

PHP基礎

Contents

- webの仕組み
- PHP概要
- PHP基礎
 - 変数など
 - 練習
- サーバへデータ送信
 - getとpost
- ファイルへデータを書き込む
- 課題発表→P2Pタイム

rules...

- 授業中は常にエディタを起動！
- 考えたことや感じたことはslackのガヤチャンネルでガンガン発信！
- 質問はslackへ！ 他の人の質問にも目を通そう！（同じ質問があるかも）
- 演習時，できた人はスクショなどslackに貼ってアウトプット！
- まずは打ち間違いを疑おう！
 - `{'";` など
- 書いたら保存しよう！（よく忘れる！）
 - `command + s`
 - `ctrl + s`

PHPの準備

以下3点ができているか確認しよう！

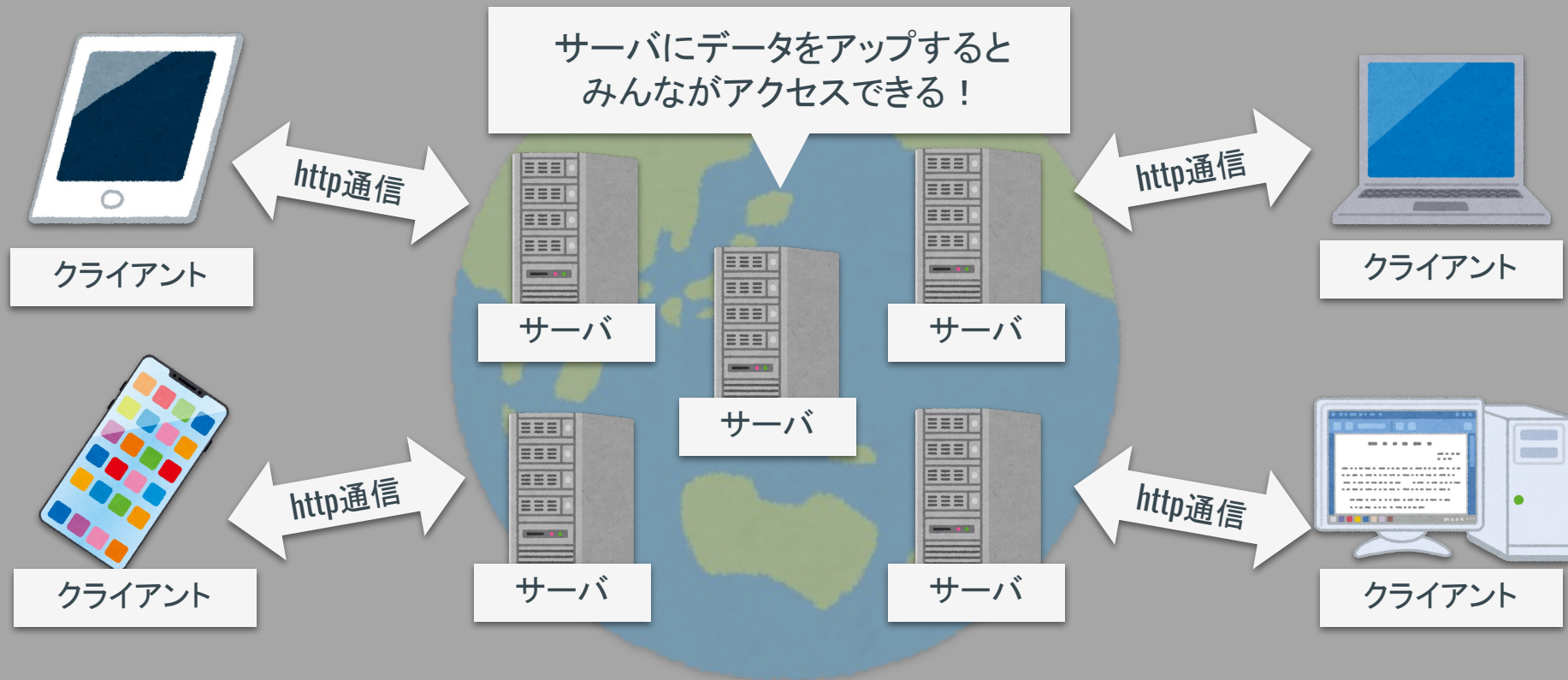
- XAMPPの起動確認
- <http://localhost/>のアクセス確認
- サンプルフォルダを「htdocs」フォルダに入れる

Goal

- webの仕組みを把握(大事)！
- PHP(の開発手順)に慣れる！
- データの送受信を知る！

webの仕組み

雑なwebの仕組み

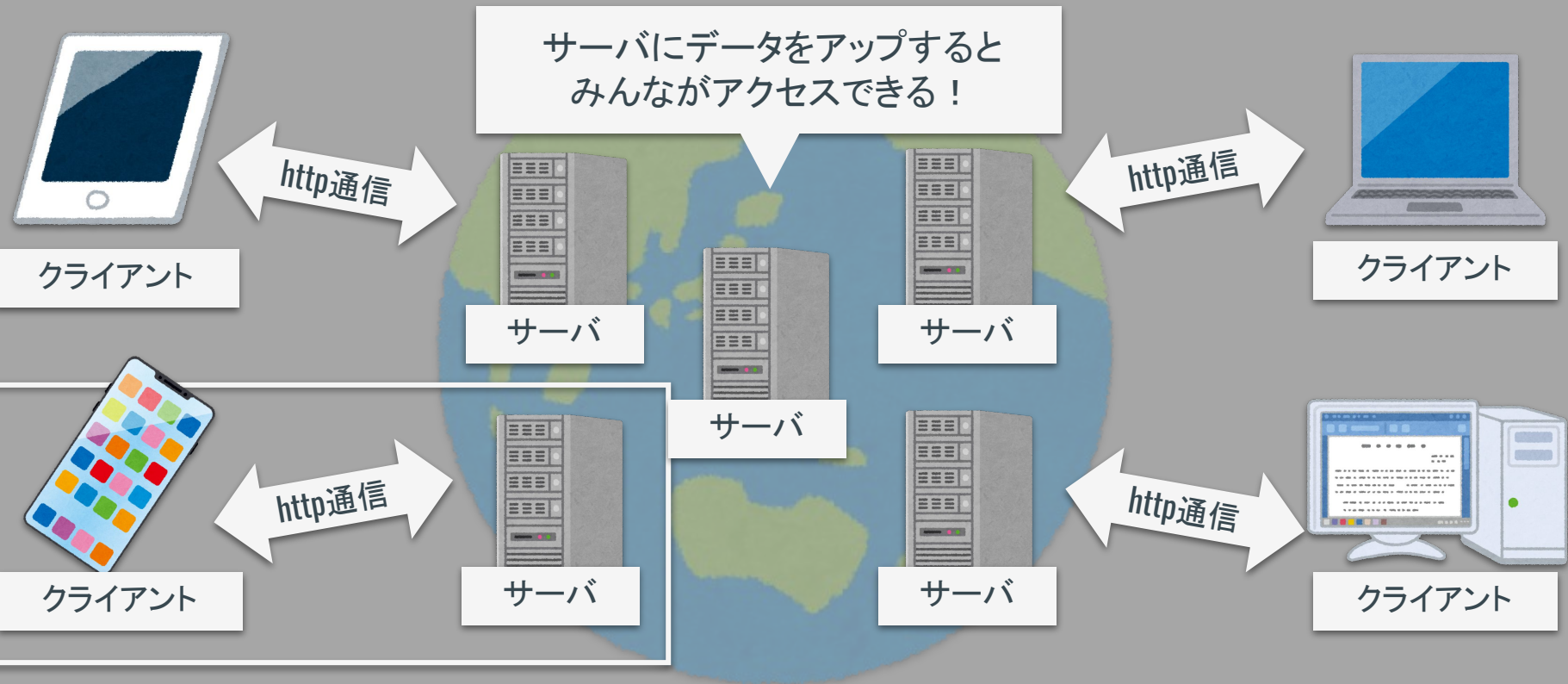


URL

- URLとは
 - web上にある情報(ファイル)の場所を指し示す住所.
 - Uniform Resource Locatorの略(覚えなくてOK).
- 例



雑なwebの仕組み



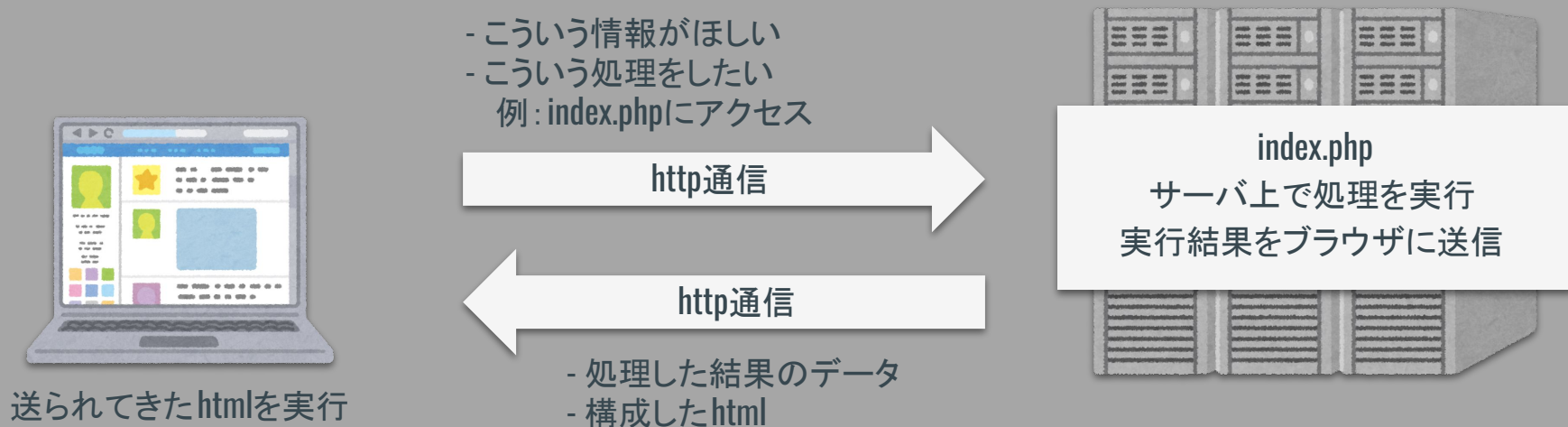
サーバとクライアント

- サーバで動作する言語(サーバサイド)
 - サーバ上でプログラムが実行される.
 - PHP, ruby, python, JAVA, (node.js), etc...
- クライアント(webブラウザ)で動作する言語(クライアントサイド)
 - webブラウザがプログラムを実行する.
 - html, css, javascript

サーバ - クライアント型のアプリケーション

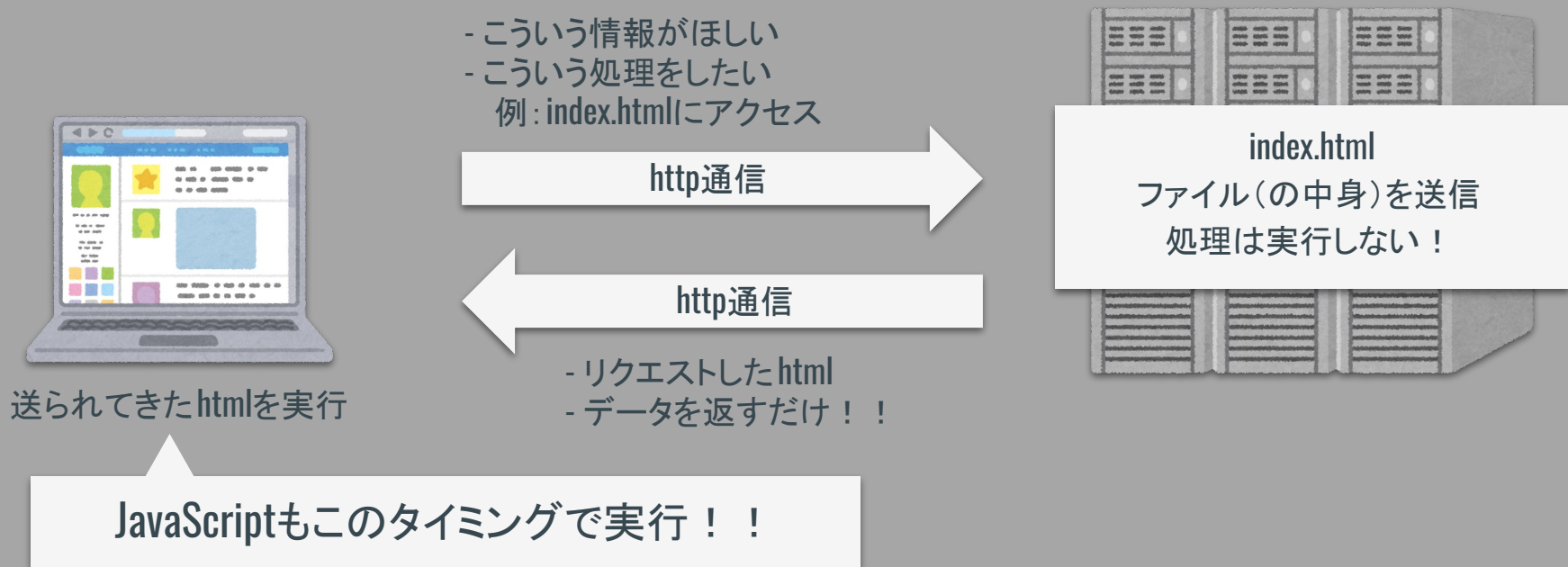
サーバサイド言語の動き方

※ 言語によらず, ファイル(プログラム)はサーバ上に存在



クライアントサイド言語の動き方

※ 言語によらず、ファイル(プログラム)はサーバ上に存在



PHP概要

サーバサイドで実行されること

例

- twitter => ツイート, 検索, タイムラインの表示, etc
- facebook => 投稿, 検索, 記事の更新, コメント, etc
- wordpress => ブログ記事の投稿, 編集, 削除, etc

※必ずしもPHPで作られているわけではない！！

サーバサイドで実行されること

例

- twitter => ツイート, 検索, タイムラインの表示, etc
- facebook => 投稿, 検索, 記事の更新, コメント, etc
- wordpress => ブログ記事の投稿, 編集, 削除, etc

データの「作成」「参照」「更新」「削除」

サーバサイドで実行されること

例

- twitter => ツイート, 検索, タイムラインの表示, etc
- facebook => 投稿, 検索, 記事の更新, コメント, etc
- wordpress => ブログ記事の投稿, 編集, 削除, etc

「Create」「Read」「Update」「Delete」

サーバサイドで実行されること

例

- twitter => ツイート, 検索, タイムラインの表示, etc
- facebook => 投稿, 検索, 記事の更新, コメント, etc
- wordpress => ブログ記事の投稿, 編集, 削除, etc

「CRUD」

PHP基礎

準備(ソースコードの整形)

vs codeに下記の拡張機能をインストール



PHP IntelliSense felixfbecker.php-intellisense

Felix Becker |  7,913,625 |  | [Repository](#) | [License](#)

Advanced Autocompletion and Refactoring support for PHP

[Disable](#) ▼

[Uninstall](#)

This extension is enabled globally.

This extension is recommended based on the files you recently opened. [Ignore Recommendation](#)



PHP Intelephense bmewburn.vscode-intelephense-client

Ben Mewburn |  3,723,211 |  | [Repository](#) | [License](#)

PHP code intelligence for Visual Studio Code

[Disable](#) ▼

[Uninstall](#)

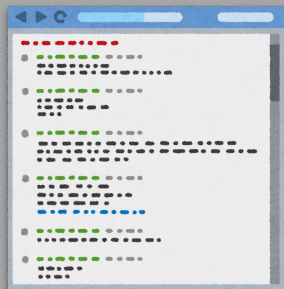
This extension is enabled globally.

This extension is recommended based on the files you recently opened. [Ignore Recommendation](#)

xamppの役割(PHP開発の環境)

htdocsディレクトリをwebサーバとして扱う！！

自分のPC



ブラウザ

PHPファイルにアクセス

PHP実行結果

- 処理した結果のデータ
- 構成したhtml



htdocsディレクトリ
擬似的なサーバ(localhost)
ここにPHPファイルを配置

準備(ソースコードを指定の場所に配置)

サンプルフォルダを下記のディレクトリへ移動しましょう！

- windows : C:\xampp\htdocs\
- mac : /Applications/xampp/xamppfiles/htdocs/

【ポイント】PHPの開発手順

1. xamppを起動する
2. phpファイルが入ったフォルダをhtdocsに配置
3. vs codeでhtdocsの上記フォルダを開く
4. phpのコードを書く
5. ブラウザでlocalhostにアクセスして動作確認

以降, ③④⑤の繰り返し！

※一通り終わったらxamppを終了させましょう(次回起動でコケます)

PHPの基礎

PHPファイルの作成

- 拡張子が「.php」
- 例:「index.php」「insert.php」.....

開始タグと終了タグ

- 「<?php」で始まり「?>」で終了
- (phpしか書かない場合, 終了タグは省略可)
- 開始タグと終了タグの間に書かれた処理がサーバ上で実行される!
- タグ以外の部分はサーバでは実行されない(そのままブラウザに送信).

変数と数値, 文字列の扱い

```
// 変数の扱い
// 変数は「$****」←$で始める！
$number = 100;           // 処理の終わりには「;」必須！
$name = 'engineer';      // 文字列は「'」「"」で囲む
$1lang = 'php';          // 数値スタートはNG
$lang2 = 'javascript';   // OK
$_lang = 'Rust';         // OK

// 変数と文字列は「.」で連結する！
$str = 'ジーズで' . $lang2 . 'を勉強中！';
// こっちのほうが楽かな...！
$str = "ジーズで{$lang2}を勉強中！";
```


配列の扱い

```
// 配列
$array = ['JavaScript', 'PHP', 'Rust', 'COBOL'];
echo $array[2];           // Rust

// console.log()的なやつ（超重要）
var_dump($array);         // 変数や配列の中身や構造を表示

// echoでも出力できるが、var_dump()では構造も見られる！
```

条件分岐

```
// 条件分岐
```

```
$num = 1;
```

```
// rand(0, 1);でも試してみよう！
```

```
if ($num == 1) {  
    echo '値は1です';
```

```
// 「echo」でブラウザに表示
```

```
} else {  
    echo '値は1以外です';  
}
```

乱数を使ったおみくじ

```
// rand(min, max);を使用しておみくじを作ろう！

$num = rand(1, 5);

if ($num == 1) {
    $result = '大吉';           // 乱数の結果によって$resultに値を入れる
} elseif ($num == ...) {
    $result = ...
} elseif (...) {
    // ...
}
echo $result;
```

結果をHTMLに埋め込み

```
// PHPとhtmlは組み合わせることができる！！  
// <?=変数名?>でhtmlに変数を埋め込める  
// PHPで処理した結果からHTMLをつくるイメージ  
<?php  
    $result = '大吉';  
?>  
<!DOCTYPE html>      // ここからhtmlの記述  
    <head>  
        // ...  
    </head>  
    <body>  
        <h1>今日の運勢は<?=$result?>です！</h1>  
    </body>
```

PHPで実行された結果が埋め込まれる！

おみくじをつくろう

練習

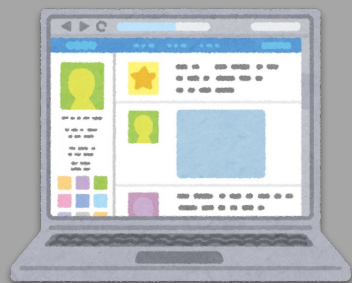
- `omikuji.php`でおみくじの処理を実装し、画面に結果を表示しよう！

サーバに情報を送る

サーバへデータ送信

サーバへデータを送る, とは??

- htmlファイルやphpファイルから別のphpファイルへデータを送る.
- データを受け取ったphpファイルがDBへの保存などの処理を実行.



ブラウザ

- テキストや数値などのデータ

http通信

http通信

- 処理した結果のデータ
- 構成したhtml



index.php

サーバ上で処理を実行
(計算や保存など)

サーバ

データの送信は2種類(GET & POST)

GET

フォームを使用してGETの処理を実装(送信ファイル)

```
// フォームに宛先と送信方法と名前を記述
<form action="todo_get_confirm.php" method="get">
  <div>
    todo: <input type="text" name="todo">
  </div>
  <div>
    deadline: <input type="date" name="deadline">
  </div>
</form>
```

todo登録画面 (GET)

todo:

deadline:

【ポイント3つ！】

- 「action="**"」で宛先のファイルが必要！
- 「method="**"」で送信方法を記述(getかpost).
- 「name="**"」で名前をつける(受取時の識別用).

受信側の処理

```
// 最初に必ずやること
<?php
    var_dump($_GET);
    exit();
?>
```

```
// 解説
// - GETで送信された情報は$_GETに入って送られる.
// - まず「情報が受け取れているかどうか」をチェックすることが大事！！
//   （情報が受け取れないと以降どうしようもない）
// - $_GETは配列になっており, name属性を指定することで取り出せる.
// `exit()`は以降の処理を中止する.
```

受信側の処理(続き)

```
// データの取り出し
$todo = $_GET['todo'];
$deadline = $_GET['deadline'];
// あとはPHPの変数として処理可能！

// GETってなんだ？？
// - サーバから情報を取得する。 URLに情報を追加して送信できる。
// - データの特定（自分の名前で検索）など、少量のデータ送信に向く。
//   （URLにデータが含まれている）
```

```
todo_get_confirm.php?todo=PHPの課題&deadline=2020-06-02
```

todo表示画面 (GET)

todo	deadline
------	----------

PHPの課題	2020-06-02
--------	------------

POST

フォームを使用してPOSTの処理を実装(送信ファイル)

```
// フォームに宛先と送信方法と名前を記述
<form action="todo_post_confirm.php" method="post">
  <div>
    todo: <input type="text" name="todo">
  </div>
  <div>
    deadline: <input type="date" name="deadline">
  </div>
</form>
```

todo登録画面 (POST)

todo:

deadline:

【ポイントGETの場合と同じ！】

- 「action="**"」で宛先のファイルが必要！
- 「method="**"」で送信方法を記述(getかpost).
- 「name="**"」で名前をつける(受取時の識別用).

受信側の処理

```
// 最初に必ずやること
```

```
<?php
```

```
    var_dump($_POST);
```

```
    exit();
```

```
?>
```

```
// 解説
```

```
// - POSTで送信された情報は$_POSTに入って送られる.
```

```
// - まず「情報が受け取れているかどうか」をチェックすることが大事！！
```

```
// （情報が受け取れないと以降どうしようもない）
```

```
// - $_POSTは配列になっており, name属性を指定することで取り出せる.
```

```
// `exit()`は以降の処理を中止する.
```

受信側の処理(続き)

```
// データの取り出し
$todo = $_POST['todo'];
$deadline = $_POST['deadline'];
// あとはPHPの変数として処理可能！

// POSTってなんだ??
// - サーバにデータを送信するときに使用. 情報を見えないように送信する方法
// - 個人情報など. (送れるデータ量がGETと比較して多い)
// - ファイルを送信する場合にも使用
```



— todo表示画面 (POST) —

todo deadline

PHPの課題 2020-06-02

【参考】XSS : クロスサイトスクリプティング

セキュリティを高める

- formに悪意あるスクリプトを埋め込まれる場合がある.
- 表示する際に対策できる.
 -  `<?=$task?>`
 -  `<?=htmlspecialchars($task, ENT_QUOTES); ?>`

※現時点では優先度低いので特に意識しなくてOK !

※最近ブラウザが優秀なので記述しなくてもわりと防いでくれる.

【練習】サーバクライアント間のデータ通信

練習

- `todo_get.php`と`todo_post.php`のform欄の設定を記述しよう！
- `todo_get_confirm.php`と`todo_post_confirm.php`でデータを受け取り, 表示！

PHPでのファイル操作

ファイルヘデータを書き込む

送信したデータをファイルに書き込んで保存する

やること

- 入力したデータをtxtファイルに書き込む
- 書き込んだデータを読み込んで表示する
- 保存場所は「data」ディレクトリの「todo.txt」

必要なファイル

- データを入力して送信するファイル(todo_txt_input.php)
- データを受け取ってファイルに書き込むファイル(todo_txt_create.php)
- ファイルのデータを読み込んで表示するファイル(todo_txt_read.php)

送信側ファイルの処理

データ送信の流れ

- 送信先のファイルを指定する(今回はtodo_txt_create.php)
- 送信方式を指定する(GET or POST)←今回はPOSTで実装
- formにname属性を指定する

送信側ファイルの処理

```
<form action="todo_txt_create.php" method="POST">
  ...
  <div>
    todo: <input type="text" name="todo">
  </div>
  <div>
    deadline: <input type="date" name="deadline">
  </div>
  <div>
    <button>submit</button>
  </div>
  ...
</form>
```

【ポイント！】

- 「action="**"」で宛先のファイルが必要！
- 「method="**"」で送信方法を記述(post).
- 「name="**"」で名前をつける(受取時の識別用).

受信側ファイルの処理

書き込みファイルの流れ

- データを受け取る
- 書き込み先のファイルを開く(なければ新たにファイルを作成)
- 他の人が書き込まないようにロックする
- データを書き込む
- ロックを解除する
- ファイルを閉じる
- 入力画面に移動

受信側ファイルの処理

// ファイル書き込み操作の流れ

```
$todo = $_POST["todo"];  
$deadline = $_POST["deadline"];  
$write_data = "{$deadline} {$todo}\n";  
$file = fopen('data/todo.txt', 'a');  
flock($file, LOCK_EX);  
fwrite($file, $write_data);  
flock($file, LOCK_UN);  
fclose($file);  
header("Location:todo_txt_input.php");
```

// データ受け取り

```
// スペース区切りで最後に改行  
// ファイルを開く 引数はa  
// ファイルをロック  
// データに書き込み,  
// ロック解除  
// ファイルを閉じる  
// 入力画面に移動
```

【参考】ファイル操作時のオプション

主な引数の種類

- r 読み込みのみで開く
- r+ 読み込み/書き込み用に開く
- w 書き込みで開く&内容を削除→ファイルがなければ作成
- w+ 読み込み/書き込みで開く&内容を削除→ファイルがなければ作成
- a 追加書き込みのみで開く→ファイルがなければ作成
- a+ 読み込み/追加書き込みで開く→ファイルがなければ作成

参考: <https://www.php.net/manual/ja/function.fopen.php>

ファイル書き込み処理の実装

練習1

- `todo_txt_input.php`のform欄の設定を記述しよう！
- `todo_txt_create.php`でデータを受け取り, ファイルに書き込もう！
- 書き込み処理の結果を`todo.txt`で確認しよう！
 - エディタからtxtファイルを開いて書き込まれていればOK.

⚠ 注意点 ⚠

PHPからファイルにアクセスするときにハマることがあります！！

- macの人
 - 「htdocsに入れた講義フォルダ」で
メニュー表示⇒情報を見る⇒共有とアクセス権
 - 全て「読み/書き」に変更⇒歯車ボタン⇒内包している項目に適用
- winの人
 - 特になし

ファイルからデータ参照

txtファイルの内容を読み込んでブラウザに表示する

読み込みファイルの流れ

- 出力用の変数を用意する
- txtファイルを開く(読み取り専用)
- ファイルをロックする
- txtファイルのデータを読み込んで出力用の変数に入れる
- ロックを解除する
- ファイルを閉じる
- (html上で表示)

読み込みファイルの処理

// ファイル読み込み操作の流れ

```
$str = ''; // 出力用の空の文字列
$file = fopen('data/todo.txt', 'r'); // ファイルを開く（読み取り専用）
flock($file, LOCK_EX); // ファイルをロック
if ($file) {
    while ($line = fgets($file)) { // fgets()で1行ずつ取得→$lineに格納
        $str .= "<tr><td>{$line}</td></tr>"; // 取得したデータを$strに入れる
    }
}
flock($file, LOCK_UN); // ロック解除
fclose($file); // ファイル閉じる
// （$strに全部の情報が入る！）
```

ファイル読み込み処理の実装

練習2

- `todo_txt_read.php`で`todo.txt`の内容を読み込もう！
- 読み込んだデータをブラウザで表示しよう！

課題

【課題】csvファイルでアンケート集計

構成

- 入力画面(index.php)
- 書き込みファイル(create.php)
- 読み込みファイル(read.php)

最低限ここまで！

- 名前, email, 任意の質問を入力
- 入力内容をcsv形式で「data/data.csv」に保存
- 読み込みファイルでcsvファイルの内容を表示

※例によってアンケート項目とか適当でOK！

【課題】csvファイルでアンケート集計

アップグレード

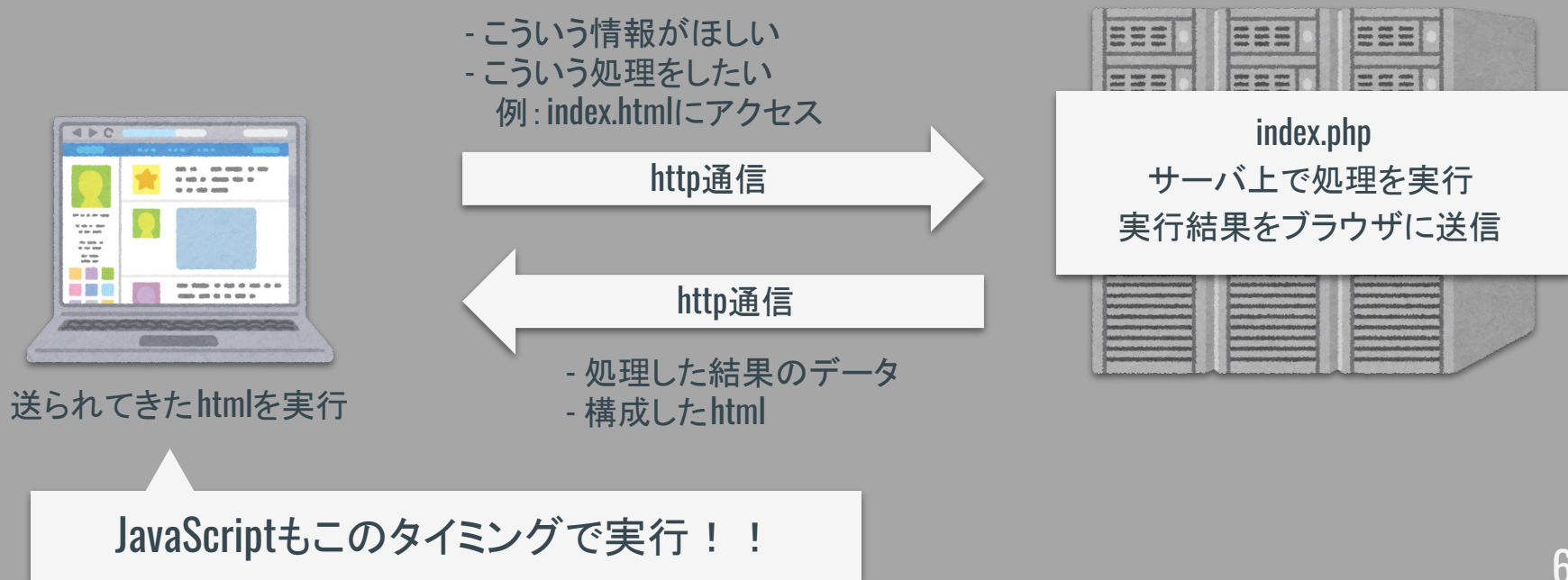
- 同じ画面で送信と表示を実行
- csvファイルの内容の集計結果をグラフ表示, デザインをカッコよく
- 結果に応じた評価機能とか統計解析とか
- 卒制のアイデアでつくってみる！！

考え方

- PHPはJavaScriptと比較して自由度が低い．．！
- 狙ったデータを確実に表示できるように！！

PHPとJavaScriptは処理の順番が大事！！

サーバでPHPが実行 → ブラウザでJavaScriptが実行



締切は次回授業前木曜「23:59:59」

P2Pタイム

まずはチーム内で解決を目指す！

訊かれた人は苦し紛れでも応える！！