500. 開発・運用環境

秋葉洋哉

2024 年 7 月 15 日

1 Docker

1.1 概要

仮想環境とは、ハードウェア上で独立した複数の環境を実行する技術で、ホスト型、ハイパーバイザー型、コンテナ型の 3 つに分類できる。それぞれは、リソースをどのように共有するかに違いがある。

- ホスト型: ホスト OS の上で仮想環境を実行する。
- ハイパーバイザー型: ハードウェア上で直接実行する。
- コンテナ型: ホスト OS のカーネルを共有して実行する。

Docker は、コンテナ型の仮想環境を提供するツールであり、コンテナ型の仮想環境を構築・運用するためのプラットフォームである。

Docker とは、アプリケーションと依存関係をコンテナとしてパッケージ化したもので、アプリケーションの依存関係をインフラから分離することで、アプリケーションの開発、テスト、デプロイを簡素化することができる仮想環境、と定義できる。

Docker は、エンジン・イメージ・コンテナの 3 つの要素で構成されている。エンジンは、Docker の実行環境であり、イメージは、コンテナの実行に必要なファイルや設定をまとめたものであり、コンテナは、イメージを実行したものである。

1.2 Docker の活用

Docker は、以下のような活用方法がある。

- アプリケーションの開発とテスト: 現代の開発の複雑さを軽減する
- 本番環境でのデプロイ: マシン毎の環境差異を解消する
- マイクロサービスアーキテクチャの実装: システムの柔軟性と耐障害性を向上する
- 環境の統一と再現性の確保: 開発、テスト、本番環境の動作を統一する

1.3 Docker のコマンド

Docker のコマンドは、以下のようなものがある。

- docker run: コンテナを起動する
- docker ps: 実行中のコンテナを一覧表示する
- docker images: ローカルに保存されたイメージを表示する
- docker build: Dockerfile を基にイメージを作成する
- docker pull: イメージを取得する
- docker push: イメージをリポジトリにプッシュする
- docker start: 停止しているコンテナの起動
- docker stop: 実行中のコンテナを停止する
- docker rm: コンテナを削除する

1.4 Dockerfile の主要な記述

Dockerfile は、Docker イメージを構築するためのスクリプトであり、以下のような記述がある。

- FROM: ベースとなるイメージを指定する [例: FROM ubuntu:20.04]
- RUN: コマンドを実行する [例: RUN apt-get update&&apt-get install -y vim]
- COPY: ファイルやディレクトリをコピーする [例: COPY ./app /app]
- CMD: コンテナが起動した際に実行するコマンドを指定する [例: CMD ["python", "app.py"]]
- WORKDIR: 作業ディレクトリを指定する [例: WORKDIR /app]
- EXPOSE: ポートを公開する
- ENTRYPOINT: コンテナが起動した際に実行するコマンドを指定する [例: ENTRYPOINT ["echo"]]
- ADD: ファイルやディレクトリをコピーする [例: ADD https://example.com/file.txt /file.txt]
- ENV: 環境変数を設定する [例: ENV MY_ENV_VARIABLE=value]

.dockerignore ファイルは、Docker イメージをビルドする際に無視するファイルやディレクトリを指定するファイルである。可能な限り、公式のイメージを利用することが推奨される。

1.5 Docker を用いた深層学習モデル開発

Docker は、GPU を利用するための機能を提供している。NVIDIA Container Toolkit を利用することで、Docker コンテナから GPU を利用することができる。深層学習モデルの開発において、Docker を使用することで、GPU だけでなく、データセットやモデルのバージョン管理を行うことができる。

1.6 コンテナオーケストレーション

コンテナオーケストレーションとは、コンテナのデプロイ、スケーリング、運用の自動化プロセスであり、大規模アプリケーションや、マイクロサービスの効果的な管理を可能にする。代表的なコンテナオーケストレーションツールとして、Kubernetes(オープンソース・大規模運用向け) や、docker-compose(開発環境・小規模デプロイ向け) といったものがある。

主な機能としては、以下のようなものがある。

- サービスディスカバリーとロードバランシング: DNS 名や IP アドレスを使って、コンテナを外部に公

開する

- ストレージオーケストレーション: 指定したストレージをコンテナにマウントする
- 自動ロールアウト・ロールバック: コンテナの状態を自動的に管理する
- セルフヒーリング: 障害が発生したコンテナを自動的に修復する