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。

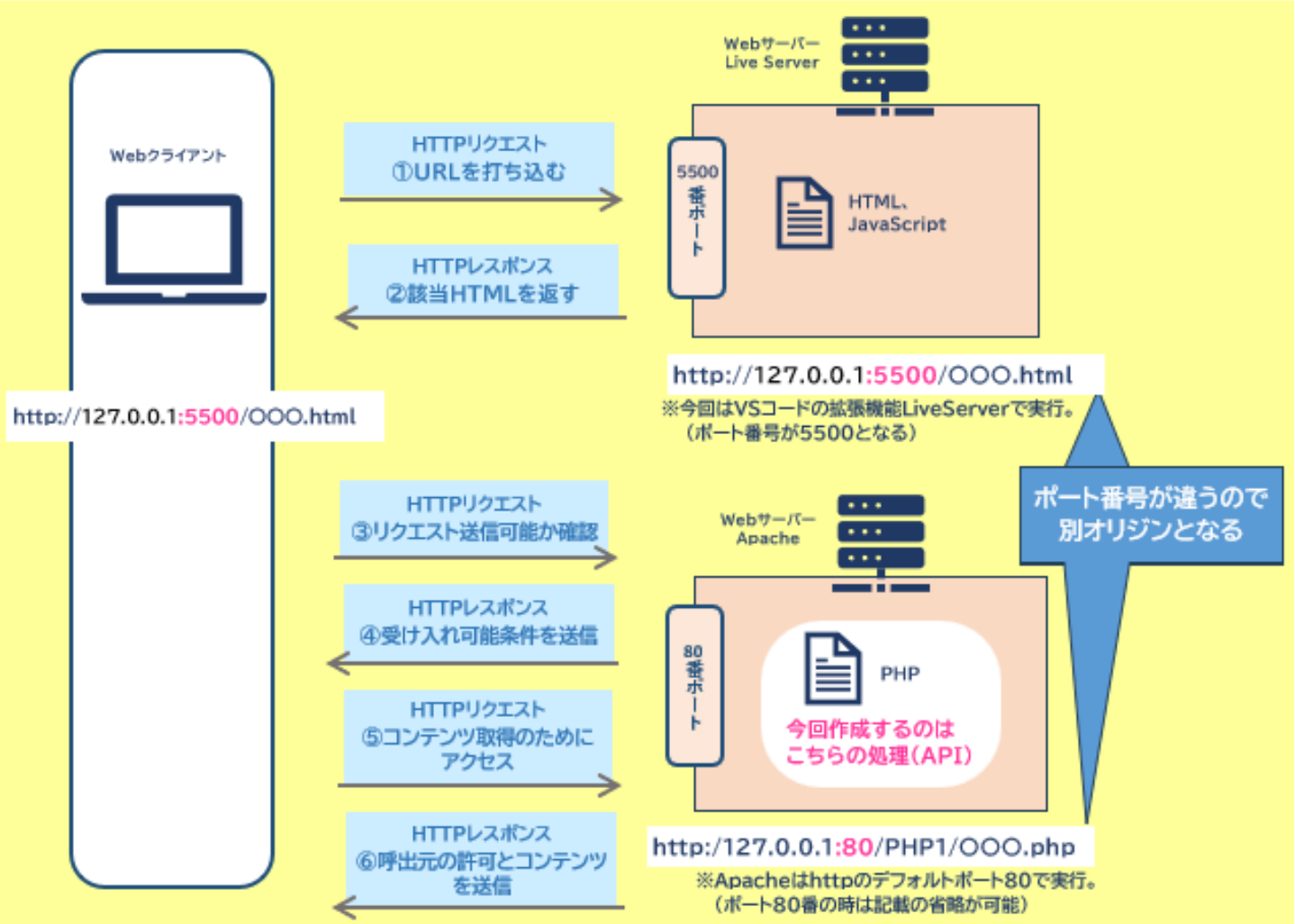
③ JavaScript にて PHP からのレスポンス（JSON）を動的に画面へ表示する。

【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

```
header( "Access-Control-Allow-Origin: origin " )
```

origin … 許可したいオリジンを指定。
「＊」(ワイルドカード)を指定するとあらゆるオリジンからのアクセスを許可する。

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

```
header( "Access-Control-Allow-Headers: header-name " )
```

header-name … 対応しているリクエストヘッダーの名前です。
ヘッダーはコンマで区切って、任意の数のリストにすることができる。

＊今回は、「Origin」「X-Requested-With」「Content-Type」「Accept」の4つをカンマ区切りで指定します。

１３－７．HTTP ステータス

■header HTTP ステータス

```
header( "HTTP/1.1  statusCode  statusMessage " )
```

プロトコルとバージョン　ステータスコード　ステータスメッセージ

それぞれ、半角スペースで区切る。

＊プロトコルとバージョンは構文のままとなる。

＊「ステータスコード」と「ステータスメッセージ」は下記の表より適したものを指定する。

■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

```
file_get_contents( ”php://input ”)
```

戻り値： 読み込んだデータ。(失敗した場合に false)

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

```
header( ”Content-type: application/json; charset=utf-8 ” )
```

※JSON 形式で文字コードは UTF-8 なので、構文のままで OK。