



# Next.js×microCMSでコンテンツ を表示するアプリを作ってみよう！

技育CAMPメンター   
宮野 奎太朗

# ▶ はじめに

学生の皆さま、  
今回ご参加いただきありがとうございます！

# ▶ はじめに

# 質問and雑談板で盛り上げていただけると  
嬉しいです！🙏

## ▶ はじめに

**早速書いてみましょう！**

**Q.本日はどちらから参加されていますか？**

# ▶ はじめに

最初に少しでも自己紹介をさせてください👤

# はじめに



【名前】宮野 奎太朗(ミヤケイ)

【出身】青森県青森市🍏

【会社】株式会社サイバー・バズ

【作っているもの】データ可視化ツール(フロント多)

【趣味】ランニング、自転車、サウナ、ゲーム

【Twitter】@38ke1

本日はよろしくお願いします！

# ▶ はじめに

[https://join.slack.com/t/academia0627camp  
/shared\\_invite/zt-1xxo5lt4i-GDGF9TJsPK  
PI6BIVCXHA](https://join.slack.com/t/academia0627camp/shared_invite/zt-1xxo5lt4i-GDGF9TJsPKPI6BIVCXHA)

まだSlackのチャンネルに入っていない方はこちらからどうぞ！

# ▶ はじめに

- **Next.js**初心者の方
- フロントエンドの技術に興味がある方
- **microCMS**を連携した開発に興味がある方



## ▶ はじめに

- ・Next.jsを使ったアプリを作ってみたい
  - ・ReactとNext.jsの**違い**を知りたい
- ・新機能である**App Router**を用いて開発したい

# ▶ はじめに

**App Router**は機能が盛りだくさん  
→今回は基本的な機能を用います！

# 今回のアジェンダ

1. Next.jsの概要について
2. App Routerについて学ぶ(設定ファイル等)
3. microCMSにコンテンツを登録してみよう！  
～休憩(5分程度)～
4. コンテンツを表示してみよう！
5. まとめ

# 今回のゴール

- ・Next.js×microCMSでコンテンツ表示
- ・App Routerの基本的な使い方を知る

# 今回のゴール

- ・Next.js×microCMSでコンテンツ表示
- ・App Routerの基本的な使い方を知る

→自分の手で動かしてみよう👍

## ▶ 今回のゴール

Slack上でサポートいたしますので、  
ご質問お待ちしております！

**それでは早速始めていきましょう！**

# ▶ 1.Next.jsの概要

Reactのフレームワーク  
→レンダリングや画像を最適化し  
てくれる機能等が豊富



# ▶ 1.Next.jsの概要

今回用意したプロジェクトの  
構成を一緒に見てみましょう💪

# ▶ 1.Next.jsの概要

その前に最新のmainブランチからpullしましょう！

# ▶ 1.Next.jsの概要

リポジトリURLはこちら！

# 1.Next.jsの概要

ファイルベースルーティング

が便利！

→ルーティングにパスを書いて、、  
が必要ない

# 1.Next.jsの概要

ファイルベースルーティング

が便利！

→ルーティングにパスを書いて、、  
が必要ない

一度見てみましょう！👁👁

# 1. Next.jsの概要

やってみよう！  
/personalにアクセス  
→文字列が表示されます

# ▶ 1.Next.jsの概要

やってみよう！

- ・ディレクトリとファイルを作成
- ・ページアクセス

# 1.Next.jsの概要

【応用】やってみよう！

- ・ディレクトリとファイル(動的パス)を作成
- ・パラメータを画面とコンソールに表示する



## ▶ 2.App Routerについて

**AppRouter**とは？

→Next.js13で追加された  
新たなファイルシステムルーター

## ▶ 2.App Routerについて

AppRouterで何が変わった？

- Server Components
- コンポーネント内でデータ取得を行ったり、処理が完了する前に描画を行うことが可能に🙌

## ▶ 2.App Routerについて

AppRouterで何が変わった？

- Server Components  
→ つまりはパフォーマンス高  
バックエンドのリソースに  
直接アクセス出来るのがメリット

## ▶ 2.App Routerについて

AppRouterで何が変わった？

- Server Components  
→ デフォルトで適用される 

## ▶ 2.App Routerについて

AppRouterで何が変わった？

- Client Components
- ユーザーの操作や状態を持ちたい  
時に必ず使う  
ファイルトップに「**use client**」  
と明示的に書く

## 2.App Routerについて

```
1  "use client";
2
3  import { ReactNode, useState } from "react";
4
5  export const ToggleButton = (props: { children: ReactNode }) => {
6      const [open, setOpen] = useState(true);
7
8      return (
9          <>
10             <button onClick={() => setOpen(!open)}>{open ? "閉じる" : "開く"}</button>
11             {open && props.children}
12          </>
13      );
14  };
```

## ▶ 2.App Routerについて

AppRouterで何が変わった？

覚えてほしいルール  
Client Componentsから  
Server Componentsは呼び出せない

## ▶ 2.App Routerについて

AppRouterで何が変わった？

ページへのリクエスト時、  
サーバーサイドでレンダリング  
→クライアントサイドで1つのページとして  
組み立てられる。



## ▶ 2.App Routerについて

AppRouterで何が変わった？

つまりClients Componentsから呼び出すと  
サーバーサイドで先にレンダリングが出来ない

## ▶ 2.App Routerについて

AppRouterで何が変わった？

つまりClients Componentsから呼び出すと  
サーバーサイドで先にレンダリングが出来ない

## 2.App Routerについて

What do you need to do?	Server Component	Client Component
Fetch data. <a href="#">Learn more.</a>	✓	⚠
Access backend resources (directly)	✓	✗
Keep sensitive information on the server (access tokens, API keys, etc)	✓	✗
Keep large dependencies on the server / Reduce client-side JavaScript	✓	✗
Add interactivity and event listeners ( <code>onClick()</code> , <code>onChange()</code> , etc)	✗	✓
Use State and Lifecycle Effects ( <code>useState()</code> , <code>useReducer()</code> , <code>useEffect()</code> , etc)	✗	✓
Use browser-only APIs	✗	✓
Use custom hooks that depend on state, effects, or browser-only APIs	✗	✓
Use <a href="#">React Class components</a>	✗	✓

結局どっち使えばいいん？！  
→基本的にはServer Components

## ▶ 2.App Routerについて

ちなみに  
appとpagesディレクトリのパスが競合  
したらどうなるか？  
→appが優先的にレンダリング

## ▶ 2.App Routerについて

app/Layout.tsxは特殊！  
→RootLayoutと呼ばれる。  
\_document.tsxの代替

## ▶ 色々お話しましたが

結論、触るのが一番の近道です。  
実際に手を動かして  
理解を深めましょう！



microCMSにコンテンツを登録してみよう

<https://microcms.io/>

こちらのページにアクセスし、  
登録をお願いします！

# microCMSにコンテンツを登録してみよう

サービスを作成

サービス名

組織名やプロダクト名などを入力してください。後から変更できます。

技育CAMP

サービスID

半角で入力してください。後から変更できます。

geekcamp .microcms.io

サービスを作成する

登録が完了したらサービスを  
作成しましょう！



# microCMSにコンテンツを登録してみよう

## APIを作成

自分で決めるか、テンプレートから選んでAPIを作成しましょう。



次にAPIを作成  
(自分で決めるを選択してください)

# microCMSにコンテンツを登録してみよう

## APIの基本情報を入力

### API名

APIの内容を入力してください。後から変更できます。

食材一覧

### エンドポイント

APIのエンドポイント名を半角で入力してください。後から変更できます。

<https://geekcamp.microcms.io/api/v1/>

ingredients

APIの基本情報を入力  
何でもOKです！👍

# microCMSにコンテンツを登録してみよう

## APIの型を選択



### リスト形式

JSON配列を返却するAPIを作成します。  
ブログやお知らせの一覧、カテゴリー等に  
適しています。



### オブジェクト形式

JSONオブジェクトを返却するAPIを作成  
します。設定ファイルや単体ページ情報  
などの取得に適しています。

## リスト形式を選択

# microCMSにコンテンツを登録してみよう

### APIスキーマを定義

コンテンツID (id) や各種日時 (createdAt, updatedAt, publishedAt, revisedAt) は自動的に作成されます。  
ファイルインポートする場合は[こちら](#)から。

ZplURQAF7v7

フィールドID

ingredients\_name

表示名

食材名

種類

テキストフィールド

必須項目 ☒

IOqzGzanch

フィールドID

ingredients\_image

表示名

食材の画像

種類

画像

必須項目 ☐

90SueLBU

フィールドID

ingredients\_type

表示名

食材の種類

種類

セレクトフィールド

必須項目 ☐

YbdyYCMfz

フィールドID

isDisplay

表示名

表示ステータス

種類

真偽値

必須項目 ☐

+ フィールドを追加

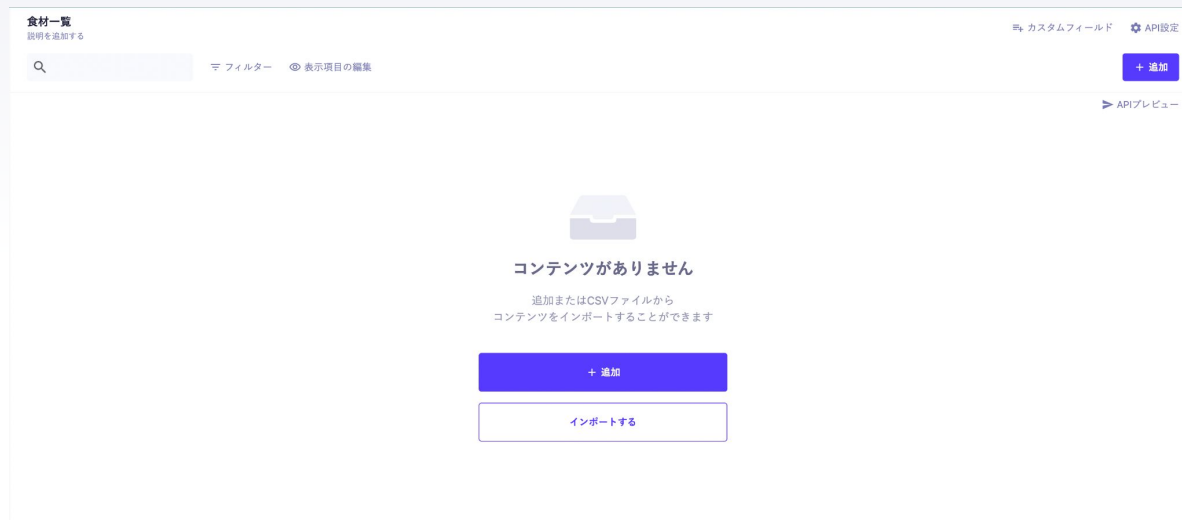
戻る

3 / 3 ステップ

作成

## APIスキーマを定義

# microCMSにコンテンツを登録してみよう



追加ボタンをクリック

# microCMSにコンテンツを登録してみよう


コンテンツID v3y-dmrsug [APIプレビュー](#)

公開日時 2023年6月26日 作成者 最終更新 数秒前 更新履歴

食材名 \*

カルビ

食材の画像



食材の種類 \*

肉

表示ステータス \*

☒

コンテンツを3つ程度登録してみましょう！

# コンテンツ登録完了！

食材一覧  
説明を追加する

カスタムフィールドAPI設定

検索



フィルター

表示項目の編集

+ 追加

1～3件目を表示 / 全3件

APIプレビュー

<input type="checkbox"/>	ステータス	食材名	食材の画像	食材の種類	表示ステータス	
<input type="checkbox"/>	● 公開中	アジ		魚	<input checked="" type="checkbox"/>	⋮
<input type="checkbox"/>	● 公開中	ナス	-	野菜	<input checked="" type="checkbox"/>	⋮
<input type="checkbox"/>	● 公開中	カルビ		肉	<input checked="" type="checkbox"/>	⋮

次は実際に画面表示を試してみましょう！

▶ コンテンツ登録完了！

まずはcurlを叩いて取得してみましょう！





コンテンツ登録完了！

JavaScriptの欄を参考に取得してみましょう！

## X-MICROCMS-API-KEYを.envに追加

**API KEYを追加してみましょう！**



画面上に表示してみよう！

設定が完了したら表示してみましょう！

## microCMS JavaScript SDKのインストール

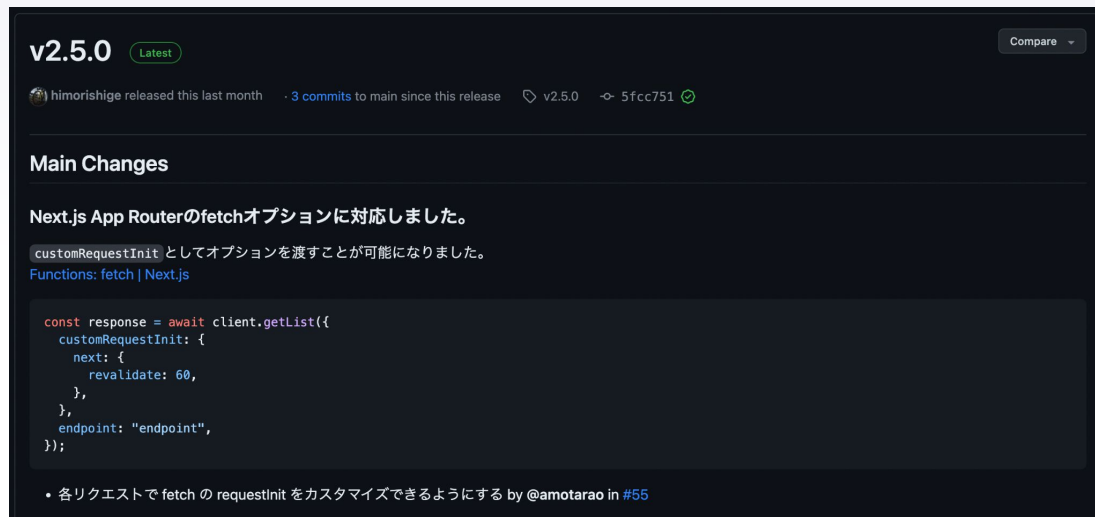
`npm install microcms-js-sdk`

or

`yarn add microcms-js-sdk`

SDKをインストール！

# 画面上に表示してみよう！



The screenshot shows the GitHub release page for Next.js v2.5.0. At the top, it says "v2.5.0" with a "Latest" badge. Below that, it mentions "himorishige released this last month" and "3 commits to main since this release". The "Main Changes" section states: "Next.js App Routerのfetchオプションに対応しました。" (Next.js App Router's fetch option is now supported). It also says "customRequestInitとしてオプションを渡すことが可能になりました。" (It is now possible to pass options as customRequestInit). A code block shows a JavaScript snippet for client.getList() with a customRequestInit object containing a next object with revalidate: 60. At the bottom, there is a note: "各リクエストで fetch の requestInit をカスタマイズできるようにする by @amotaroo in #55".

v2.5.0 Latest

himorishige released this last month · 3 commits to main since this release v2.5.0 5fcc751

### Main Changes

Next.js App Routerのfetchオプションに対応しました。

customRequestInitとしてオプションを渡すことが可能になりました。

Functions: [fetch](#) | [Next.js](#)

```
const response = await client.getList({
  customRequestInit: {
    next: {
      revalidate: 60,
    },
  },
  endpoint: "endpoint",
});
```

• 各リクエストで fetch の requestInit をカスタマイズできるようにする by @amotaroo in #55

## オプションが渡せるように！

## Day.jsをインストール

```
npm i dayjs
```

**Day.js**とは？

→ 日付フォーマットを操作出来るライブラリ

## next.config.jsに以下を追加

```
You, 今 | 1 author (You)
1  /** @type {import('next').NextConfig} */
2  const nextConfig = {
3    images: {
4      domains: ['images.microcms-assets.io']
5    }
6  }
7
8  module.exports = nextConfig
9
```

▶ 表示完了！

表示することは出来ましたか？  
→出来た方はぜひ雑談板にコメントを👏



## ▶ まとめ

今回はNext.js×microCMSの構成で  
アプリを作りました！

## まとめ

今回はNext.js×microCMSの構成で  
アプリを作りました！

→まずはお疲れ様でした！  
感想をチャンネルに書いていただけると  
嬉しいです！

# まとめ

## 今後のチャレンジTODO

1. 簡単なブログアプリを作ってみよう
2. ページネーションを実装してみよう
3. 編集機能を実装してみよう

## ▶ まとめ

皆さま本日はお疲れ様でした！👏