1章 なぜ開発用サーバーが必要なのか？

01 Webアプリの仕組みについておさらいしよう

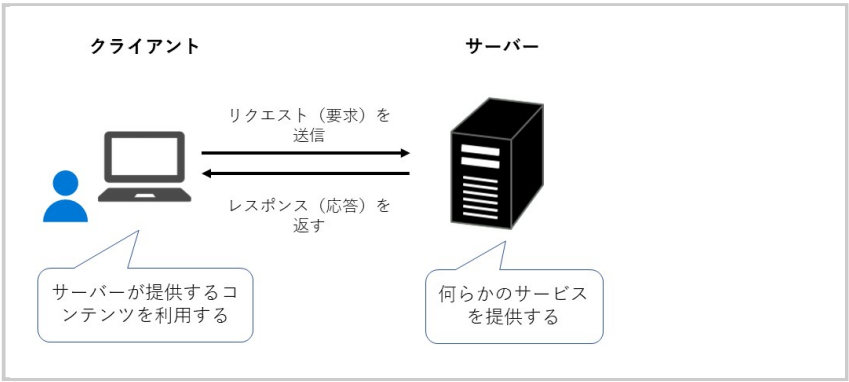
本書はコンテナで簡単にサーバーを調達する方法を解説しますが、サーバーの必要性を理解するためにも、Webアプリの仕組みのおさらいから始めましょう。
#サーバー／#Webサイトの種類／#サーバーが必要な場面を考える

Webはクライアント・サーバーシステム

本書を手に取られた皆さんは、何らかの形でWebの制作／開発に携わっていたり、それらを学んでいたりされている方が多いでしょう。静的なHTMLやCSSで制作するWebデザイナーやコーダーの人もいれば、問い合わせフォームのスクリプトを作っている人、WordPressなどのCMS（Contents Management System）をカスタマイズしている人、さらには、SNSなどのWebアプリを開発している人もいるかもしれません。

ご存じのようにWebとは、クライアントであるWebブラウザと、そこにコンテンツを供給するWebサーバーで構成されるシステムです。これは**クライアント・サーバーシステム**と呼ばれます。

サーバーは、ユーザー（クライアントのコンピューター）からのリクエストに応じて、何らかのサービスを提供するコンピューターやソフトウェアのことです。これに対して**クライアント**は、サーバーに対してリクエスト（要求）を送信することで、サーバーが提供するコンテンツを利用するコンピューターやソフトウェアを指します。



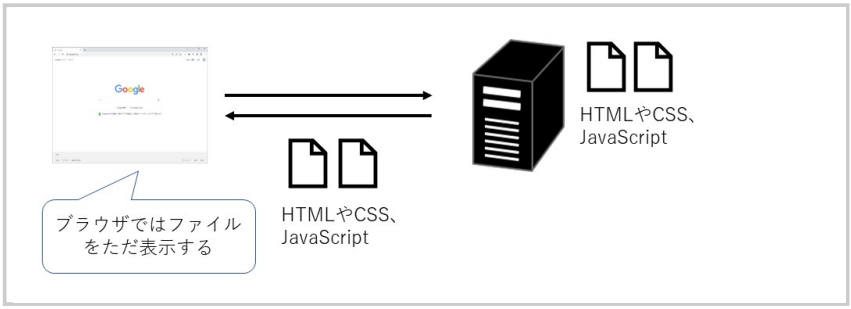
Webサーバーのほかに、データの保存や検索を行う「データベースサーバー」、メールの送受信を担う「メールサーバー」などがあります。

Webサイトには大きく2種類の構成がある

一口にWebサイトといっても、大きく「静的サイト」と「動的サイト」の2通りに分かれます。本書で扱うコンテナやDockerが関わるのは動的サイトのほうです。おさらいとなるかもしれませんが、両者の違いを説明しておきましょう。

静的サイト

静的サイトとは、Webサーバーに保存されているHTMLなどのファイルをそのまま表示するだけのサイトのことです。CSSで装飾していたり、JavaScriptでスライドバナーなどの動く要素を組み込んでいたりしていても、HTML、CSS、JavaScriptの内容が変わることがなければ、静的サイトとなります。そのため、ページの内容を更新したい場合は、サーバーに配置されたファイルを直接書き換える必要があります。このようなサイトを制作する場合は、ローカル（手元のパソコン）にサーバーを用意する必要はありません。Webブラウザのみで検証できます。

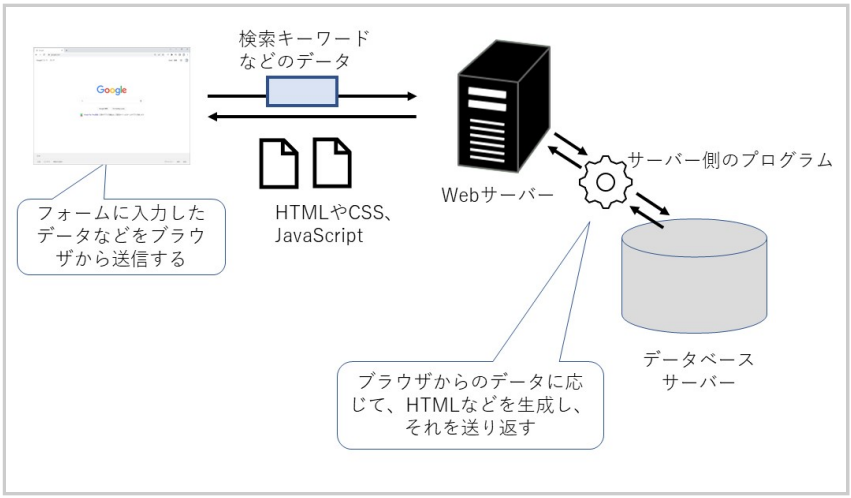


動的サイト

動的サイトは、アクセス時の状況やリクエストの内容によって、表示する内容を変えるサイトのことです。たとえばGoogleなどの検索エンジンは、Webブラウザから送られて来た検索キーワードを見て、そのキーワードを含むWebページの一覧を返してきます。これは動的サイトです。そのほか、ログインの有無によって表示が

変わる会員制サイト、Amazonのように注文を受けて決済や発送を行うECサイト、Twitterなどの SNS サイトも動的サイトです。サイト全体ではありませんが、お問い合わせフォームなども、フォームから受けたデータをメールで送るので、その部分については動的といえます。

動的サイトでは、Web サーバーのほかに、応答を生成するプログラム（独立したアプリサーバーとすることもある）や、データを保存するデータベースサーバーなどが働いています。開発時に動作検証するには、サーバーが必要です。



同期通信と非同期通信

少し話が脱線しますが、動的サイトはさらに「同期通信」型と「非同期通信」型に分かれます。**非同期通信**は、かつては Ajax とも呼ばれ、SPA（Single Page Application）の基本的な仕組みでもあるため、そうと知らずに使っていた人もいるかもしれません。

同期通信とは、古くからある動的サイトの通信方式で、Web ブラウザのフォームからデータが送られたあと、Web サーバーから結果の HTML が送り返されてきて、ページ全体が更新されるものです。

それに対して、非同期通信は JavaScript を使って Web サーバーとの通信を行います。結果の受信も JavaScript が行い、ページ全体を更新するのではなく、ページの一部だけを書き換えます。同期通信に比べてユーザーの操作に対する反応が早いため、先進的な動的サイトで採り入れられています。

動作検証にサーバーが必要となるタイミング

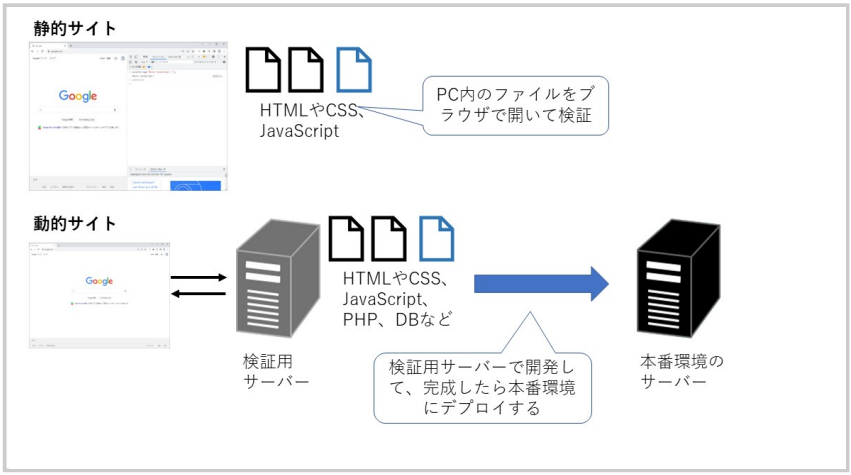
先に説明したように、静的サイトでは、制作時の動作検証にあたってサーバーは必要ありません。Webブラウザで検証しながら制作を進め、完成したらFTPソフトでWebサーバーのストレージにアップロードすれば済みます。

しかし、そのサイトにお問い合わせフォームを付けるとなると、フォームからのデータを受け取ってメールを送信するプログラムを動かすために、サーバーが必要となります。サーバーがなければ、フォームの送信ボタンを押した時点で、エラーが表示されるだけです。

WordPressのようなCMSをカスタマイズするケースでも、CMSは動的サイトの一種である**Webアプリ**なので、サーバーが必要です。新たに独自のSNSを開発するようなケースであれば、サーバーが必要なことはいうまでもありませんね。

また、検証に必要となるサーバーはWebサーバーだけではありません。たいていの場合、PHPやPython、Javaなどのサーバー側プログラムが動く環境、データベースサーバーなども必要です。

このように何らかの動的サイトを開発するためには、まず開発用のサーバーを構築しなければいけません。HTMLやCSS、各種プログラムコードなどを書くのは、そのあとの作業です。



Linuxサーバーなら本番環境に近い検証が可能

Windowsや Macでも、サーバーソフトウェアをインストールすればサーバーとして機能します。代表的なものに XAMPP（ザンプ）があり、これは Webサーバーの Apache、データベースサーバーの MariaDB、プログラミング環境の PHPや Perlなどがセットになったものです。1つインストールするだけで、一般的な動的サイトの検証環境がそろう便利なものですが、いくつか問題もあります。

1つは、普段使いの**パソコンにインストール済みのソフトウェアが原因で、サーバーがうまく動作しない**ケースがあることです。よくあるのが、ポート番号（インターネットにおける通信アプリに割り振られる番号）の衝突で起動しなくなるトラブルです。

2つ目は、**本番環境と OSが違うために細かなトラブルが起き、場合によっては改修が必要となる**点です。インターネット上のサーバーの多くは**Linux**を使用しています。Linuxがよく使われる理由については次のセクションで解説しますが、Linuxと Window/Macでは利用できるソフトウェアが異なるので、微妙な違いが生じることは避けられません。微妙といっても、それが原因で Webアプリが動かなくなることもありうる無視できない問題です。

そのため、開発用サーバーの OSを Linuxにしておくと、事前に Linuxでの検証が行えるので、本番環境へ反映したときのリスクを減らせます。

02 サーバーにLinuxが使われるのはなぜ？

サーバーにLinuxが使われる理由を理解するには、Linuxの特徴を理解することが重要です。
#Linuxの特徴／#ディストリビューション／#Linuxが使われる理由を追う

サーバーに使われる理由はLinuxの特徴にあり

Linux（リナックス）というOSは、1991年に当時フィンランドの学生であったリーナス・トーバルズによって、開発・公開されました。最初は個人的なプロジェクトとして、IBM PC/AT互換機（現在のWindows PC）向けに開発されたLinuxですが、現在では大きく成長を遂げ、Webアプリや組み込みシステム、スーパーコンピューターまで、幅広い用途で利用されています。

サーバーにはLinuxがよく使われます。では、なぜサーバーにLinuxがよく使われるのかというと、Linuxには次の特徴があるからです。

オープンソースかつ無料で利用できる

Linuxはプログラムのソースコードが公開されている、つまり**オープンソース**なので、利用や改変、再配布が自由を行えます。これは、世界中から多くの技術者が開発に携われること、コミュニティによる修正や機能強化がよく行われていることを表します。コードが公開されているので、何か問題があったときに、自分でコードを調べて対策を講じることが可能なのもメリットの1つです。

Linuxは、後述するディストリビューションにもよりますが、基本的には無料で利用できます。サーバー構築の初期費用を抑えられるのに加えて、複数台のサーバーによる構成にも適しています。

安定して稼動できる

サーバー用OSには「落ちない」ことが求められます。WindowsやMacは個人向けのOSであり、使いやすさや対応機器の多さなど、さまざまな要望に応えるよう設計されています。いずれも優れたOSですが、サーバー専用ではなく、安定性は最優先事項ではありません。それに対してLinuxは、カーネル（OSの中核）にソフトウェアの部品を自由に組み合わせて構築することができます。サーバー用途であれば、

GUI（デスクトップ）を取り外し、サーバー専用の構成が取られます。サーバーに不要な要素がない分、ファイルサイズなどの消費リソースも少なくなり、安定性の向上にもつながります。

細やかなユーザーやファイルの管理

サーバーコンピューターは不特定多数のユーザーにアクセスされるため、セキュリティも重要です。Linuxにはファイルやディレクトリ（フォルダー）へのアクセス制限を行う、**パーミッション**という設定があり、ファイルごとにどのユーザーがアクセス可能かを設定できます。Webサーバーであれば、外部からアクセスするユーザーは、特定ディレクトリのHTMLなどを読み取ることしかできないよう制限します。また、サイト制作者がFTPにアップロードする場合も、そのユーザーは特定のフォルダーにしか書き込めません。

Linuxの豊富なディストリビューション

コンテナを使うにあたって、Linuxの**ディストリビューション**についても押さえておきましょう。ディストリビューションとは、OSの中核となるプログラムである**カーネル**に加えて、各種コマンドやライブラリなどをまとめた配布形態のことです。厳密には、「Linux」はOS全体ではなく、カーネルのみを指します。

カーネルはアプリの実行管理や資源管理などを担いますが、カーネルだけあっても何もできないので、必要なライブラリやツールを集めた形にして、ユーザーが使いやすいようにしているというわけです。

ディストリビューションは、企業向けや教育向けなど、用途に合わせて多数開発されています。また、CUIで操作することを想定したディストリビューションだけではなく、画面で操作できるようにGUIが付属したディストリビューションもあります。1つのディストリビューションでGUI版（デスクトップ版）とCUI版（サーバー版）が公開されていることもあります。さまざまなディストリビューションがあるので、目的に合わせて選べます。主なディストリビューションを紹介しましょう。

主なディストリビューション

ディストリビューション	概要
Debian GNU/Linux	コミュニティベースで開発。古くから使われており人気が高い
Ubuntu	Debian GNU/Linuxから派生したディストリビューション。デスクトップ用途でも人気が高い
Red Hat Enterprise Linux	Red Hat社が企業向けに開発。RHELとも呼ばれる
Alpine	シンプルで非常に軽量なので、コンテナでよく使われる

一般的にサーバー用途では、軽量なCUIのみのディストリビューションが使われます。

03 気軽にサーバー構築するなら「コンテナ」を使おう

サーバー構築にはいくつか方法があります。その中でもなぜコンテナなのでしょうか。その理由を追っていきましょう。
#サーバー構築の方法／#コンテナ／#サーバー構築の種類を理解

Linuxサーバーを建てるにはさまざまな方法がある

Linuxサーバーを建てるには、主に次のような方法があります。

- ①物理マシンにLinuxをインストールする
- ②仮想マシンを利用する
- ③レンタルサーバーやクラウドを利用する
- ④コンテナを利用する

1つずつ、概要とメリット／デメリットを説明していきましょう。

①物理マシンにLinuxをインストールする

1つ目は、Linuxを直接、物理マシンにインストールする方法です。この方法では、物理的なマシンを買ってきて、Linuxをインストールし、Linux上で動作させたいアプリやソフトウェアを順番にインストールしていく、という手順が必要です。LinuxはWindowsパソコンにインストールできますが、専用に新たなマシンを用意するのはハードルが高いですし、その上、未対応のハードウェアもある場合も存在するので、どのWindowsパソコンでも100%動くとは限りません。そのため初心者がLinuxの環境を自力で構築するのは、少々ハードルが高いといえるでしょう。

②仮想マシンを利用する

2つ目は、**仮想マシン**を使う方法です。「仮想マシン」という言葉は、聞いたことがある方も多いでしょう。仮想マシンは、仮想化技術によって作られたコンピューターのことで、**仮想化（かそうか）**は、存在していない機器や資源をあたかも存在しているように見せる技術のことです。仮想マシンは、物理マシンに仮想化ソフトウェアをインストールし、そこにさらにOSをインストールして作成します。

仮想マシンは1つの物理マシン上に、複数作成できますし、物理マシン上のOSと

は異なるOSを持つ仮想マシンも作れるので、複数のサーバーを建てることがしやすいです。ただし、OSの上にさらにOSを載せるという関係上、仮想マシンは、後述するコンテナに比べるとオーバーヘッドが大きいというデメリットがあります。この点についての詳細は、後述します（P.■■参照）。

仮想マシンを作成するソフトウェアで有名なものには、Microsoft社のHyper-VやOracle社のVirtualBoxなどがあります。

Hyper-V

VirtualBox

③レンタルサーバーやクラウドを利用する

3つ目は、**レンタルサーバー**を使う方法です。レンタルサーバーとは、利用料金を払うことで、インターネット上の事業者からサーバーを借りられるサービスです。たとえばWordPressのサーバーが欲しい場合、通常はレンタルサーバーを使うことになります。レンタルサーバーには、物理マシンを用意せずとも安定したサーバーをすぐに使い始められるというメリットがあります。また環境構築やメンテナンスは基本的に事業者が行うので、保守の手間を省けるというメリットもあります。しかし、学習や検証したいといったときにレンタルサーバーを使うのは、少々オーバースペックですし、維持費もかかります。また、レンタルサーバーだとPHPやMySQLが特定のバージョンしか使えないなど、環境の選択肢が狭い、というデメリットもあります。

レンタルサーバーに似ている方法としては、クラウドで提供されている仮想サーバー構築サービスを使う方法もあります。たとえば、Amazon Web Servicesというクラウドなら Amazon EC2、Azureというクラウドなら Virtual Machines、Google Cloudというクラウドなら Compute Engineという仮想サーバーサービスがあります。

Amazon EC2

Virtual Machines

Compute Engine

クラウドの仮想サーバーは構築が簡単ですし、利用できるOSが豊富で、レンタルサーバーと比べてカスタマイズしやすいです。また、このサーバーも仮想的なものなので、スペックを変更したり自動でオートスケールしたりを柔軟にできることがメリットです。ただし利用には、クラウドの基本的な知識が必要です。また従量課金制なので、使い方によっては、高額になる可能性もあります。

このように、サーバーを作るにはいくつか方法があります。それぞれに多くの特徴とメリットがあるのでこれらを使ってももちろん問題ありませんが、本書では、コンテナを使って構築していきます。

手軽にサーバー構築するならコンテナが便利

4つ目は、本書においてメインで解説していく「コンテナ」を使う方法です。詳しくは次章以降で説明していきますが、コンテナは、仮想マシンなどに比べて、OSや各種ソフトウェアを導入する手間が圧倒的に少なく、複数台のサーバーを構築するのもとても簡単です。手軽にサーバー構築したい、サーバーはよくわからないというWebクリエイターや新米エンジニアに、うってつけの方法です。たとえば

WordPressのサーバーが欲しい場合、通常はレンタルサーバーを使うことになりますが、コンテナなら、数コマンドの実行ですぐに構築できます。またコンテナなら、WindowsやMac上にも、簡単にLinuxサーバーを用意できます。

もちろん、コンテナを本番環境で動かすにはさまざまな知識が不可欠です。また、コンテナは大規模開発で使うもの、という印象を持っている方もいるかもしれません。しかし、大規模開発でないと使ってはいけない、というわけではなく、「とりあえずサーバーが欲しい」というニーズに、とても適した技術なのです。そのため、ぜひチャレンジしてみましょう。コンテナについては、次章から詳しく解説していきます。

point コンテナを学ぶための基礎知識

コンテナを使うにあたって、Webアプリ開発やネットワークに関する基礎的な用語を押さえておくと、学習しやすくなります。そのためここでは、基礎的な用語について紹介しておきましょう。

コンテナを学ぶ際に知っておきたい用語

項目	内容
IPアドレス	インターネット上の機器に割り当てられる番号。通信先の相手を識別するために使われる
ゲートウェイ	異なるネットワーク同士を接続するための仕組み
ポート番号	通信先のアプリやソフトウェアを識別するための番号
DNS	ドメイン名とIPアドレスの紐づけを担う
NAT	プライベートIPアドレスとグローバルIPアドレスの変換を担う
データベース	データの保存や管理、検索を担うソフトウェア
ライブラリ	汎用的なプログラム・機能をほかの人が使いやすいようにまとめたもの
Webアプリフレームワーク	Webアプリを開発しやすくするための機能や設定ファイルをまとめたもの
API	あるサービス上の機能やデータを使う際のインターフェイス（窓口）

column コラムタイトル

コラムテキスト

コード例

- Dockerfile

```
FROM python:3.10 -------- Pythonイメージ
WORKDIR /code
ADD requirements.txt /code/
RUN pip install -r requirements.txt
CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

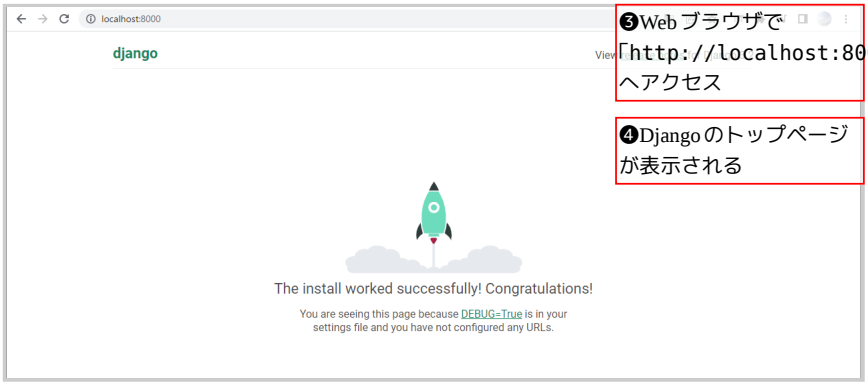
- compose.yaml

```
001 services:
002   db:  -------- PostgreSQLコンテナ
003     image: postgres:14.2
004     environment:
005       POSTGRES_DB: testdb
006       POSTGRES_USER: testuser
007       POSTGRES_PASSWORD: testpass
008     volumes:
009       - db-data:/var/lib/postgresql/data
010   web:   -------- Djangoコンテナ
011     build: .
012     depends_on:
013       - db
014     ports:
015       - "8000:8000"  -------- ポート番号
016     volumes:
017       - .:/code------- ボリューム
018 volumes:        -------- ボリュームの作成
019   db-data:
```

- Ruby

  https://www.ruby-lang.org/ja/

手順サンプル



❸Web ブラウザで「http://localhost:8000/」へアクセス

❹Django のトップページが表示される