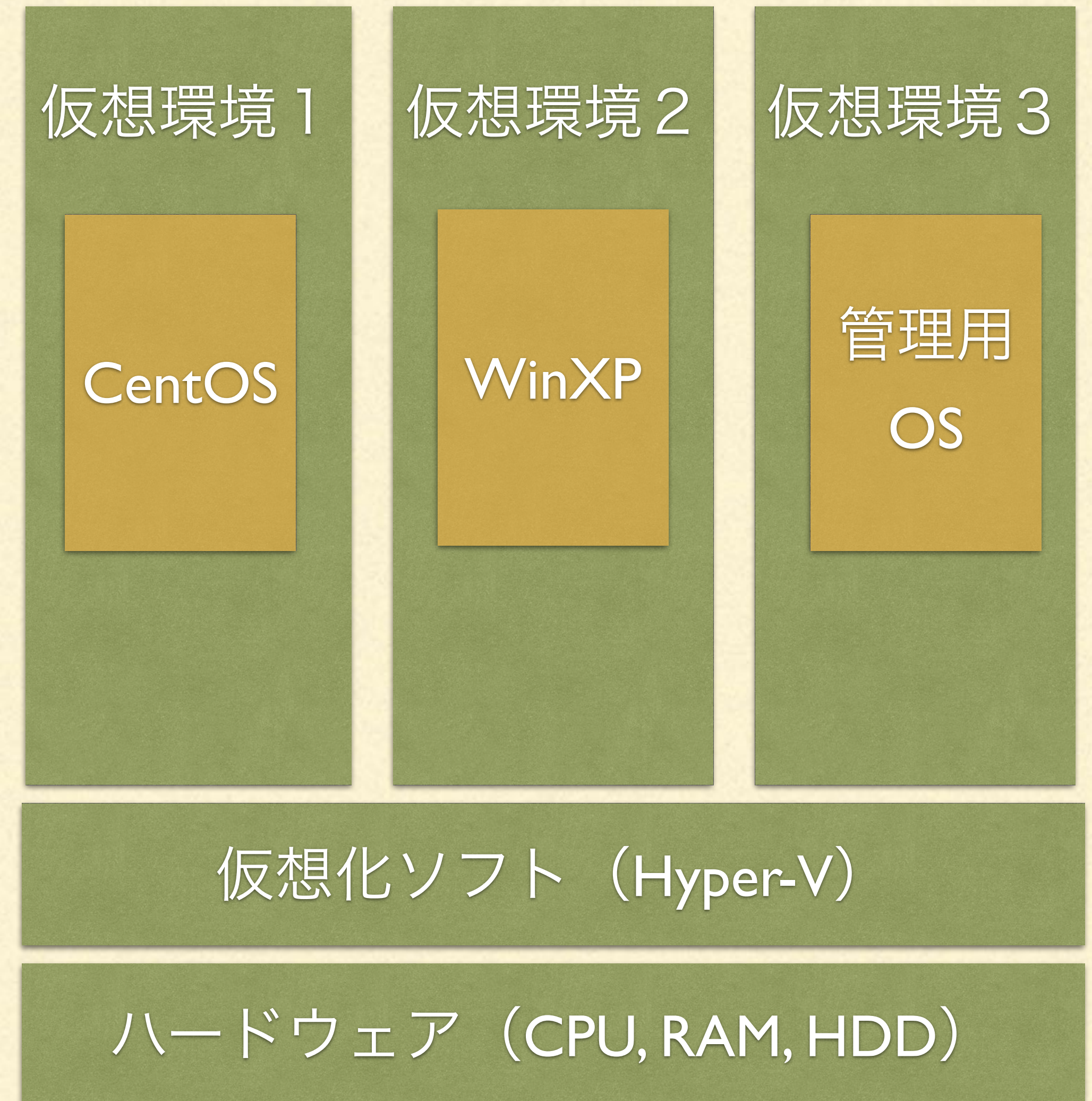
なんでDockerが流行ってるの？

・ ・ ・ の前に、従来の仮想化は....

ホストOS型仮想化



ハイパーバイザ型仮想化



-
- 仮想環境ごとにOSイメージが必要（数GB～数十GB）
 - OSを丸ごと起動するので、リソース（RAM・ROM）の消費が激しい。
⇒ 開発環境だと、複数の仮想環境を作成するのが厳しい。
 - ホストOS型の仮想化の場合、ホストOSとゲストOSが同時に動作しているため動作速度が遅い。（ソースのコンパイル速度が数倍遅くなる等）
-

で、 Dockerは...

- ・ **コンテナ型仮想化**

ホストOS上に「独立したサーバと同様の振る舞いをする区画」 **（コンテナ）** を作成し、それをユーザやサービスに割り当てることで、仮想化を実現する方式。Linuxには「Linux Container（LXC）」という、コンテナ型仮想化の仕組みがデフォルトで提供されている。コンテナ型仮想化自体は、2000年から存在しています。

⇒ **DockerはLinux専用の、コンテナを管理するためのソフトウェア。**

※仮想化そのものは、**Linuxの機能**を使用している。

コンテナ型仮想化のメリット

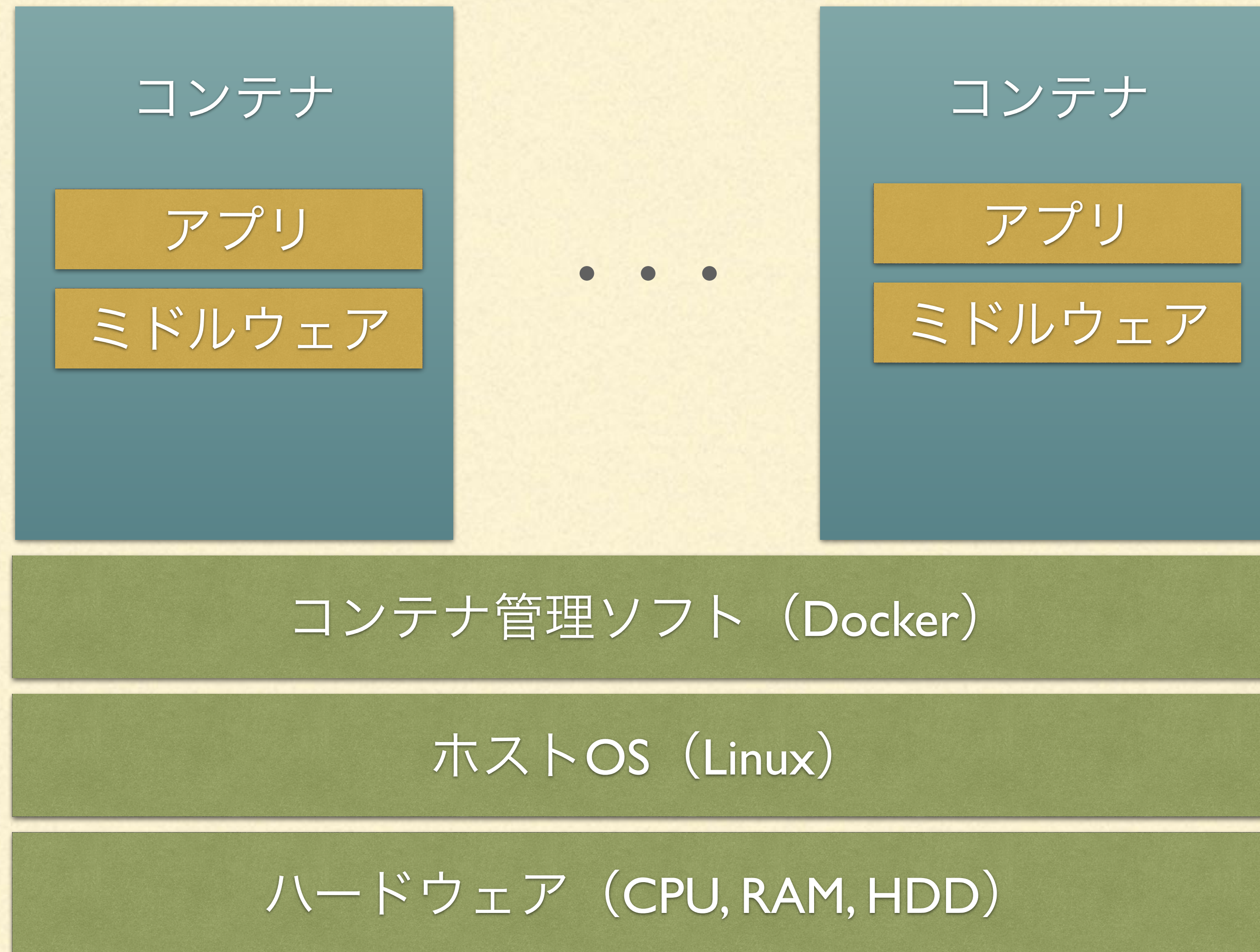
コンテナ型仮想化のメリット

- 個別にCPUやメモリ、ストレージを割り当てないのでゲストOSの起動が速い。
 - リソースを消費しないので、多くのコンテナ（仮想的環境）を作成可能。
-

コンテナ型仮想化のデメリット

- ホストOSと同じOSしか仮想化できない。
⇒ DockerはLinuxしか扱えないので、Windows系のOSは仮想化できない。
 - 一つのOSを共有しているので、ホストOSに対して脆弱性などを突かれた場合複数のコンテナに影響が出てくる。
-

コンテナ型仮想化（**Docker**を使用してコンテナ管理）



で、なんでDockerが流行ってるの？

コンテナの雛形から、
実体を作成できる仕組みが提供されているから！！

Dockerには、コンテナを作成する為の元ネタとして**イメージ**というものがあります。

(ホスト型・ハイパーバイザ型仮想化でいう、OSイメージに近い)

Dockerでは、基本的にこのイメージを元にしてコンテナの作成を行います。

```
takaaki — zsh — 91x9
Last login: Wed Jan 11 22:56:56 on ttys002
~ >>> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mysql                5.7                 78cdcf394b4e       4 days ago         400.2 MB
nginx                stable              c2d83d8cde8d       3 weeks ago        180.7 MB
jenkins              alpine              79539be96238       5 weeks ago        263.4 MB
php                  7-fpm              578f20d5a43c       8 weeks ago        374 MB
busybox              latest              e02e811dd08f       3 months ago       1.093 MB
~ >>>
```

イメージの入手方法には主に、

1. DockerHubから入手する。

2. Dockerfileから生成する。

の2つです。

DockerHubから入手する

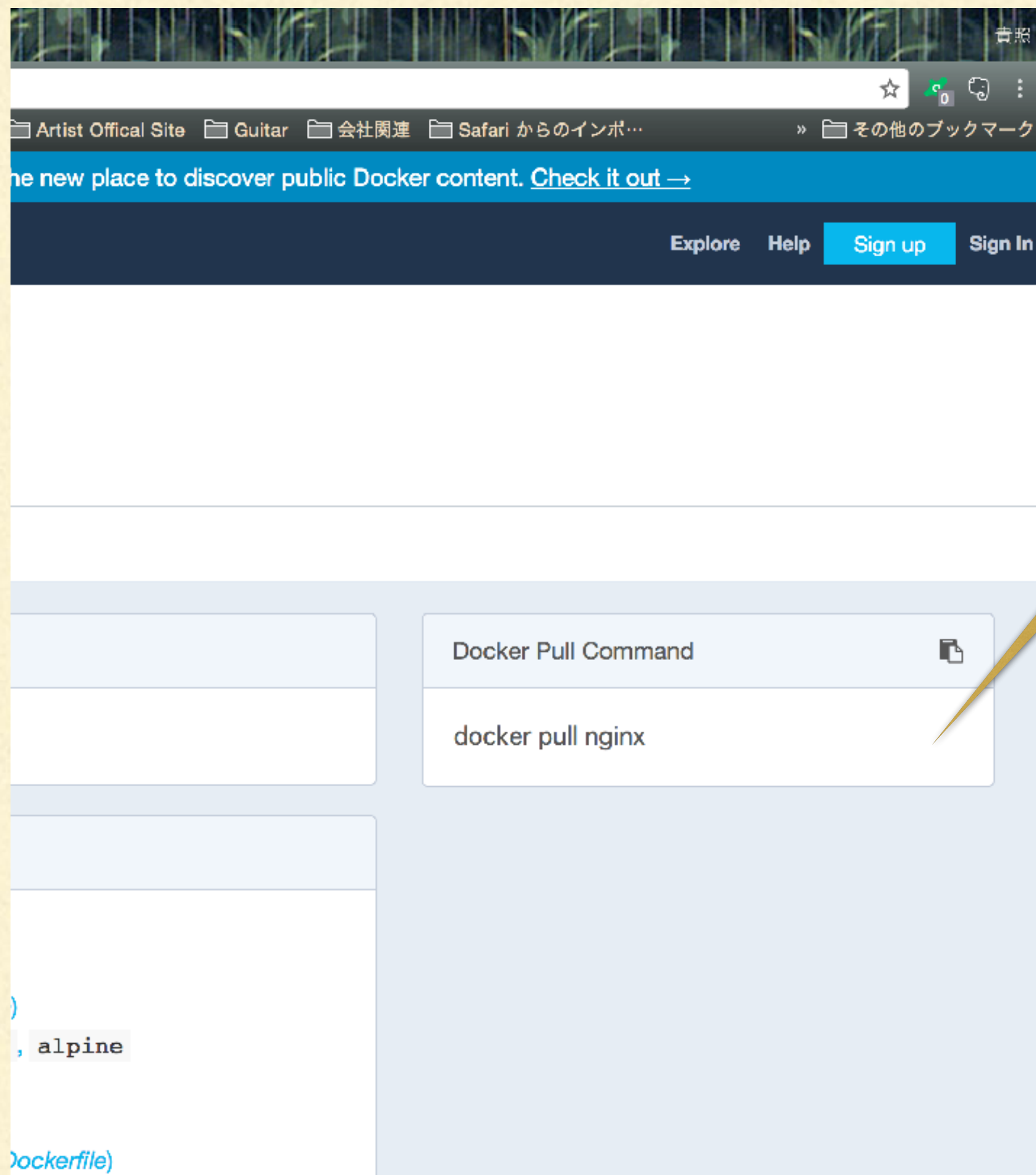
nginxのDockerHubページ

The screenshot shows the Docker Hub page for the official nginx repository. The browser's address bar shows the URL `https://hub.docker.com/_/nginx/`. A blue banner at the top of the page states: "nginx is now available in the Docker Store, the new place to discover public Docker content. [Check it out →](#)". The page header includes a search bar, links for "Explore", "Help", "Sign up", and "Sign In".

The repository is identified as the "OFFICIAL REPOSITORY" for "nginx", with a star icon and the text "Last pushed: 16 days ago". Below this, there are tabs for "Repo Info" (selected) and "Tags".

The "Repo Info" section is divided into two columns:

- Short Description:** Official build of Nginx.
- Docker Pull Command:** `docker pull nginx`
- Full Description:** Supported tags and respective Dockerfile links
 - 1.11.8, mainline, 1, 1.11, latest ([mainline/jessie/Dockerfile](#))
 - 1.11.8-alpine, mainline-alpine, 1-alpine, 1.11-alpine, alpine ([mainline/alpine/Dockerfile](#))
 - 1.10.2, stable, 1.10 ([stable/jessie/Dockerfile](#))
 - 1.10.2-alpine, stable-alpine, 1.10-alpine ([stable/alpine/Dockerfile](#))



Docker Pull Command



```
docker pull nginx
```

DockerHubから入手する場合は、
各ページにコマンドが記載されているので
同様のコマンドを叩けばOK!

Dockerfileから生成する

Dockerfileから生成する

Dockerでは、インフラの構成情報をコード化したファイル（**Dockerfile**）として作成し、そのファイルからイメージを生成することができます。
コード化されているため、作成したDockerfileを他のユーザと共有できます。

⇒この仕組みが、**Docker**が普及した一番の理由だと思っています。
コンテナ型仮想化 + Dockerfileによるインフラ構成の共有化 = バリ便利！

GitHubで公開されている、MySQL8.0のDockerfileの一部

MySQL8.0 - Dockerfile

```
FROM debian:jessie

# add our user and group first to make sure their IDs get assigned consistently, regardless of whatever dependencies get added
RUN groupadd -r mysql && useradd -r -g mysql mysql

# add gosu for easy step-down from root
ENV GOSU_VERSION 1.7
RUN set -x \
    && apt-get update && apt-get install -y --no-install-recommends ca-certificates wget && rm -rf /var/lib/apt/lists/* \
    && wget -O /usr/local/bin/gosu "https://github.com/tianon/gosu/releases/download/$GOSU_VERSION/gosu-$(dpkg --print-architecture)" \
    && wget -O /usr/local/bin/gosu.asc "https://github.com/tianon/gosu/releases/download/$GOSU_VERSION/gosu-$(dpkg --print-architecture).asc" \
    && export GNUPGHOME="$(mktemp -d)" \
    && gpg --keyserver ha.pool.sks-keyservers.net --recv-keys B42F6819007F00F88E364FD4036A9C25BF357DD4 \
    && gpg --batch --verify /usr/local/bin/gosu.asc /usr/local/bin/gosu \
    && rm -r "$GNUPGHOME" /usr/local/bin/gosu.asc \
    && chmod +x /usr/local/bin/gosu \
    && gosu nobody true \
    && apt-get purge -y --auto-remove ca-certificates wget

RUN mkdir /docker-entrypoint-initdb.d
```



```
takaaki — zsh — 101x19
~ >>> docker build -t my-centos7 Develop/docker_demo
Sending build context to Docker daemon 2.048 kB
Step 1 : FROM centos:7
---> 67591570dd29
Step 2 : CMD "echo create CentOS7 image"
---> Running in 4660a54b4e71
---> 699097ae9d57
Removing intermediate container 4660a54b4e71
Successfully built 699097ae9d57
~ >>> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
my-centos7	latest	699097ae9d57	3 seconds ago	191.8 MB
mysql	5.7	78cdcf394b4e	5 days ago	400.2 MB
nginx	stable	c2d83d8cde8d	3 weeks ago	180.7 MB
centos	7	67591570dd29	3 weeks ago	191.8 MB
jenkins	alpine	79539be96238	5 weeks ago	263.4 MB
php	7-fpm	578f20d5a43c	8 weeks ago	374 MB
busybox	latest	e02e811dd08f	3 months ago	1.093 MB

```
~ >>> █
```

my-centos7 という名前で、イメージを生成

イメージが生成されました

Dockerバリ便利

まとめ

Dockerって何？

- ・ コンテナ型仮想化を手軽に行うためのツール。
 - ・ 仮想環境（コンテナ）を作成する元ネタが簡単に入手・共有できるので、便利。
-

参考にした書籍・サイト

～書籍～

- ・ プログラマのためのDocker教科書

～サイト～

- ・ Docker 入門
 - ・ 巷で話題のDockerとは？
 - ・ 【図解】コレ1枚で分かるコンテナ型仮想化とDocker
 - ・ はじめてのDocker - SlideShare
-

ご静聴、ありがとうございました。
