WORKBOXを使った実践的CACHE APIの使い方

WorkboxはオープンソースのCACHE API用ライブラリです。細かく条件を指定してキャッシュを行えます。この章では、Workboxを使ってすでに設定されているキャッシュの設定を変更してみましょう。

Workboxのインストール

Workboxはsw.jsに次のように記述してインストールできます。次の内容をsw.jsの一番最初の行に記述してください。

```
importScripts('https://storage.googleapis.com/workbox-cdn/releases/4
```

Workboxを試す

Workboxでキャッシュを行うため、`install` イベントで実行されているキャッシュ処理を削除します。

```
// 空にします
self.addEventListener('install', event => {
});
```

例えば `http://127.0.0.1:8887` 以下のアクセスをすべてキャッシュする場合は次のように記述します。

```
workbox.routing.registerRoute(
  /127.0.0.1:8887/,
  workbox.strategies.staleWhileRevalidate()
);
```

キャッシュ戦略の違い

Workboxでは複数のキャッシュ戦略を用意しています。

- workbox.strategies.staleWhileRevalidate()
- workbox.strategies.CacheOnly()
- workbox.strategies.CacheFirst()
- workbox.strategies.NetworkFirst()
- workbox.strategies.NetworkOnly()

`workbox.strategies.staleWhileRevalidate()` は最初にキャッシュを返し、もしなければネットワークアクセスしてくれるモードです。一番よく使われる形式になります。

細かくカスタマイズする

例えばスタイルシートだけをキャッシュしたいならば、次のように記述します。

```
workbox.routing.registerRoute(
  /\.css$/,
  workbox.strategies.staleWhileRevalidate()
);
```

有効期限を指定する

キャッシュに有効期限を設けられます。以下は1週間のキャッシュになります。

```
workbox.routing.registerRoute(
  /\.js$/,
  new workbox.strategies.CacheFirst({
    cacheName: 'js-cache',
    plugins: [
      new workbox.expiration.Plugin({
        maxAgeSeconds: 7 * 24 * 60 * 60,
        maxEntries: 10,
      }),
    ]
  })
);
```