第１３回　API作成

１３．API 作成

１３－１． APIとは

Application Programming Interfaceの略であり、アプリケーションやソフトウェアを繋ぐインターフェイスのことを指します。

１３－２．REST APIとHTTPメソッド

REST は Representational State Transfer の略で、REST APIとはREST アーキテクチャスタイルの設計原則に従う アプリケーション・プログラミング・インタフェース (API) です。RESTの設計で作ったAPI（REST API）では、APIのエンドポイントの役割ごとにHTTPメソッドを使い分けるのが普通です。

■代表的なHTTPコマンド

|  |  |
| --- | --- |
| メソッド | 用途 |
| GET | 指定したリソースを取得する |
| POST | 新しいリソースを作成する |
| PUT | 既存のリソースを更新する |
| DELETE | 既存のリソースを削除する |

【注意事項】

今回は課題でPOSTメソッドでの挙動を実行したいことと、処理の難易度の兼ね合いから、検索の際にPOSTを使用しています。

**今回の課題は都合上、RESTの原則からは外れていますので、それは念頭に置いていてください。**

通常POSTメソッドはCreateの際に用います。

１３－３．今回のシステム仕様

【処理概要】

■入力画面（HTMLファイル）　※PHPでは作成しない。



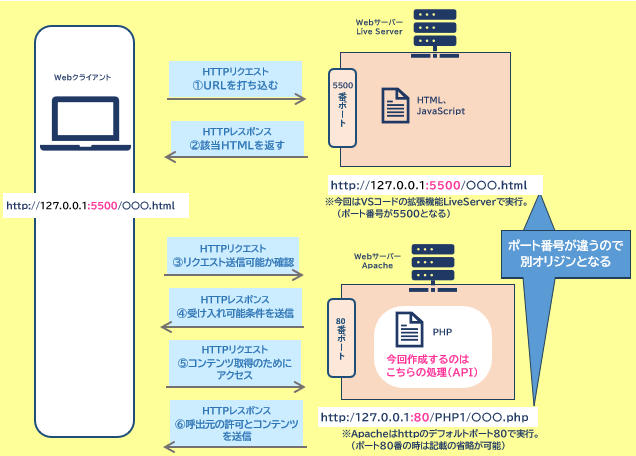
1. 「検索」ボタン押下時、JavaScriptで「キーワード」を送信し、API（PHPファイル）を呼び出す。
2. PHPファイルは、「キーワード」に入力された値を含む商品を検索し、結果をJSON形式にてレスポンスを返す。PHPは処理のみ。
3. JavaScriptにてPHPからのレスポンス（JSON）を動的に画面へ表示する。

【システム構成】

**画面（フロント）側は、デスクトップにフォルダを作成し、VSCodeの拡張機能LiveServerにて起動します。PHP（サーバーサイド）側は前回までの授業と同じフォルダ内にファイルを作成します。すなわち、フロントとサーバーサイドが別オリジンで動作する環境を作ります。（オリジンについては後述）**

**※概要図は以下に示します。**

■概要図



１３－４．オリジンとは

ウェブコンテンツのオリジン (Origin) は、アクセスするために使われる URL の下記によって定義されます。

■ スキーム （プロトコル）

■ ホスト （ドメイン）

■ ポート番号

すべて一致した場合のみ、２つのオブジェクトは同じオリジンであると言えます。

今回の構成では、フロント側とサーバー側でプロトコルとドメインは同じですが、ポートが違うため同一オリジンとはみなされません。

オリジンが同一ではない場合、操作によっては同一オリジンポリシーによって制限が掛かります。この制約は CORS を使用して緩和することができます。



１３－５．CORS（Cross-Origin Resource Sharing）

追加の HTTP ヘッダーを使用して、あるオリジンで動作しているウェブアプリケーションに、異なるオリジンにある選択されたリソースへのアクセス権を与えるようブラウザに指示するための仕組みです。ウェブアプリケーションは、自分とは異なるオリジン (ドメイン、プロトコル、ポート番号) にあるリソースをリクエストするとき、オリジン間 HTTP リクエストを実行します。

**レスポンスヘッダーに、Access-Control-Allow-Originの指定**を行います。レスポンスヘッダーはheader関数を用います

■header　 Access-Control-Allow-Origin

グラフィカル ユーザー インターフェイス, テキスト

自動的に生成された説明

１３－６．プリフライトリクエスト

CORS のプリフライトリクエストは CORS のリクエストの一つであり、事前にブラウザから自動で送信されるものです。サーバーが対象のリクエストを受け付ける許可をしているかを確認するために送信されます。（単純リクエストの場合は省略されます。）

プリフライトリクエストは、OPTIONS リクエストです。

つまり、サーバー側で $\_SERVER["REQUEST\_METHOD"] の値を取得した際には、値は OPTIONS となります。

今回作成するAPIは単純リクエストではないので、このプリフライトリクエストに対するレスポンスヘッダーの指定をしていない場合、APIでコンテンツを返す処理にたどり着く前にエラーとなってしまいます。

プリフライトリクエストに応じたレスポンスヘッダーの設定を行う必要があります。

1. HTTPステータス２００　※ステータスについては後述
2. Access-Control-Allow-Originの指定　※セキュリティを鑑みて「＊」にしないこと。
3. Access-Control-Allow-Headersの指定。　※設定値は下記に記載。

■header　 Access-Control-Allow-Headers

テキスト

自動的に生成された説明

１３－７．HTTPステータス

■header HTTPステータス

テキスト が含まれている画像

自動的に生成された説明

■HTTPで利用可能な主なHTTPステータス

|  |  |  |
| --- | --- | --- |
| 分類 | HTTPステータス | 意味 |
| 100（情報） | 100 Continue | 接続可能 |
| 200（成功） | 200 OK | 成功 |
| 201 Created | 成功（サーバー側に新しいリソースを生成） |
| 202 Accepted | 受付完了（未処理） |
| 300（リダイレクト） | 301 Moved Permanently | リソースが恒久的に移動した |
| 302 Found | リソースが一時的に移動した |
| 303 See Other | リソースが別の場所に存在する |
| 304 Not Modified | リソースが変更されていない |
| 400（クライアントエラー） | 400 Bad Request | 不正なリクエスト |
| 401 Unauthorized | HTTP認証を要求 |
| 403 Forbidden | アクセスを拒否 |
| 404 Not Found | リソースが見つからない |
| 405 Method Not Allowed | HTTPメソッドが不許可 |
| 407 Proxy Authentication Required | プロキシで認証の必要がある |
| 408 Request Time-out | リクエストタイムアウト |
| 500（サーバーエラー） | 500 Internal Server Error | サーバーエラー |
| 501 Not Implemented | 応答に必要な機能が未実装 |
| 503 Service Unavailable | HTTPサーバーが利用不可 |

１３－８． POSTによる値の取得

サーバーサイド側にて、JavaScriptからJSON形式でPOSTされたデータを取得するには、file\_get\_contents関数を使用します。

■file\_get\_contents ※JSONのPOSTデータ取得

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

１３－９． Content-Typeの設定

JSON形式でデータを返却する方には、json\_encode関数を使用します（12回目の資料参照）が、JSONデータそのものだけではなく、レスポンスヘッダーにおいては、Content-Typeでコンテンツの実際の種類を伝える必要があります。

■header Cotent-type



　※JSON形式で文字コードはUTF-8なので、構文のままでOK。