CORS (オリジン間リソース共有)

CORSまたは「オリジン間リソース共有」は、ブラウザで実行されているフロントエンドにバックエンドと通信するJavaScriptコードがあり、そのバックエンドがフロントエンドとは異なる「オリジン」にある状況を指します。

オリジン

オリジンはプロトコル (`http` 、 `https`) とドメイン (`myapp.com` 、 `localhost` 、 `localhost.tiangolo.com`) とポート (`80` 、 `443` 、 `8080`) の組み合わせです。

したがって、以下はすべて異なるオリジンです:

- `http://localhost`
- `https://localhost`
- `http://localhost:8080`

すべて `localhost` であっても、異なるプロトコルやポートを使用するので、異なる「オリジン」です。

ステップ

そして、ブラウザ上で実行されているフロントエンド (`http://localhost:8080`) があり、そのJavaScriptが `http://localhost` で実行されているバックエンドと通信するとします。(ポートを指定していないので、ブラウザはデフォルトの `80` ポートを使用します)

次に、ブラウザはHTTPの `OPTIONS` リクエストをバックエンドに送信します。そして、バックエンドがこの異なるオリジン (`http://localhost:8080`) からの通信を許可する適切なヘッダーを送信すると、ブラウザはフロントエンドのJavaScriptにバックエンドへのリクエストを送信させます。

これを実現するには、バックエンドに「許可されたオリジン」のリストがなければなりません。

この場合、フロントエンドを正しく機能させるには、そのリストに `http://localhost:8080` を含める必要があります。

ワイルドカード

リストを `"*"` (ワイルドカード) と宣言して、すべてを許可することもできます。

ただし、Bearer Tokenで使用されるような認証ヘッダーやCookieなどのクレデンシャル情報に関するものを除いて、特定の種類の通信のみが許可されます。

したがって、すべてを正しく機能させるために、許可されたオリジンの明示的な指定をお勧めします。

`CORSMiddleware` の使用

FastAPI アプリケーションでは `CORSMiddleware` を使用して、CORSに関する設定ができます。

- `CORSMiddleware` をインポートします。
- 許可されたオリジンのリストを (文字列として) 作成します。
- これを「ミドルウェア」として **FastAPI** アプリケーションに追加します。

以下も、バックエンドに許可させるかどうか指定できます:

- クレデンシャル情報 (認証ヘッダー、Cookieなど) 。

- 特定のHTTPメソッド (`POST` 、 `PUT`) またはワイルドカード `"*"` を使用してすべて許可。
- 特定のHTTPヘッダー、またはワイルドカード `"*"` を使用してすべて許可。

```
{!../../../docs_src/cors/tutorial001.py!}
```

`CORSMiddleware` 実装のデフォルトのパラメータはCORSに関して制限を与えるものになっているので、ブラウザにドメインを跨いで特定のオリジン、メソッド、またはヘッダーを使用可能にするためには、それらを明示的に有効にする必要があります

以下の引数がサポートされています:

- `allow_origins` - オリジン間リクエストを許可するオリジンのリスト。例えば、 `['https://example.org', 'https://www.example.org']` 。 `['*']` を使用して任意のオリジンを許可できます。
- `allow_origin_regex` - オリジン間リクエストを許可するオリジンの正規表現文字列。例えば、 `'https://.*\.example\.org'` 。
- `allow_methods` - オリジン間リクエストで許可するHTTPメソッドのリスト。デフォルトは `['GET']` です。 `['*']` を使用してすべての標準メソッドを許可できます。
- `allow_headers` - オリジン間リクエストでサポートするHTTPリクエストヘッダーのリスト。デフォルトは `[]` です。 `['*']` を使用して、すべてのヘッダーを許可できます。CORSリクエストでは、 `Accept` 、 `Accept-Language` 、 `Content-Language` 、 `Content-Type` ヘッダーが常に許可されます。
- `allow_credentials` - オリジン間リクエストでCookieをサポートする必要があることを示します。デフォルトは `False` です。
- `expose_headers` - ブラウザからアクセスできるようにするレスポンスヘッダーを示します。デフォルトは `[]` です。
- `max_age` - ブラウザがCORSレスポンスをキャッシュする最大時間を秒単位で設定します。デフォルトは `600` です。

このミドルウェアは2種類のHTTPリクエストに応答します...

CORSプリフライトリクエスト

これらは、 `Origin` ヘッダーと `Access-Control-Request-Method` ヘッダーを持つ `OPTIONS` リクエストです。

この場合、ミドルウェアはリクエストを横取りし、適切なCORSヘッダーと共に情報提供のために `200` または `400` のレスポンスを返します。

シンプルなリクエスト

`Origin` ヘッダーのあるリクエスト。この場合、ミドルウェアは通常どおりリクエストに何もしないですが、レスポンスに適切なCORSヘッダーを加えます。

より詳しい情報

CORSについてより詳しい情報は、Mozilla CORS documentation を参照して下さい。

!!! note "技術詳細" `from starlette.middleware.cors import CORSMiddleware` も使用できます。

FastAPI は、開発者の利便性を高めるために、`fastapi.middleware` でいくつかのミドルウェアを提供します。利用可能なミドルウェアのほとんどは、Starletteから直接提供されています。