

# EC2+Dockerのアプリ

---

大森裕介

# Dockerとは

- アプリケーションごとに仮想環境(コンテナ)を作成し管理するアプリケーション
- 複数のコンテナを管理するdocker-composeを使うことでコンテナの起動、終了を簡単に行うことができる
- コンテナを簡単に削除、作成することができる
- Linuxの仮想環境



## 3つのアプリケーションを作成

1. [Anime Library](#)
2. [ブログサイト](#)
3. [Play Langs](#)

※全てDockerで作成しました。

# Anime Library

---

# Anime Library

[OHMORIYUSUKE/animeapp](https://OHMORIYUSUKE/animeapp)

## Anime Library

アニメの放送時期

2021 / 秋アニメ

✓ 2021 / 秋アニメ が表示されています。



がんばれ同期ちゃん

AtelierPontdarc



月曜日のたわわ2

横浜アニメーションラボ



スター・ウォーズ：ビジョンス

神風動画



ブルーピリオド

Seven Arcs



# 放送時期からアニメを探せるアプリ

TOP画面で放送時期  
からアニメを探し、  
詳細ページで詳細を  
確認することができる  
アプリケーション

Anime Library

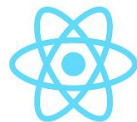


# 仕組み



バックエンド

フロントエンド



Request 放送時期



Flask

Request  
放送時期

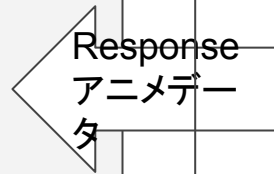
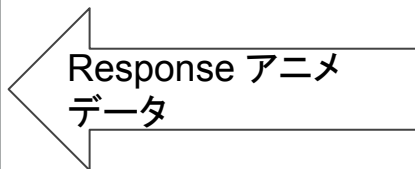
ShangriLa  
Anime API

Response  
アニメデー  
タ

Response アニメ  
データ

画像をスクレイピング

ユーザー



## 頑張ったところ

1. アニメの名前に?,/,=,&が入っているとおかしくなる  
(URLに意図しない記号が入り、想定しないパスに遷移してしまう)
2. ポートでバックエンド、フロントエンドを管理  
(80/tcpをフロントエンド、3031/tcpをバックエンド)



# ブログサイト

---

# ブログサイト

[OHMORIYUSUKE/Blog-PHP-2](#)

## うーたんのブログ

 [HOME](#)

 [Search](#)

 [ABOUT](#)

 [Feed](#)

 [Portfolio](#)

記事 : 1 件

[好きなアニメリスト](#)

1 / 1

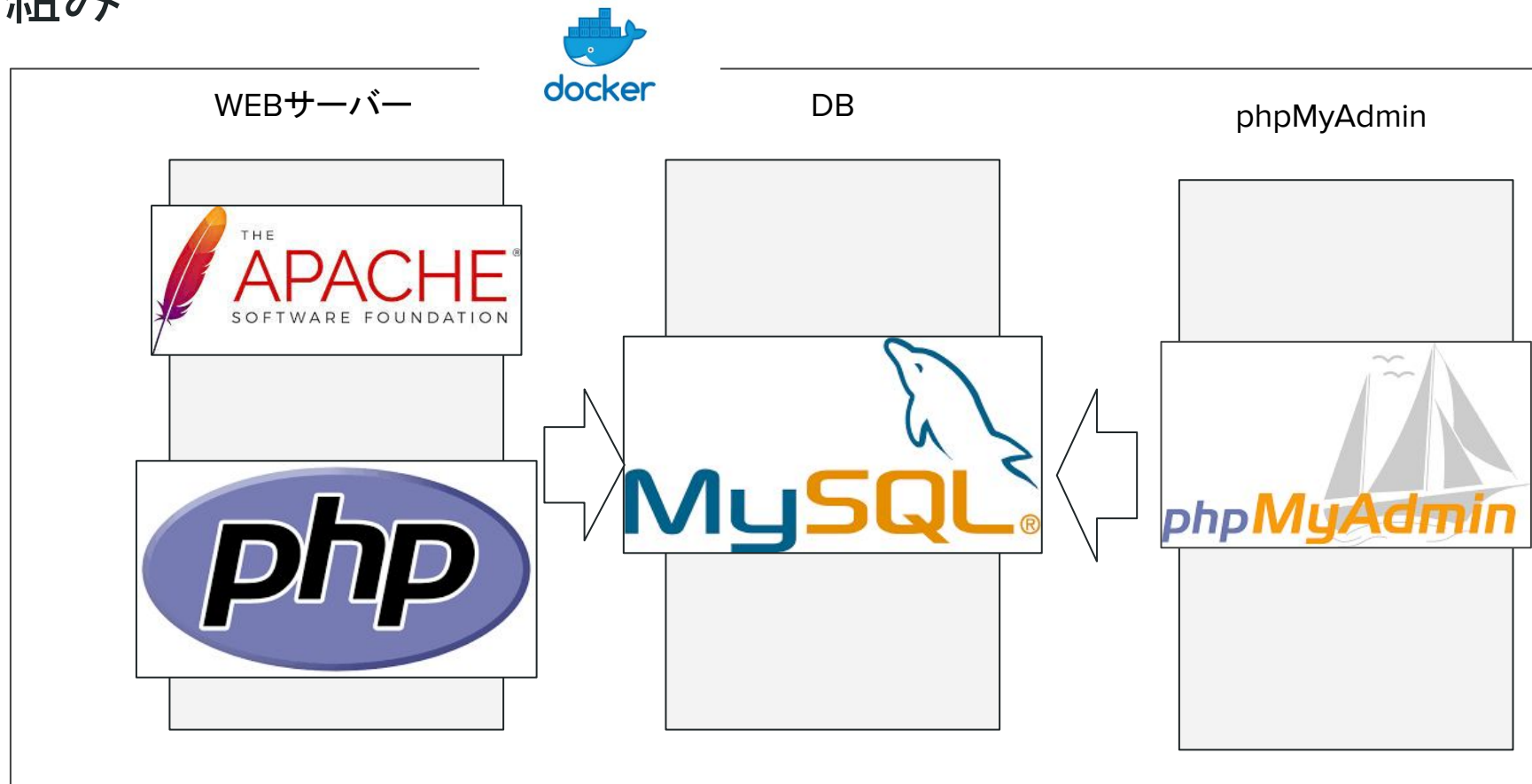
### カテゴリー

- [カテゴリー1](#)
- [カテゴリー1](#)
- [カテゴリー1](#)
- [カテゴリー1](#)
- [カテゴリー1](#)

### アーカイブ

- [アーカイブ1](#)
- [アーカイブ2](#)

# 仕組み



Play Langs

---

# Play Langs [OHMORIYUSUKE/play-langs](https://ohmoriyusuke.github.io/play-langs)

Play Langs

## プログラムを実行しよう !! 🏃

様々なプログラミング言語をオンラインで実行することができます。

実行できる言語



C



Haskell



Java



JavaScript



C++



Ruby

```
1 #include <stdio.h>
2 int main(){
3     printf("Hello World !!");
4     return 0;
5 }
```

```
$ ./a.out
Hello World !!
```

試してみる ▶

# オンラインで言語を実行できるアプリ

## 実行可能な言語

- C
- C++
- Java
- Python
- Ruby
- JavaScript
- Haskell
- Shell
- Go

```
13
14
15 import "fmt"
16
17 func main() {
18     fmt.Printf("hello world Go !!")
19 }
```

標準入力

```
1
```

実行結果

```
hello world Go !!
```

コピー

### 注意事項

3秒以内で実行できるコードにしてください。

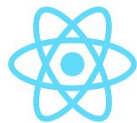
Java を実行する際は、実行クラスの名前を **hello** にしてください。

# 仕組み



バックエンド

フロントエンド



POST ソースコード,  
言語名,入力



Response 実行結果



Flask

uWSGI

shellを実行



## 頑張ったところ

1. 実行時間を設定(3秒以上処理しない)
2. userを作成し、sudoを実行させない
3. パーミッションを設定し、ファイル・ディレクトリを削除させない
4. ReactのためにNginxを設定(root以外でもindex.htmlを参照するように設定)
5. Pythonからshellを実行