

PHP / HTML 講習会

合宿事前講座 PHP編 Part.1 (基礎編) u306065 櫛田一樹

1: PHPとは?

HTML+プログラミング

という感じのものです (少なくとも現在はこの程度の認識で構いません)

1: PHPとは?

あれ?

Webサイトでプログラミングって
以前やりませんでしたっけ??

1-1: JSとPHPの違い

そう！

以前やったWebサービスプログラミング言語

それは**JavaScript**でした。

では、JSとPHP、一体何が違うのでしょうか。

1-1: JSとPHPの違い

一番の違い、それは……

処理をサーバー側で行うかPC側で行うかです。

サーバー側で処理を行っているのがPHP
PC側(以降 クライアント側と表記)で処理を行っているのがJSです。

1-1: JSとPHPの違い

んで、それぞれどんな違いがあるかと言うと、

- ・**サーバーサイド処理**(バックエンドとも言う)：
 - データそのものを処理する。
 - データベースにアクセスできる。(SNS等が作れる)
 - PCが持っていないデータを参照するときに使う
- ・**クライアントサイド処理**(フロントエンドとも言う)：
 - リアルタイムな動作ができる。(投稿フォームに何も入力されてなければエラー等)
 - データベースへのアクセスはできない。

1-1: JSとPHPの違い

主に以下のような言語が用いられます。

- サーバーサイド：

- PHP
- Ruby
- Kotlin などなど……

- クライアントサイド：

- JavaScript
- CSS
- HTML などなど……

1: PHPとは?

で、先程学んだ通り、PHPはサーバーサイドの言語なので、

サーバーが必要です。

1: PHPとは?

なので、何かしらの手段を用いてサーバーを用意しましょう。

- ・レンタルサーバーを借りる
 - ・いらないPCをサーバーにする
 - ・自分のPC内でサーバーを動かす
- などなど……

1: PHPとは?

今回はXAMPPというソフトウェア環境を使って
自分のPCで簡単にサーバーを動かします。

一緒に進めたいという人で
XAMPPをまだインストールしていない人は
調べてインストールしておいてください。
わからなければSlackで聞いてね！

2: PHP開発の前に

先程から言っているとおり
今回からXAMPPというものを使っていきます。

そのため、XAMPPフォルダ内の
htdocs フォルダの場所(インストールした場所)を
確認しておいてください。

Windowsだと **C:\xampp\htdocs** かもです
自分の環境(Ubuntu 18.04)では **/opt/lampp/htdocs** でした
Macは知らぬいです(冷酷)

2-1: PHP開発をする上で便利なサイト

ぼく「**関数の使い方がわからないよお…(‘> ω <。)ふええ…**」
そんなときは、以下のサイトがおすすめだっつ！！

- **PHP公式リファレンス** <https://www.php.net/manual/ja/>
- **Qiita** <https://qiita.com/>

2-1-1: PHP公式リファレンスの見方

読み方がわからにくいので一緒に読んでいきましょう

date

(PHP 4, PHP 5, PHP 7)

date – ローカルの日付/時刻を書式化する

説明

```
date ( string $format [, int $timestamp = time() ] ) : string
```

指定された引数 `timestamp` を、与えられた フォーマット文字列によりフォーマットし、日付文字列を返します。タイムスタンプが与えられない場合は、現在の時刻が使われます。つまり `timestamp` はオプションでありそのデフォルト値は `time()` の値です。

- `:string` (最後)
→ 戻り値が文字列型である
- `string $format` (カッコ内)
→ 引数に文字列が必要
→ 与えるべき文字列は後述
- []内 (カッコ内)
→ 省略可能
→ 省略した場合、デフォルト値となる。

2-1-1: PHP公式リファレンスの見方

パラメータ

format

出力される日付文字列の書式。以下のオプションを参照ください。[定義済みの定数](#)も用意されており、たとえば `DATE_RSS` はフォーマット文字列 '`D, d M Y H:i:s`' と同じ意味になります。

以下の文字が `format` パラメータ文字列として認識されます

format 文字	説明	戻り値の例
<code>日</code>	---	---
<code>d</code>	日。二桁の数字（先頭にゼロがつく場合も）	01 から 31
<code>D</code>	曜日。3文字のテキスト形式。	<code>Mon</code> から <code>Sun</code>
<code>j</code>	日。先頭にゼロをつけない。	1 から 31
<code>l (小文字の'L')</code>	曜日。フルスペル形式。	<code>Sunday</code> から <code>Saturday</code>
<code>N</code>	ISO-8601 形式の、曜日の数値表現 (PHP 5.1.0 で追加)。	1 (月曜日) から 7 (日曜日)
<code>s</code>	英語形式の序数を表すサフィックス。2 文字。	<code>st, nd, rd</code> または <code>th</code> 。 <code>j</code> と一緒に使用することができる。
<code>w</code>	曜日。数値。	0 (日曜)から 6 (土曜)

- ・パラメータ
→引数のこと

先程の `$format` と書かれているところに `format` の内容が対応する。

例 : `date('d');`

関数によってはこのように一覧で出てくる場合もある。

2-2: ファイル構成について

今回はhtdocsフォルダ内で作業を行います。
VSCode等のエディタでhtdocsフォルダを開きましょう！

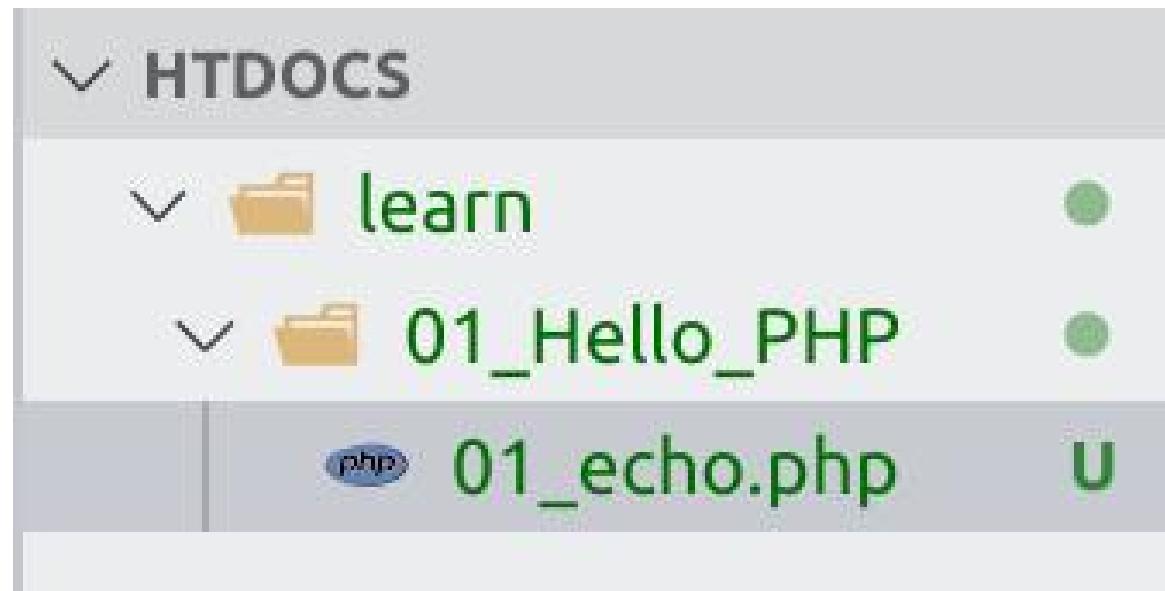
ファイル → フォルダを開く → htdocsを選択

htdocsの場所は前のスライドを参考にしてね！

2-2: ファイル構成について

今回はたくさんファイルをつくります。
そのため、私の場合はファイル構成を以下のようにしました。
きれいなファイル構成を心がけましょう！

htdocs/learn/num_内容/num_名前.php



3: はじめよう！PHP！

それでは準備OKですね？早速開発を始めていきましょう。

今回のおしながき

- | | |
|--------------------|----------|
| 3-1: はじめまして！ | (変数, 出力) |
| 3-2: データを送受信してみよう！ | (POST送信) |
| 3-3: バリデーション！ | (条件分岐) |
| 3-4: 何度も繰り返す | (ループ) |
| 3-5: 複雑なデータを管理しよう | (配列) |

3-1：はじめまして！

はじめよう！PHP！

3-1: はじめてまして！

？？？「PHP開発は初めてか？力抜けよ」

一番はじめは簡単なものからはじめていきましょう。

3-1: はじめて！

では、新たな言語を始める儀式としておなじみのHelloWorldを
やっていきましょう。

以下のコードを入力してみてください。

```
php 01_echo.php ×  
learn > 01_Hello_PHP > php 01_echo.php  
1 <?php  
2 echo 'Hello_PHP!';
```

3-1: はじめて！

入力したコードを見ていきましょう！

echo構文はC言語でいうprintf文です。

```
1 <?php  
2 echo 'Hello_PHP!';
```

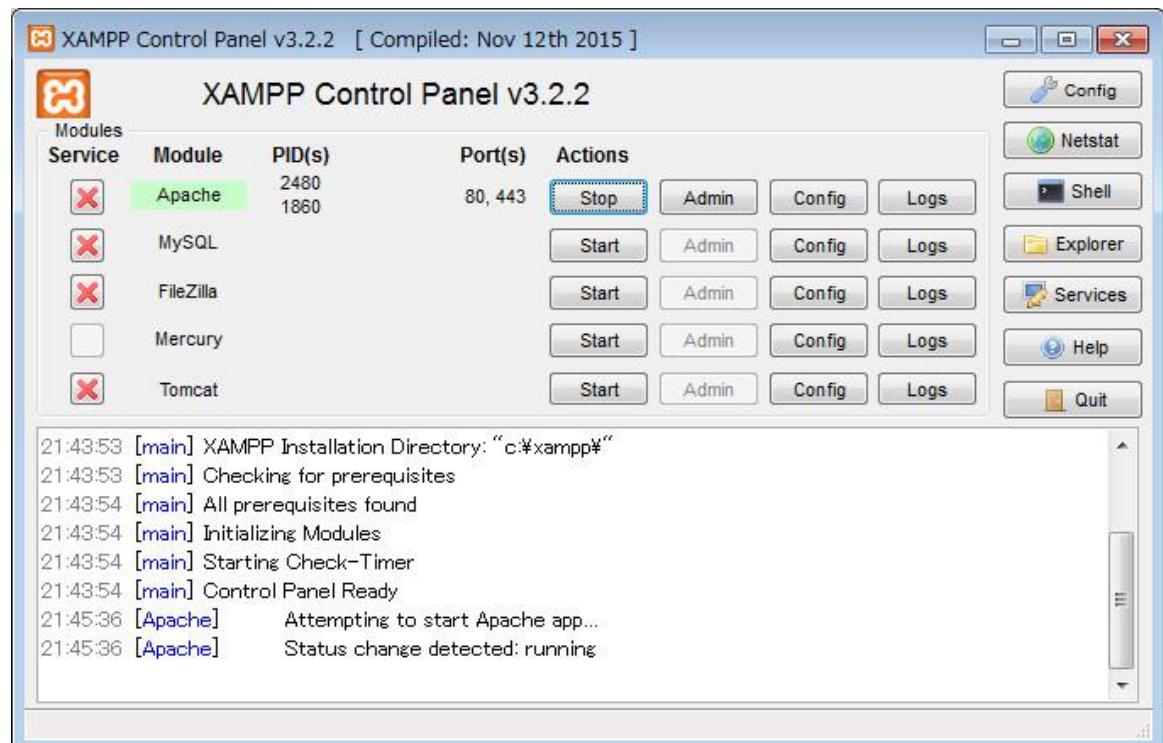
文字列の前後を'で囲む

「<? php」は、これからPHPが始まるという合図。
HTML内にPHPを埋め込むことも可能なのだが、その場合はPHPの
終わりを表す「?>」が必要。ファイル内がPHPだけの場合は「?>」を
書かないことが推奨されている。(クソ長文)

3-1: はじめて！

では、XAMPPを立ち上げて、Apacheを立ち上げましょう！

ApacheのところのStartボタンをクリック！



←こんな感じの表示になつたらOK!
(バージョンによって表示が変わる場合があります)

3-1: はじめて！

ブラウザで以下のURLにアクセスしよう

http://localhost/learn/num_内容/num_名前.php

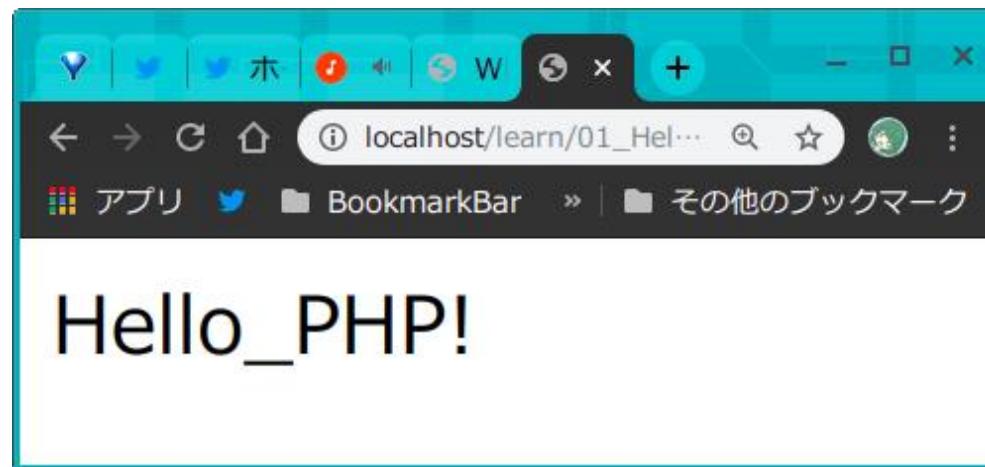
(今回は http://localhost/learn/01_Hello_PHP/01_echo.php)

以降は「ブラウザで実行してみよう！」と書いてあつたら
赤字のURLの”num_内容”にフォルダ名(例: 01_Hello_PHP)
”num_名前.php”にファイル名(例: 01_echo.php)を入れたURLにアクセスしよう！

3-1: はじめて！

はいプロ。

こんな感じにうまく表示されれば成功！おめでとう！



3-1: はじめて！

さて、プロの皆さんなら容易いと思いますが
変数というもう聞き飽きたであろう概念をやっていきましょう。
(変数を知らない人はSlackで聞いてください！！)

先日のJS講習会の資料でも書いてあったとおり、
PHPも**型推論**です なんでも入ります
型推論がわからない人は調べてみよう！(言語化して説明ができない)

3-1: はじめて！

では、早速変数を用意してみましょう。
phpでは、変数はすべて \$ で宣言します。

以下のコードを入力してみましょう！

02_variable.php ×

learn > 01_Hello_PHP > 02_variable.php > ...

```
1  <?php  
2  
3  // 変数の宣言と代入を行う  
4  $name = 'カズ之助';  
5  
6  // 変数の中身を確認する( ' で囲わない ! )  
7  echo $name;
```

ここで 「echo '\$name';」 と、
\$nameを ' で囲ってしまうと変数の中身ではなく
「\$name」と出力されてしまうので注意！

3-1-1: 変数を扱おう！

ブラウザで実行してこんな感じに
変数の内容が表示されれば成功です！



3-1-1: 変数を扱おう！

皆さん特に問題はないとは思いますが
一度代入のルールを確認しましょう。

先程のコードに一文追加してブラウザで実行しましょう。

```
4 $name = 'カズ之助';  
5 // 変数の内容を上書きする  
6 $name = 'おっぱい';  
7 // 変数の中身を確認する( ! )
```



3-1-2: 結合演算子

続いて、文字列の連結です。

今までC言語とかには無かったと思います。

PHPでは、文字列の連結を .(ドット)で行います。

この . のことを 「**結合演算子**」 といいます。

```
1 // プログラム名 - オン フォント ,  
2  
3 // 変数の中身を確認する( ' で囲わない ! )  
4  
5 echo $name .'さん、こんにちは';
```



これで、名前の部分だけを閲覧したユーザー名に置き換えることが可能になるよ！！Webページのログイン機能とかでよくあるよね！

3-1-2: 結合演算子

結合演算子には仲間がいます。

それは、めっちゃ長い文字列を変数にぶち込むときに便利な

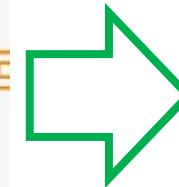
「結合代入演算子」 です。

これは .= で表します。

```
<?php
```

```
$tmp = 'こんにちは。私はカズ之助といいます。今回
```

例えば、こんなに長い文字列の代入も……



```
<?php
```

```
$tmp = 'こんにちは。';  
$tmp .= '私はカズ之助といいます。';  
$tmp .= '今回は、PHPを勉強しましょう';
```

こんな感じに分けることができる！

3-1-3: 型推論を体感しよう！

さて、変数には色々なものを入れることができます。

型推論はデータに合わせて自動で変数を最適な型にします。
次のプログラムでは、型推論を体験してみましょう。

3-1-3: 型推論を体感しよう！

```
03_type.php ×  
learn > 01_Hello_PHP > 03_type.php > ...  
1  <?php  
2  
3 // 文字列  
4 $data1 = 'こんにちは';  
5  
6 // 整数( ' をつけると文字列として認識される ! )  
7 $data2 = 3;  
8  
9 // 論理値  
10 $data3 = TRUE;  
11  
12 // 実数  
13 $data4 = 3.2;  
14  
15 // null  
16 $data5 = null;  
17  
18  
19 // 出力  
20 var_dump($data1);  
21 var_dump($data2);  
22 var_dump($data3);  
23 var_dump($data4);  
24 var_dump($data5);
```

左のプログラムを入力してみてください！

var_dumpという関数を使って
変数の中身を表示しましょう！

echoと違うのは変数の型や
バイト数まで出してくれます！

入力が終わったらブラウザで実行！！

3-1-3: 型推論を体感しよう！

下のように表示されると思います。

変数の型(変数のサイズ もしくは内容) の順に表示されています

型推論で適切な型に変換しているのがわかったと思います！



3-1-4: 変数の計算

さて、何気に長い道のりでした。

次が3-1の最後です。最後は変数どうしの計算です。
簡単ですのでパパっとやっちまいましょう。

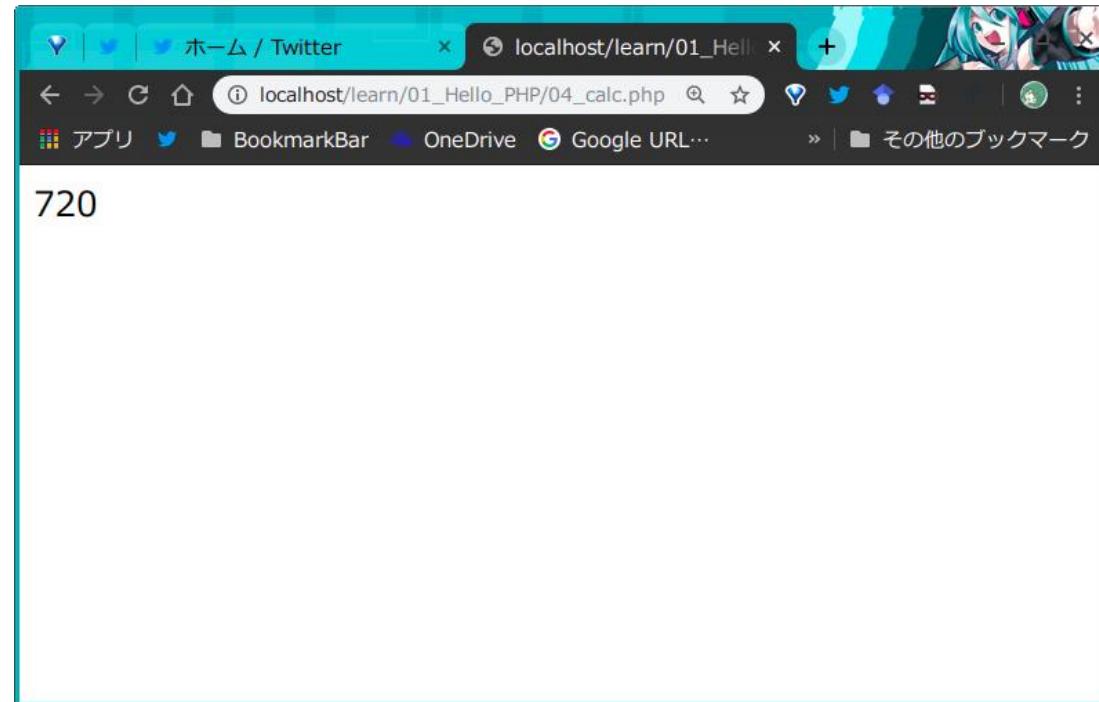
3-1-4: 変数の計算

次のコードを入力して実行してみましょう！

```
php 04_calc.php ×  
learn > 01_Hello_PHP > php 04_calc.php > ...  
1  <?php  
2  $x = 3;  
3  $y = 4;  
4  $z = 5;  
5  
6  // 変数を使用した計算  
7  $sum = $x + $y;  
8  echo $sum;  
9  
10 // echo文内の計算  
11 echo $y * $z;
```

3-1-4: 変数の計算

あらら。720が出てきてしましました。
これは改行が無かったため、 $3 + 4$ の 7 と、 $4 * 5$ の 20 が
連続して出てきてしまったためです。



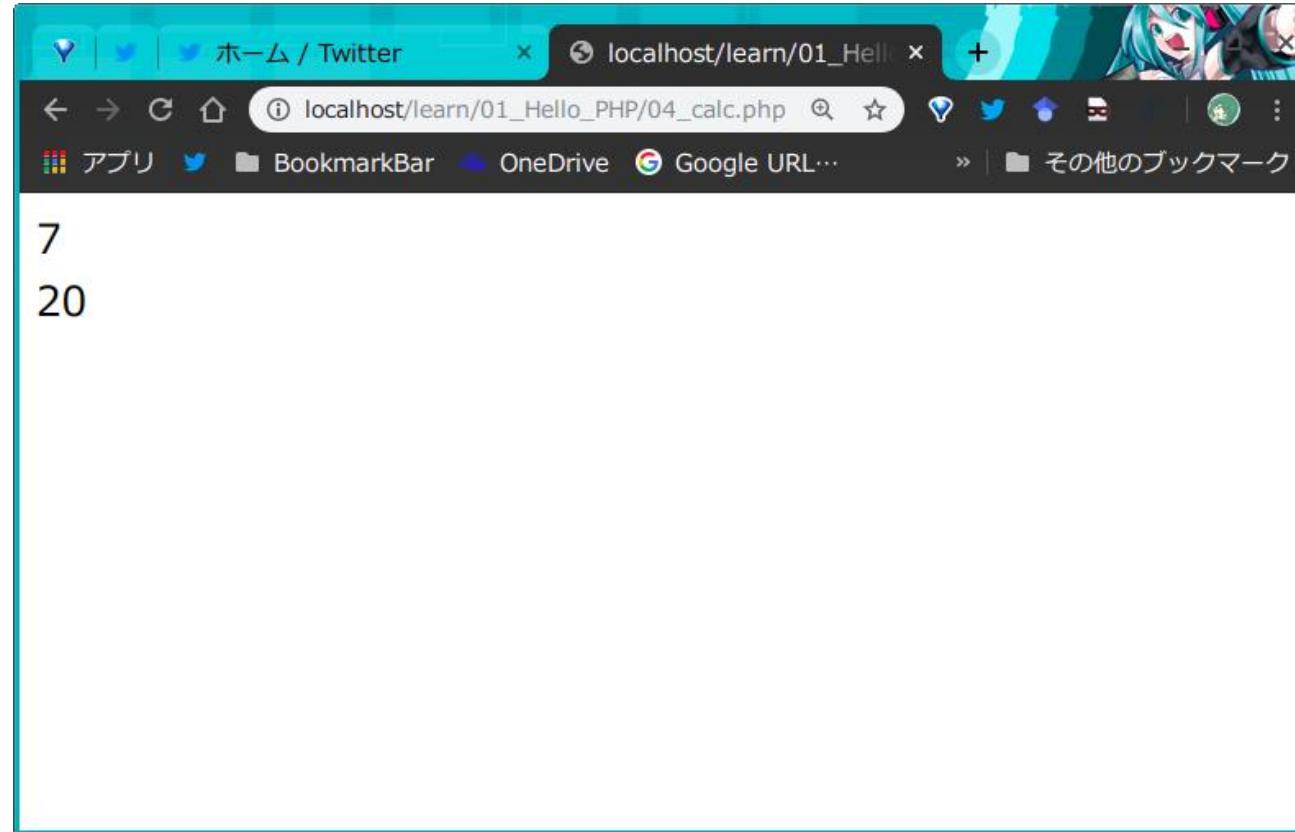
3-1-4: 変数の計算

結合演算子で改行タグを結合してやりましょう。
このように出力文では、HTMLのタグも出力することができます。

```
5
6 // 変数を使用した計算
7 $sum = $x + $y;
8 echo $sum . '<BR>';
9
10 // echo文での計算
```

3-1-4: 変数の計算

今度こそいい感じに表示できましたね。
おめでとうございます！超有能です！



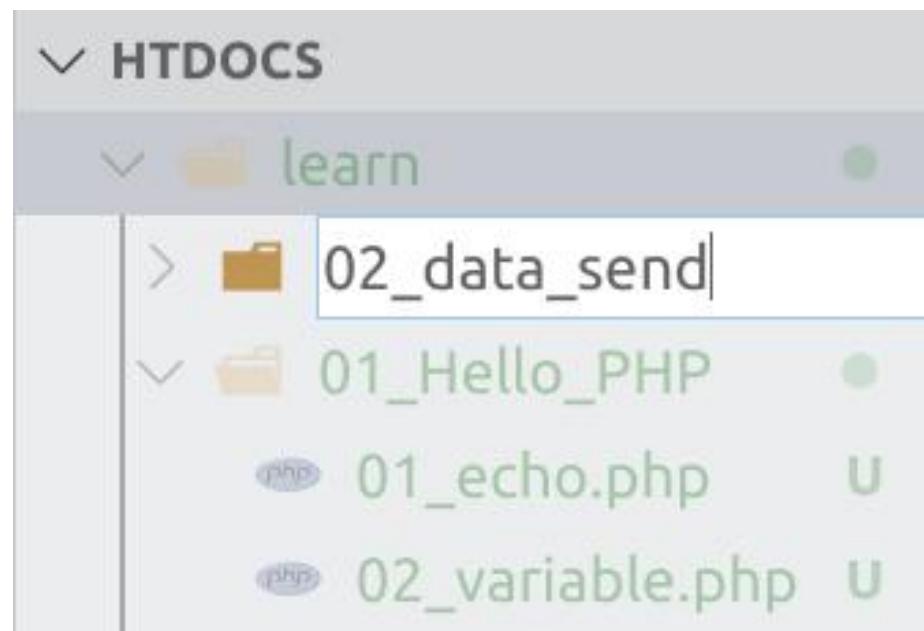
3-2: データを送受信してみよう！

はじめよう！PHP！

3-2: データを送受信してみよう！

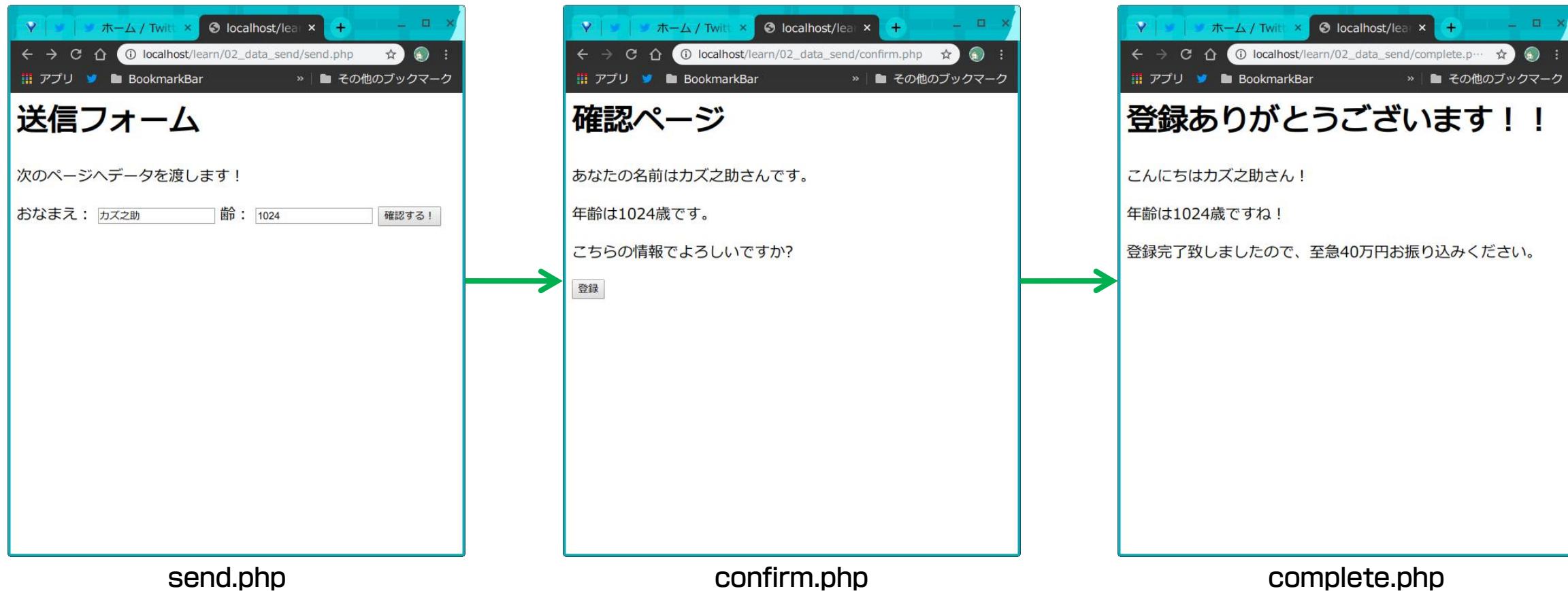
さて、今までの知識プラスαで、データの送受信ができます。
やってみましょう！！

…と、その前に、作業フォルダを変えましょう。



3-2: データを送受信してみよう！

送信フォームからデータを送信して、
確認画面と完了画面でデータを表示するプログラムです。



3-2-1：要件定義

システムが何をしなければならないかを決定したものを

「要件定義」 といいます。

要件定義をしっかりと行うことでの
いい感じにシステムが構築できるように開発をすすめられます。

開発のやり方とかは長くなっちゃうので別な機会に…

3-2-1：要件定義

今回は以下のように要件を定義しました。

- ・名前と年齢を入力するフォームを用意する
- ・確認ページにデータを渡し、表示
- ・確認ページから登録ボタンだけで完了ページにデータを渡す

3-2-2: 送信ページを作っていこう！

さあ、張り切って開発を始めていきましょう！
まずは、send.phpに以下のHTMLコードを書いていきましょう！

```
php send.php ×

learn > 02_data_send > php send.php

1 <html>
2   <head>
3     <meta charset="utf-8">
4   </head>
5   <body>
6     <h1>送信フォーム</h1>
7     <p>次のページへデータを渡します！</p>
8     <!-- この下からフォームを追加していく -->
9
10    </body>
11  </html>
```

3-2-2: 送信ページを作っていこう！

入力フォームを作っていきます！

<form>タグとその中身を以下のように作っていこう！

POSTという方法でデータを送信するよ！

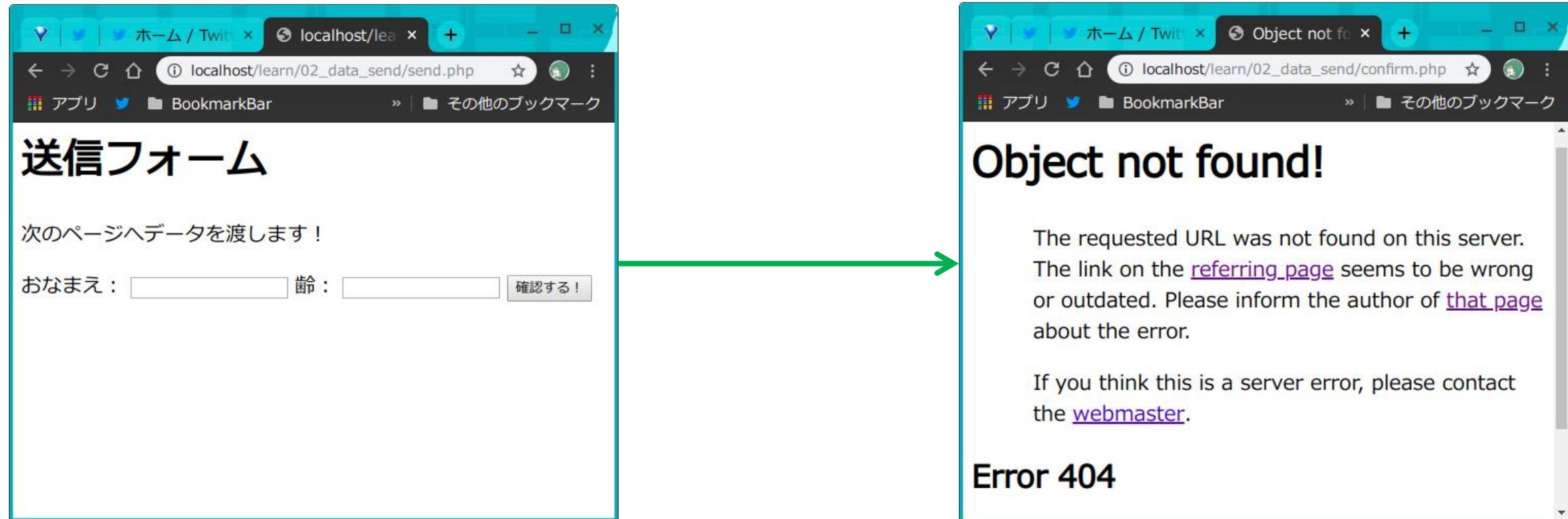
これでブラウザ上に表示されることなく送信できます

(他にはGETという方法があるけど、詳しくは次回あたりに……)

```
8   <!-- この下からフォームを追加していく -->
9   <form action="./confirm.php" method="POST">      ← 送信先のページと送信手段(POST)を決定
10  <label>おなまえ:</label>
11  <input type="text" name="user_name">            ← 受信先の名前をつける
12  <label>齢:</label>
13  <input type="text" name="old">                  ← 上に同じ
14
15  <input type="submit" value="確認する!">        ← 送信ボタンを用意
16  </body>
```

3-2-2: 送信ページを作っていこう！

一度この状態でブラウザで実行してみよう！！
見た感じ上手くできているけど……



「ファイルがないゾ」と怒られが発生してしまった

3-2-3: 確認ページを作っていこう！

とりあえずsend.php自体は上手く作れているっぽいので、
次はデータを受信してみよう！！

まずはテンプレとしてconfirm.phpに以下のHTMLコードを書こう！

```
confirm.php ×  
learn > 02_data_send > confirm.php  
1  <?php  
2  // POSTされたデータを受信する  
3  ?>  
4  
5  <html>  
6  |   <head>  
7  |   |   <meta charset="UTF-8">  
8  |   </head>  
9  |   <body>  
10 |   |   <h1>確認ページ</h1>  
11 |   |   <p>あなたの名前は さんです。</p>  
12 |   |   <p>年齢は 歳です。</p>  
13 |   |   <p>こちらの情報でよろしいですか?</p>  
14 |   |   <form action="complete.php" method="POST">  
15 |   |   |   <input type="submit" value="登録">  
16 |   |   </form>  
17 |   </body>  
18 </html>
```

3-2-3: 確認ページを作っていこう！

送信されたデータを取得するには
\$_POST という「スーパーグローバル変数」というものを使うよ！
これは送信ボタンを押した瞬間に作成されます。

inputのnameで指定した名前を使用して下のようになります！

```
1 <?php
2 // POSTされたデータを受信する
3 $user_name = $_POST['user_name'];
4 $old = $_POST['old'];
5 ?>
```

3-2-3: 確認ページを作っていこう！

HTML内で変数の内容を出力しよう！

受け取った名前と年齢をHTML内で出力するよ！

このように、HTMLの一部だけプログラムを書き込めるのがPHPの特徴！

```
12 <!!> 姓氏'ハーン'!!>
13 <p>あなたの名前は<?php echo $user_name; ?>さんです。</p>
14 <p>年齢は<?php echo $old; ?>歳です。</p>
15 <n>こちらの情報を上書きしますか?</n>
```

3-2-3: 確認ページを作っていこう！

このページでは入力欄がないけど、どうやってデータを渡すの？

と思った画面の前のキミ！安心してほしい。

hiddenという方法を使えば隠してデータを送信できる！

下の画像の17, 18行目を入力しよう！

渡すデータはvalueで設定できるからecho文を使ってセットしよう！

```
15 <p> ヒント: おまけで php のタグ</p>
16 <form action="complete.php" method="POST">
17   <input type="hidden" name="user_name" value="<?php echo $user_name; ?>">
18   <input type="hidden" name="old" value="<?php echo $old; ?>">
19   <input type="submit" value="登録">
20 </form>
```

3-2-3: 確認ページを作っていくう !

ここまでできたら一度実行してみよう !!
必ず**send.php**から実行してね !

下の画像のように動いたら成功だよ !



3-2-4: 完了ページを作っていくう！

最後のページは受け取ったデータを表示するだけ！簡単でしょ？
(一般的にはこの後、データベースへの追加処理等が行われる)

コメントをヒントに自分で作ってみよう！！

```
PHP complete.php ×  
learn > 02_data_send > PHP complete.php  
1  <?php  
2  // POSTされたデータを取得する  
3  ?>  
4  
5  <html>  
6  <head>  
7  <meta charset="utf-8">  
8  </head>  
9  <body>  
10 <h1>登録ありがとうございます！！</h1>  
11 <!-- HTML内にechoプログラムを埋め込もう！ -->  
12 <p>こんにちは さん！</p>  
13 <p>年齢は 歳ですね！</p>  
14 <p>登録完了致しましたので、至急40万円お振り込みください。</p>  
15 </body>  
16 </html>
```

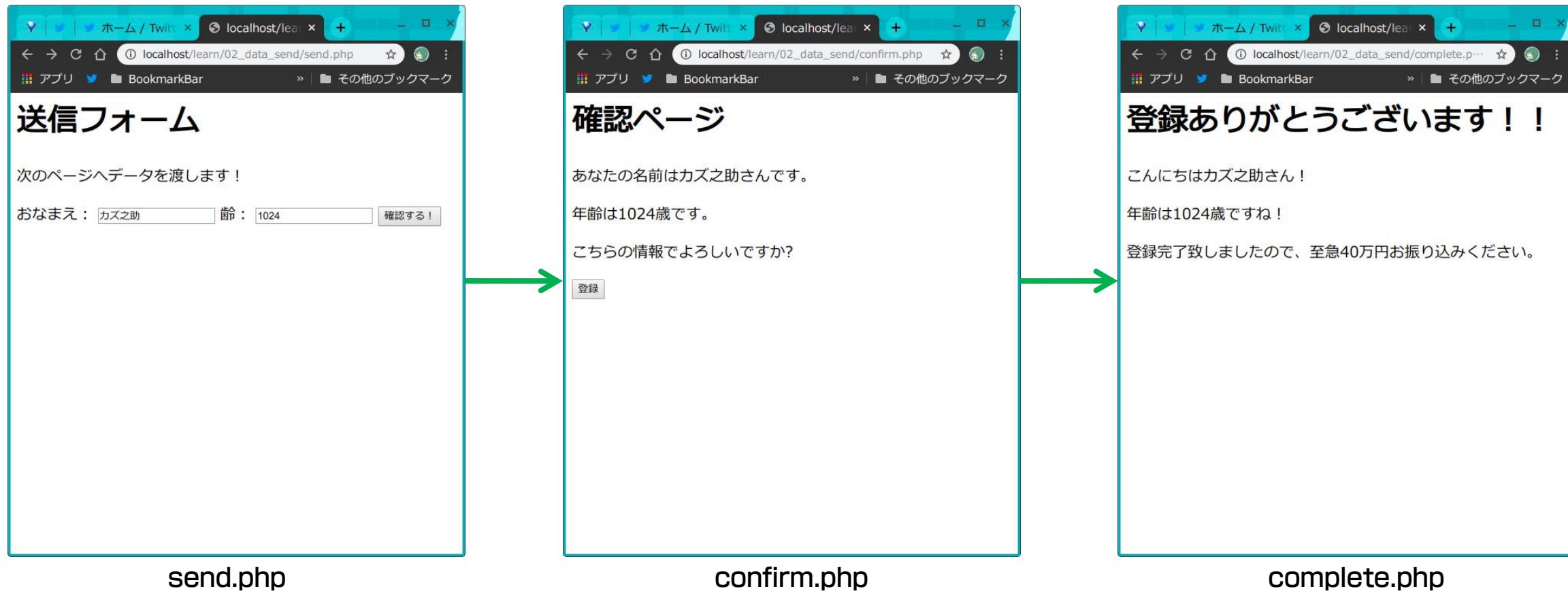
3-2-4: 完了ページを作っていこう！

答え合わせ

```
php complete.php ×  
learn > 02_data_send > complete.php > ...  
1  <?php  
2  // POSTされたデータを取得する  
3  $name = $_POST['user_name'];  
4  $old = $_POST['old'];  
5  ?>  
6  
7  <html>  
8  |  <head>  
9  |  |  <meta charset="utf-8">  
10 |  </head>  
11 |  <body>  
12 |  |  <h1>登録ありがとうございます！！</h1>  
13 |  |  <!-- HTML内にechoプログラムを埋め込もう！ -->  
14 |  |  <p>こんにちは<?php echo $name; ?>さん！</p>  
15 |  |  <p>年齢は<?php echo $old; ?>歳ですね！</p>  
16 |  |  <p>登録完了致しましたので、至急40万円お振り込みください。</p>  
17 |  </body>  
18 </html>
```

3-2-4: 完了ページを作っていくう !

実行して以下のようになれば成功だ！お疲れ様！！
必ずsend.phpからページ内のボタンで移動しよう！



3-3: バリデーション！

はじめよう ! PHP !

3-3: バリデーション！

そういえば、皆さん気づいたかもしれません、先程のPOST送信のやつ重大な欠陥があるのです。気づいてなかった？？

えっと、試しに先程のsend.phpで何も入力しないで送信してみてください。

そのまま次のページに遷移できてしまうはずです。

こ れ は い け な い

3-3: バリデーション！

入力値が適切なものかどうかを検査することをバリデーションといいます。

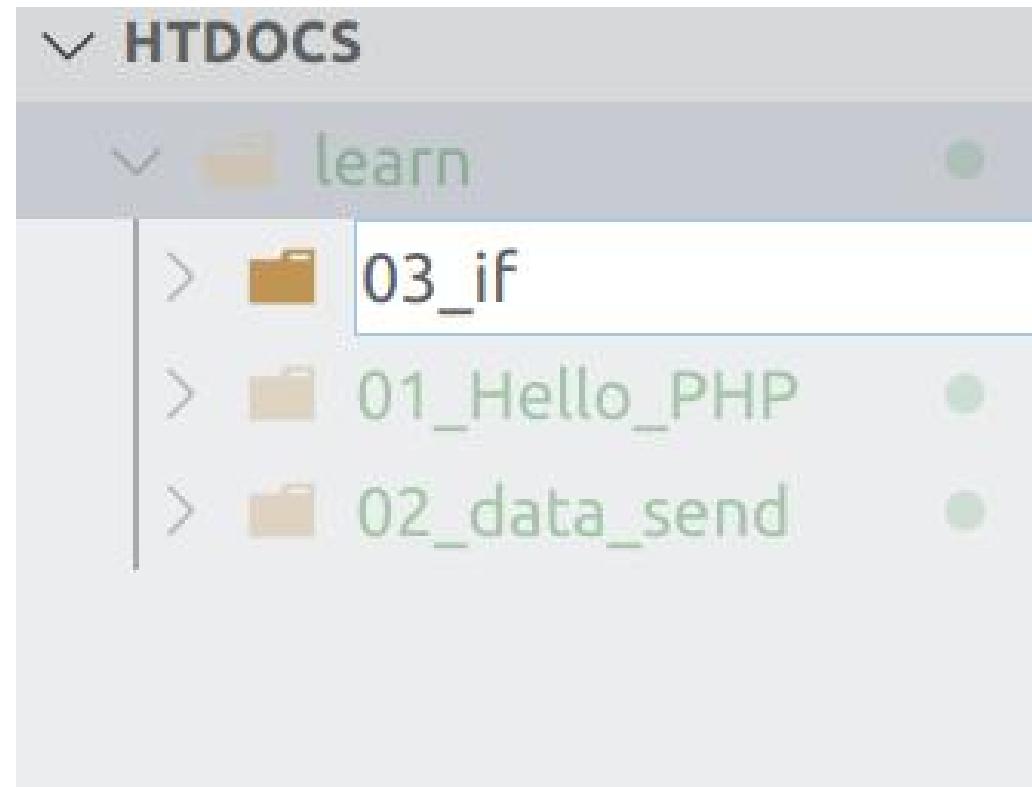
今回は条件分岐を使って簡単にバリデーション機能のあるページを作ろう！

その前に、if文の使い方について説明します。

基本的にはC言語と同じなので安心してください。

3-3-1: if文

では、簡単にif文について一緒に勉強していきましょう。
その前に、learnフォルダ内に 03_ifフォルダを作りましょう。



3-3-1: if文

```
php 01_if.php ×  
learn > 03_if > 01_if.php > ...  
1  <?php  
2  // まずは簡単なif文から  
3  
4  $lang = 1;  
5  
6  if( $lang === 1){  
7      echo 'こんにちは';  
8  }elseif( $Lang === 2){  
9      echo 'Hello';  
10 }elseif( $Lang === 3){  
11     echo 'Bonjour';  
12 }else{  
13     echo 'err';  
14 }
```

左のコードを見てみましょう。
C言語と似ている……けど違う？

なんか=が2つじゃなくて3つだし……

elseとifの間が空いてないし……

3-3-1: if文

ひとつずつ見ていきましょう！
難しくないです！大丈夫！

3-3-1: if文

```
php 01_if.php ×  
learn > 03_if > 01_if.php > ...  
1 <?php  
2 // まずは簡単なif文から  
3  
4 $lang = 1;  
5  
6 if( $lang === 1){  
7     echo 'こんにちは';  
8 }elseif( $Lang === 2){  
9     echo 'Hello';  
10 }elseif( $Lang === 3){  
11     echo 'Bonjour';  
12 }else{  
13     echo 'err';  
14 }
```

① === になっている！何故だ！

実はC言語と同じように == でも書けます。
でも、厳密な型まで判定しないので、

**文字列 'TRUE'と論理値TRUEを同じもの
として判定したり……**

いろいろとゴミなんです。

だから==としようね！！

割とマジでセキュリティホール※に陥りかねないので……

※セキュリティホール【名】：セキュリティ上の欠陥。脆弱性。

例: 7payはサービスに重大なセキュリティホールが多数存在したため、2ヶ月でサービス終了を余儀なくされた。

3-3-1: if文

```
php 01_if.php ×  
learn > 03_if > 01_if.php > ...  
1 <?php  
2 // まずは簡単なif文から  
3  
4 $lang = 1;  
5  
6 if( $lang === 1){  
7     echo 'こんにちは';  
8 }elseif( $Lang === 2){  
9     echo 'Hello';  
10 }elseif( $Lang === 3){  
11     echo 'Bonjour';  
12 }else{  
13     echo 'err';  
14 }
```

②elseとifの間が空いていない！！

今までどおりelse ifも別に大丈夫です。
以上！閉廷！！皆んな解散！！

3-3-2: 比較演算子

等しい、等しくないなどの比較演算子一覧です。

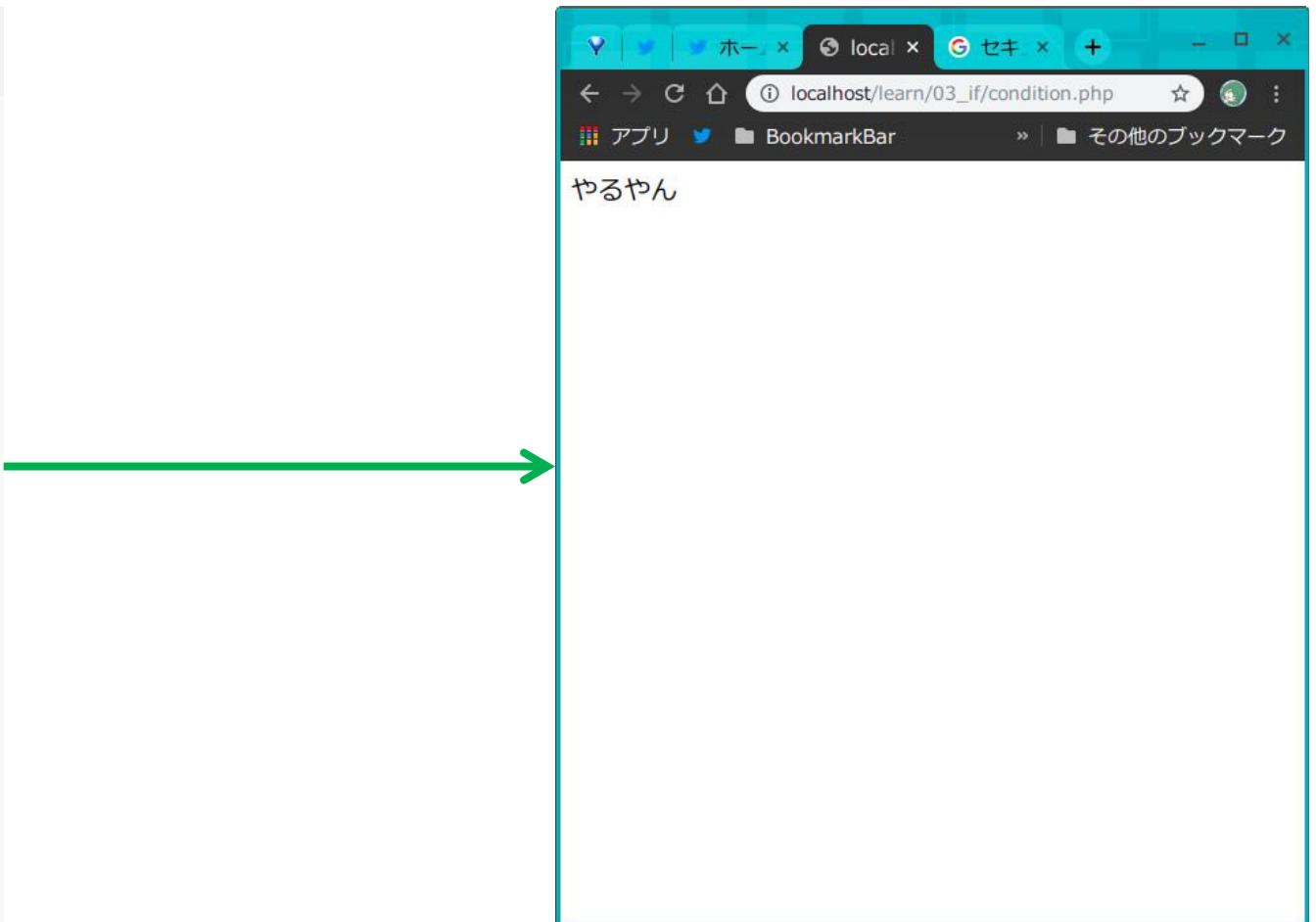
上2つだけ注意してもらえば後は今までどおりです。

演算子	意味	使い方
<code>==</code>	値が同じかつ型が同じ	<code>46 == 64 //FALSE</code>
<code>!=</code>	値が違うもしくは型が違う	<code>'46' != 46 //TRUE</code>
<code>></code>	左辺が右辺より大きい(右辺超)	<code>\$x > 56 // \$xが57以上ならTRUE</code>
<code><</code>	左辺が右辺より小さい(右辺未満)	<code>\$x < 45 // \$xが44以下ならTRUE</code>
<code>>=</code>	左辺が右辺以上	<code>\$x >= 20 // \$xが20以上ならTRUE</code>
<code><=</code>	左辺が右辺以下	<code>\$x <= 50 // \$xが50以下ならTRUE</code>

3-3-3: 論理演算子

PHPでもC言語などと同じように()内に条件式をいくつも書くことができます。

```
php 02_condition.php ×  
learn > 03_if > 02_condition.php > ...  
1 <?php  
2  
3 $score = 82;  
4 if($score >= 0 && $score < 60){  
5     echo '再履おめでとうございます！';  
6 }elseif($score >= 60 && $score < 80){  
7     echo 'うーん…合格！w';  
8 }elseif($score >= 80 && $score < 100){  
9     echo 'やるやん';  
10 }elseif($score === 100){  
11     echo '俺の負け！w';  
12 }else{  
13     echo '不正な点数故に再履！w';  
14 }
```



3-3-3: 論理演算子

ここで主に使う論理演算子は3つあるよ！

演算子	意味	使い方
&&	左辺かつ右辺(両方TRUE)	<code>3 === 3 && 5 === 5 //TRUE</code>
	左辺もしくは右辺(片方or両方TRUE)	<code>3 === 4 5 === 5 //TRUE</code>
!	否定(~でなかつたら)	<code>!\$x === 20 // \$xが20で無い場合TRUE</code>

3-3-4: 宇宙船演算子

右のコードを見てください

宇宙船演算子は2つの値を比較し、

- ・ 同値なら0
- ・ 左辺が大きければ1
- ・ 右辺が大きければ-1を返す

演算子です。

03_space_if.php ×

learn > 03_if > 03_space_if.php

```
1  <?php
2  // 宇宙船演算子
3  echo 1 <=> 1; // 0
4  echo 2 <=> 1; // 1
5  echo 1 <=> 2; // -1
```

自分は使ったこと無いです(小声)

3-3-5: if文の入れ子構造(ネスト)

C言語と同じようにネスト構造にすることができます。
ネストを作る際はタブキーでインデントをしっかりしましょう。

(詳しく説明しません。わからない人はSlackか当日聞きに来てね！)

3-3-6: switch文

```
php 04_switch.php ×  
learn > 03_if > php 04_switch.php > ...  
1  <?php  
2  
3  $language = 2;  
4  
5  switch($language){  
6      case 1:  
7          echo 'こんちは';  
8          break;  
9      case 2:  
10         echo 'Hello';  
11         break;  
12      case 3:{  
13          // 実は{}をつけたほうが自動インデント入って見やすい  
14          // もちろん{}無くても大丈夫です。  
15          echo 'Bonjour';  
16          break;  
17      }  
18      default:{  
19          echo 'err';  
20      }  
21 }
```

PHPでもswitch文を使うことができます。
マジで何でもできるなお前。

使い方はC言語と同じです。
左にコードを載せておきます。

これも詳しくやらないです。
別にできなくても苦労しないので。

(自分はcase文には{}を付ける派です)

3-3: バリデーション！

ようやくバリデーションに戻ってきました。
結構丁寧に復習したつもりだったけどどうだったかな?
だるかったかな?

さらっと流した人もいるかもしれないね。
さて、バリデーション！張り切っていってみよう！

3-3: バリデーション！

——要件定義——

- ・ 空(文字数0)で送信していないかのチェックを行う
- ・ 20文字をオーバーしていないかチェックする
- ・ エラーがある場合、その内容を報告する

3-3: バリデーション！

```
05_validate.php ×  
learn > 03_if > 05_validate.php  
1  <?php  
2 // ポスト内容を確認し、正しいかどうか判断する  
3 ?>  
4  
5 <html>  
6   <head>  
7     <meta charset="utf-8">  
8     <style type="text/css">  
9       .center{  
10         text-align: center;  
11     }  
12     input{  
13       margin: 5px;  
14     }  
15   </style>  
16 </head>  
17 <body>  
18   <div class="center">  
19     <h1>フォームを検証しよう</h1>  
20     <p>  
21       <?php  
22         // 入力内容にエラーがあればエラーメッセージ  
23         // なければ「あなたの好きな映画は～～です」と出力  
24       ?>  
25     </p>  
26  
27     <!-- actionが空の場合、同じファイルに送信 -->  
28     <form action="" method="POST">  
29       <label>好きな映画</label>  
30       <input type="text" name="movie"><br>  
31       <input type="submit">  
32     </form>  
33   </div>  
34 </body>  
35 </html>
```

まずはフォームを作っていくよ！
左のコードを書いていってね！！



3-3: バリデーション！

```
05_validate.php ×  
learn > 03_if > 05_validate.php > ...  
1  <?php  
2  // ポスト内容を確認し、正しいかどうか判断する  
3  $movie = $_POST['movie'];  
4  
5  if(mb_strlen($movie) === 0){  
6      // 文字数が0だった場合  
7      $err = '文字を入力してください';  
8  }elseif(mb_strlen($movie) > 20){  
9      // 20文字を超えていた場合  
10     $err = '20文字以内で入力してください';  
11 }  
12 ?>
```

mb_strlen()は文字列の長さを取得する関数！

mbはマルチバイトのこと！

「a」が1バイトで、「あ」が3バイトなので、そのまま計算すると文字数にずれが生じる！

それを補正してくれるよ！

mbからはじまる関数はいっぱいあるよ！

3-3: バリデーション！

isset()は変数が既にセットされているかを調べる関数！

変数は値を入力した時点で存在することになるため、\$errが存在するか否かで判断しているよ！

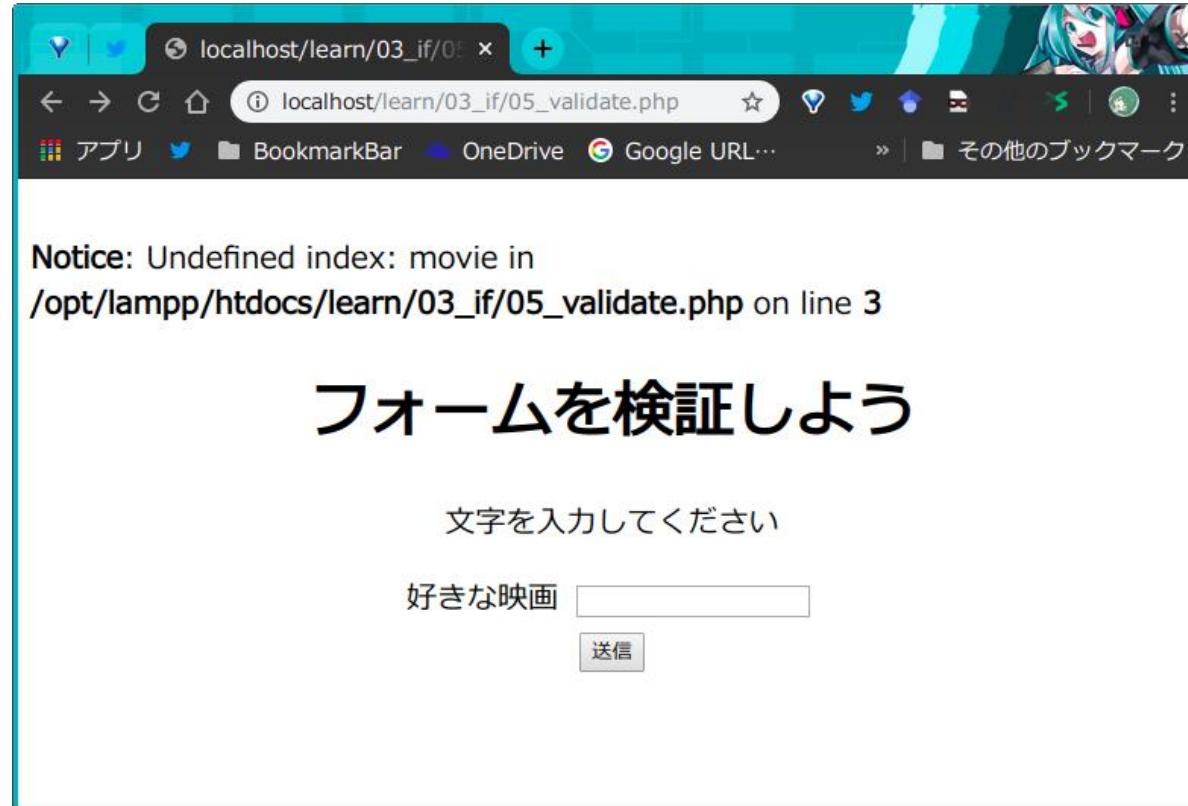
これを書いたら一度保存してブラウザで実行してみよう！

```
29          <p>
30          <?php
31          // 入力内容にエラーがあればエラーメッセージ
32          // なければ「あなたの好きな映画は～～です」と出力
33          if(isset($err)){
34              echo $err;
35          }else{
36              echo 'あなたの好きな映画は' . $movie . 'です。';
37          }
38      ?>
39      </p>
```

3-3: バリデーション！

およよ？何かエラーが出てるね。見てみよう。

変数movieは定義されてないよ！と書いてあるね



3-3: バリデーション！

on line 3と書いてあるね！

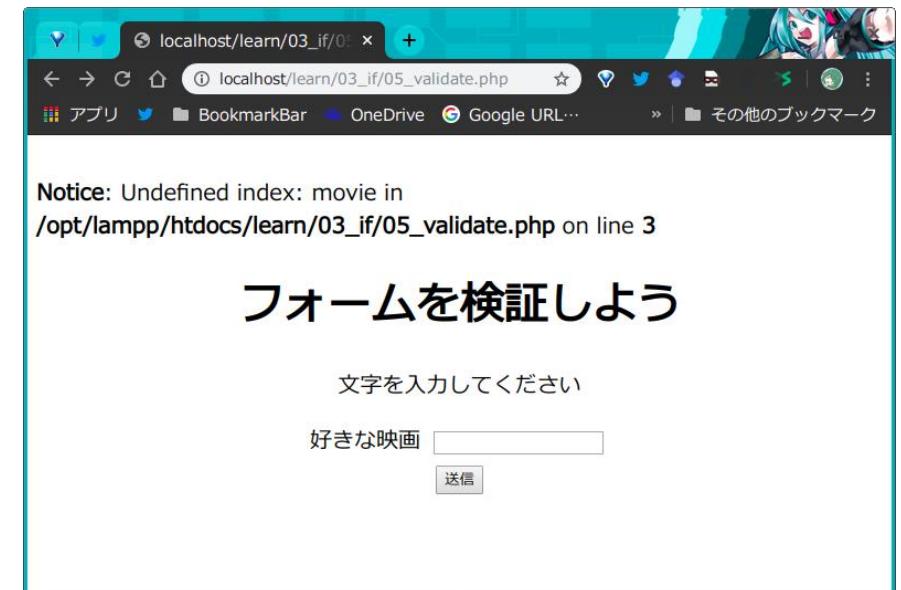
3行目だから、POST取得のあたりかな？

コードを読んでいくと、最初にPOSTの取得から入るよね。

しかし、**URLから訪れただけでは、POSTの取得はしていない。**

よって、\$_POST['movie']はまだ存在しておらず、
undefinedになってしまふんだ！

修正しよう！



3-3: バリデーション！

```
05_validate.php ×  
learn > 03_if > 05_validate.php > ...  
1  <?php  
2  // undefinedが出ないように対策する  
3  $movie = '';  
4  
5  if($_SERVER['REQUEST_METHOD'] === 'POST') {  
6      // POST送信されたら以下の処理を実行する  
7  
8      // ポスト内容を確認し、正しいかどうか判断する  
9      $movie = $_POST['movie'];  
10  
11     if(mb_strlen($movie) === 0){  
12         // 文字数が0だった場合  
13         $err = '文字を入力してください';  
14     }elseif(mb_strlen($movie) > 20){  
15         // 20文字を超えていた場合  
16         $err = '20文字以内で入力してください';  
17     }  
18 }  
19 ?>  
20  
21 <html>
```

左のように修正しよう！

\$movie = ""; と、「」内を空にしておくよ！

この書き方を「**空文字**」といいます！

\$_SERVERはスーパーグローバル変数です。
自動で生成され、色々な情報を得られます。

キーに**REQUEST_METHOD**を指定すると
POSTされたかどうかを判定できます！

3-3: バリデーション！

完成品！お疲れ様！！ここまでできれば立派なif文マスターだ！

The image shows two screenshots of a web browser window side-by-side, illustrating a form submission process.

Left Screenshot: The title bar says "localhost/learn/03_if/0". The page content is:

フォームを検証しよう

文字を入力してください

好きな映画

送信

Right Screenshot: The title bar says "localhost/learn/03_if/0". The page content is:

あなたの好きな映画はちんちんくんのぼうけんです。

好きな映画

送信

A large green arrow points from the left screenshot to the right one, indicating the progression of the form submission.

3-4: 何度でも繰り返す

はじめよう ! PHP !

3-4: 何度も繰り返す

当然PHPにもループ処理を行う関数が存在します。
このセクションでは以下の2つの関数についてお勉強していきましょう！

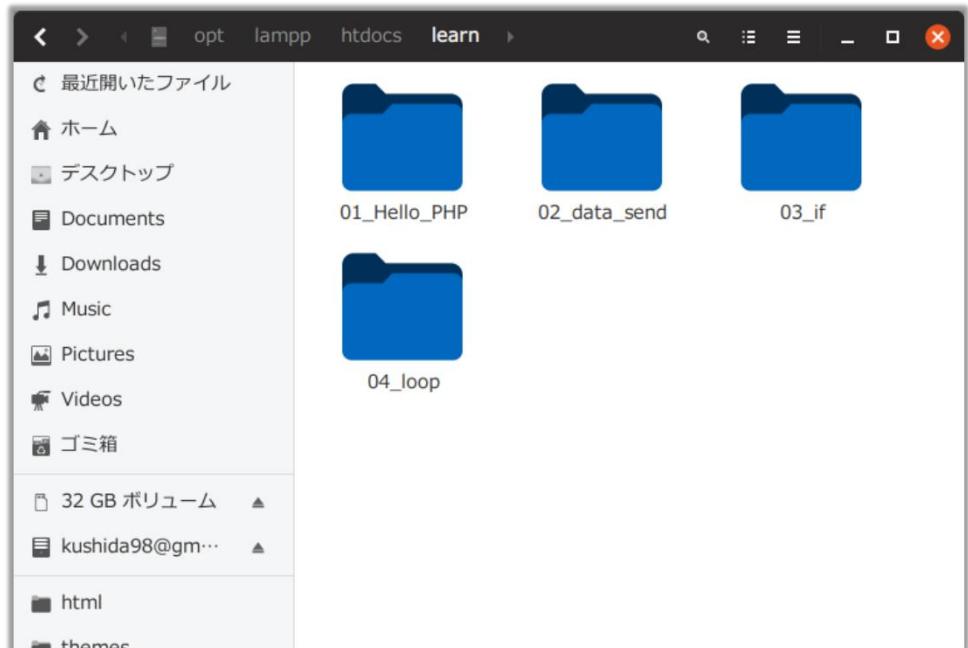
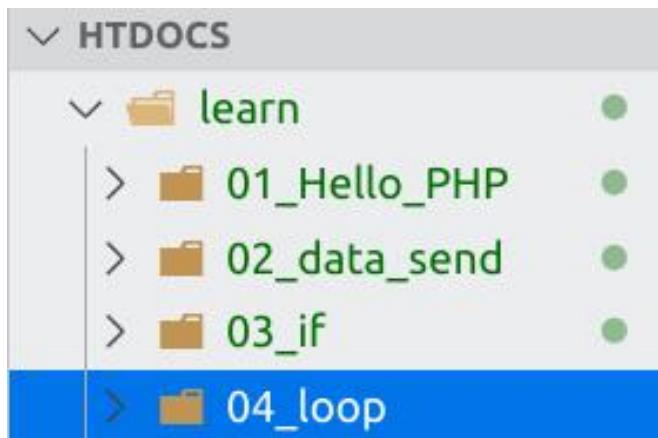
といっても文法とかC言語と同じなのでわかる人は
飛ばしてもらって構いません。

- **for文**
- **while文**

3-4: 何度も繰り返す

はい、では毎度おなじみ
セクションごとに作業フォルダを変えましょう。
このクセをつけておくと見やすいフォルダ構成になるよ！

今回は「04_loop」としました。



ファイルマネージャ(Windowsのエクスプローラ)で
htdocs内のlearnフォルダの中を見た感じ

3-4-1: while文

まずwhile文の抑えておきたい基本構文を一緒に勉強していきましょう！

```
while(継続条件) {  
    継続条件がTRUEだった際の処理;  
}
```

このコードで継続条件がfalseになるまで何度も処理が行われます。

3-4-1: while文

ではwhile文を使って20行出力するプログラムを作ってみましょう。
次のページに答えの一例が載っています。

仕様

- 各行の内容は「n行目です。」とすること
→nの部分に行数が入ります。



A screenshot of a web browser window titled 'localhost/learn/04_loop/while'. The page displays a continuous loop of text: '1行目です。' followed by '2行目です。' and so on up to '20行目です。'. The browser interface includes a toolbar with icons for back, forward, search, and refresh, and a tab bar showing 'Google' and 'localhost'.

```
1行目です。
2行目です。
3行目です。
4行目です。
5行目です。
6行目です。
7行目です。
8行目です。
9行目です。
10行目です。
11行目です。
12行目です。
13行目です。
14行目です。
15行目です。
16行目です。
17行目です。
18行目です。
19行目です。
20行目です。
```

3-4-1: while文

答え合わせ

```
php while.php ×  
learn > 04_loop > php while.php > ...  
1  <?php  
2  
3  $i = 1;                      // 変数iの初期化  
4  while( $i <= 20){           // ()内に条件式を設定(この条件式がTRUEの場合{}内の処理が実行される)  
5      echo $i.'行目です。<BR>';  
6      $i++;                     // 変数iを1増やす  
7  }  
8
```

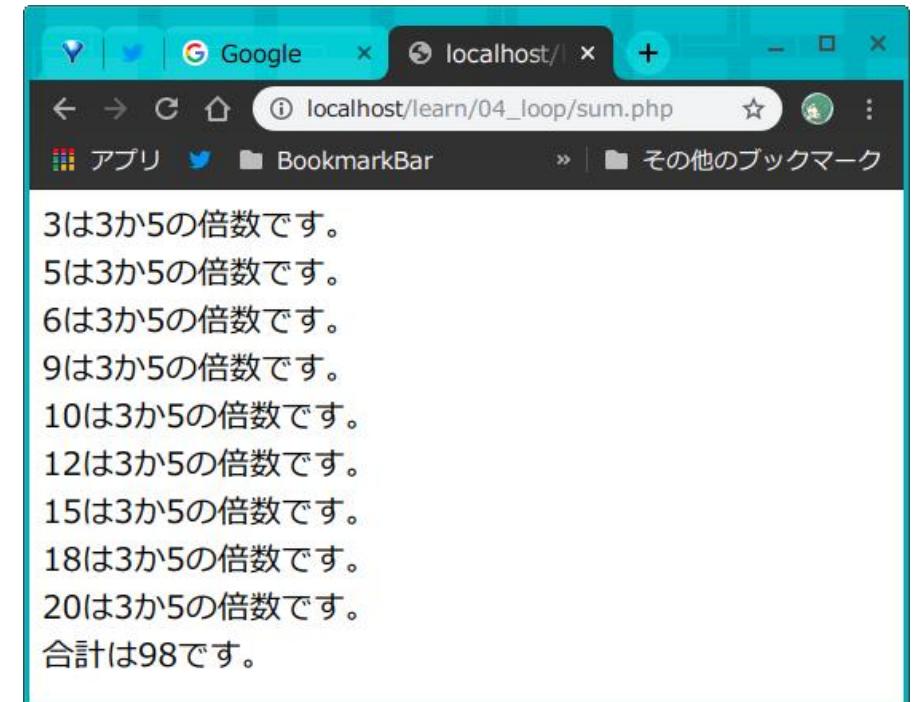
3-4-2: 複合演算子

あなたは何故か突然 1から20の数字のうち3と5の倍数の合計を求めたくなつてしましました。

気でも狂ったのでしょうか? さて、求めていきましょう。

以下のコードを修正して合計を求めましょう。

```
sum.php •  
learn > 04_loop > sum.php > ...  
1  <?php  
2  
3  $i = 1;  
4  $sum = 0;  
5  while(条件式){  
6      if(条件式){  
7          echo $i.'は3か5の倍数です。<br>';  
8          // 前の値に上書きして和を求める  
9          处理;  
10     }  
11     $i++;  
12 }  
13  
14 echo '合計は' . $sum . 'です。';
```



3-4-2: 複合演算子

答え合わせ

```
php sum.php ×  
learn > 04_loop > sum.php > ...  
1  <?php  
2  
3  $i = 1;  
4  $sum = 0;  
5  while($i <= 20){  
6      if($i % 3 == 0 || $i % 5 == 0){  
7          echo $i.'は3か5の倍数です。<br>';  
8          // 前の値に上書きして和を求める  
9          $sum += $i;  
10     }  
11     $i++;  
12 }  
13  
14 echo '合計は' . $sum . 'です。';
```

さて、ここで `+=` とかいう新しい書き方が出てきました。(9行目)

これを**複合演算子**といいます。

やっている処理は `$sum = $sum + $i` と全く同じなのでこれでも正解です。

3-4-2: 複合演算子

さて、複合演算子は `+=` だけではありません。

前にやった `.=` だって複合演算子の一種です。

複合演算子

演算子	説明	同一の処理
<code>+=</code>	値を加えて代入する。	<code> '\$i += 1'</code> は <code> '\$i = \$i + 1'</code> と同じ
<code>-=</code>	値を引いて代入する。	<code> '\$i -= 1'</code> は <code> '\$i = \$i - 1'</code> と同じ
<code>*=</code>	値を掛け代入する。	<code> '\$i *= 2'</code> は <code> '\$i = \$i * 2'</code> と同じ
<code>/=</code>	値を割って代入する。	<code> '\$i /= 2'</code> は <code> '\$i = \$i / 2'</code> と同じ
<code>%=</code>	値を割った余りを代入する。	<code> '\$i %= 2'</code> は <code> '\$i = \$i % 2'</code> と同じ
<code>.=</code>	文字列を連結して代入する。	<code> '\$a .= \$b'</code> は <code> '\$a = \$a . \$b'</code> と同じ

3-4-3: for文

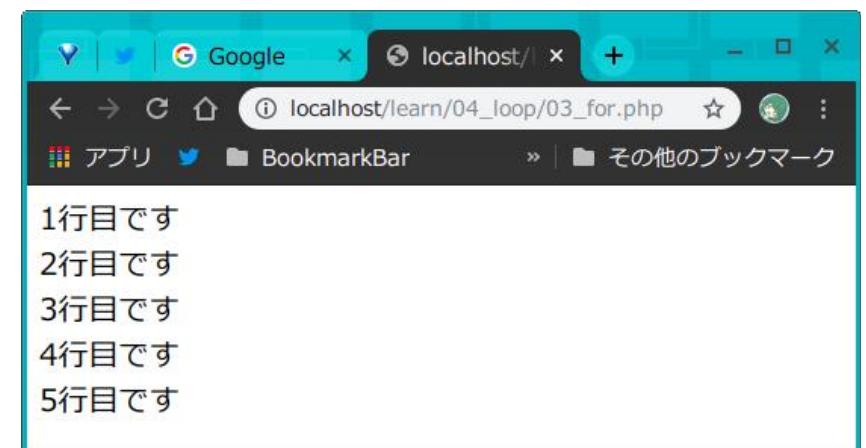
さて、続いてはfor文です。一緒に見ていきましょう。

```
for(初期値 ; 条件式 ; 増減式) {  
    条件式がTRUEだった際の処理;  
}
```

はい。そこまで難しくないです。実例も見ていきましょう。



```
03_for.php ×  
learn > 04_loop > 03_for.php > ...  
1  <?php  
2  
3  for( $i = 1; $i <= 5; $i++){  
4      echo $i.'行目です<br>;  
5  }
```



localhost/learn/04_loop/03_for.php

```
1行目です  
2行目です  
3行目です  
4行目です  
5行目です
```

3-4-4: 生年月日を選択するフォーム

実は今までの知識で、生年月日を選択するフォームを作れます！
せっかくなので一緒に作っていきましょう！

要件定義

- ・<select>タグのオプション部分をプログラムで作ろう！
- ・西暦は1950年から現在の年まで選択できるようにしよう！
- ・月日：月は12月まで、日は31日まで選択できるようにしよう！

3-4-4: 生年月日を選択するフォーム

まずはテンプレづくり！以下の内容のphpファイルを作ろう！

```
php birth_form.php ×
learn > 04_loop > birth_form.php
1 <html>
2   <bday>
3     <h1>生年月日を入力するフォーム</h1>
4     <label for="year">西暦</label>
5     <select name="year">
6       <option value="1980">1980</option>
7       <option value="1981">1981</option>
8     </select>年
9     <br>
10    <select name="month">
11      <option value="1">1</option>
12      <option value="2">2</option>
13    </select>月
14    <select name="day">
15      <option value="1">1</option>
16      <option value="2">2</option>
17    </select>日
18  </body>
19 </html>
```

なんとなくどこをループにすればいいかわかるかな？

3-4-4: 生年月日を選択するフォーム

じゃあ、まずは現在の年を取得しよう！ date関数で取得できるよ！
date()に指定できるパラメータはPHP公式リファレンスを参照してね！

```
4      <label for="year">西暦</label>
5      <select name="year">
6          <?php
7              // 現在の年を取得する
8              $now = date("Y");
9          ?>
10         </select>年
11         <br>
```

3-4-4: 生年月日を選択するフォーム

次に西暦のオプションをfor文で作っていくよ！

```
4   <label for="year">西暦</label>
5   <select name="year">
6     <?php
7       // 現在の年を取得する
8       $now = date("Y");
9       for( $i = 1950; $i <= $now; $i++){ ?>
10         <option value="<?php echo $i; ?>"><?php echo $i; ?></option>
11       <?php } ?>
12     </select>年
13     <br>
14     <select name="month">
```

1950年から現在まで繰り返す

年の選択欄を表示する

forの括弧を閉じる(忘れないで！)

3-4-4: 生年月日を選択するフォーム

同じようにfor文で月日のオプションを作ろう！

```
13     <br>
14     <select name="month">
15         <?php for( $i = 1; $i <= 12; $i++){ ?>
16             <option value="<?php echo $i; ?>"><?php echo $i; ?></option>
17         <?php } ?>
18     </select>月
19     <select name="day">
20         <?php for( $i = 1; $i <= 31; $i++){ ?>
21             <option value="<?php echo $i; ?>"><?php echo $i; ?></option>
22         <?php } ?>
23     </select>日
```

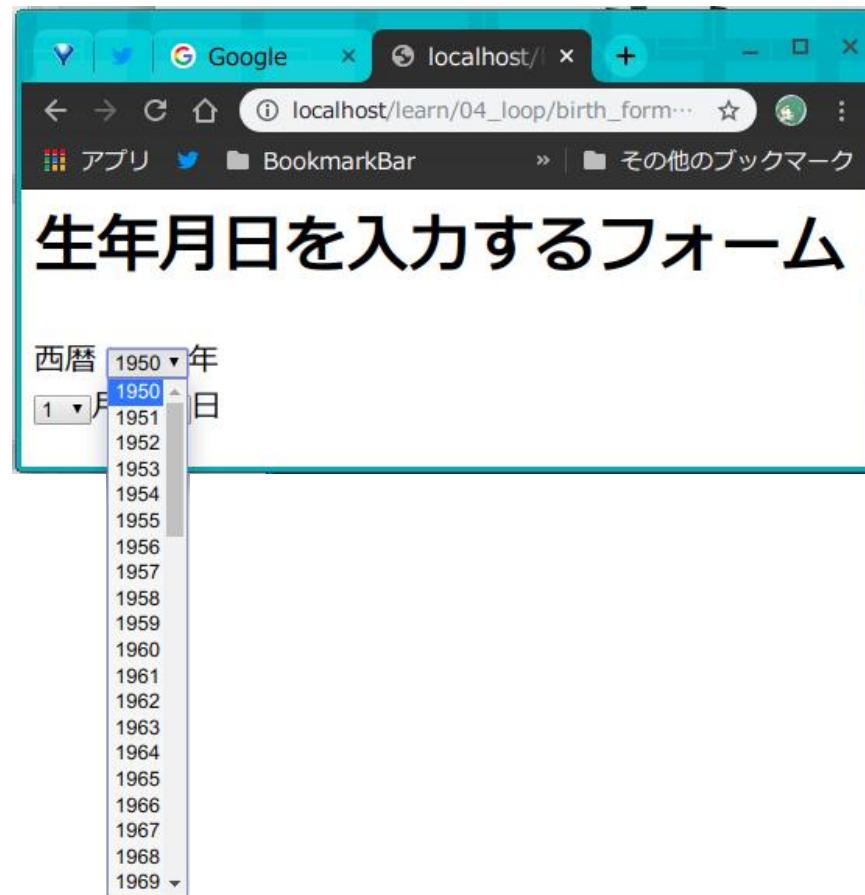
3-4-4: 生年月日を選択するフォーム

実行して以下のようになつたら成功だ！！



生年月日を入力するフォーム

西暦 1950 年
1 月 1 日



生年月日を入力するフォーム

西暦 年
1950
月 日

1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969

3-4-4 MEMO: 非同期通信

選択した月によって日数を変更したいよね。例えば1月は31日、2月は28日。でもそれはPHPではなく、JavaScriptの分野なんだ。

PHPでは、月が変更されたことをサーバで感知しなければならないため、一度送信ボタンを押す必要があるよ。

JavaScriptでは、処理をブラウザで行っているため、オプションの選択を感じして、リアルタイムでそれに応じた処理を行うことができる。

これはサーバと連動していないので「**非同期通信**」と呼ばれるよ。

でも、JSの方が優秀かと言うとそうでもなく、JSはデータベースを操作することができない。

言語の得意不得意を理解して開発できることが望ましいですね。

3-4-5: 繰り返し処理の使い分け

以下のように使い分けよう！

- **while文**

- 条件が合う限り繰り返し処理をさせたいとき。
- 繰り返す回数がわからないとき

- **for文**

- 繰り返す回数を指定したい時

3-5: 複雑なデータを管理しよう

はじめよう ! PHP !

3-5: 複雑なデータを管理しよう

さあ、ようやくここまできました。ひとまずお疲れ様です。
連続して勉強するのも大変なので休憩をはさみながらお勉強しましょう。

さて、次の章はデータベースなのですが
データベースから取得したデータは配列の状態になっています。

PHPでの配列が一体どのようなものか、一緒にお勉強していきましょう。

※PHPでの配列はC言語とは少し異なります。一緒にお勉強しましょう。

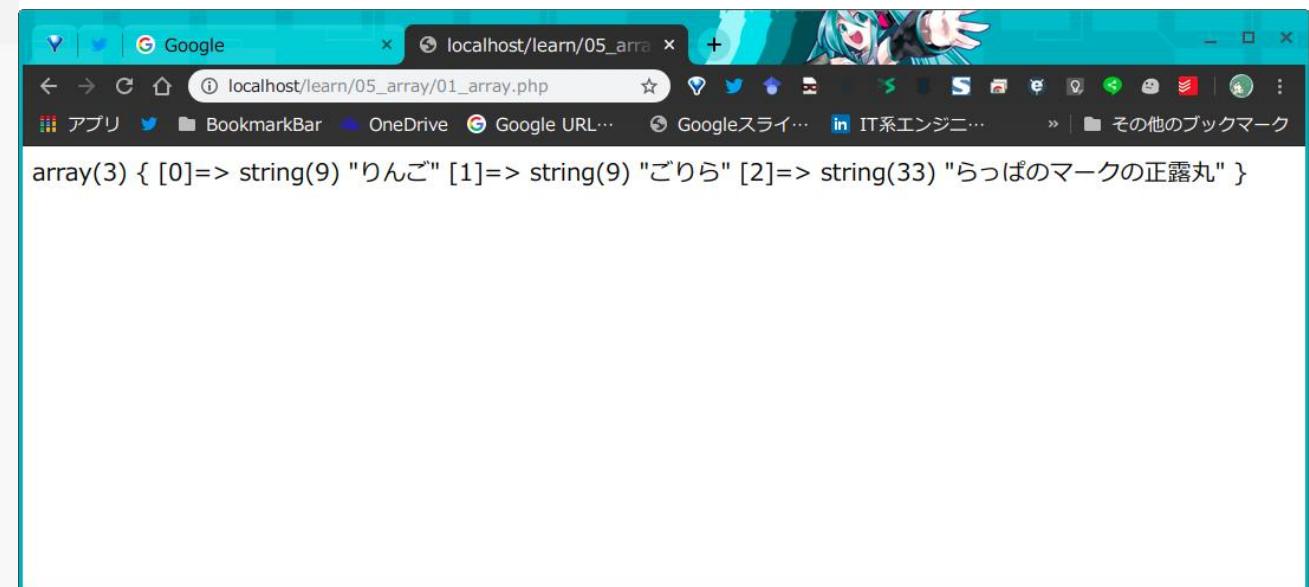
3-5-1：配列の基本

では、一緒に書きながらおぼえていこう！

セクションが変わったのでフォルダを新しく作ってね！（以降は特にうるさく言いません）

では、以下の内容のPHPファイルを作ろう！

```
php 01_array.php ×  
php 01_array.php > ...  
1 <?php  
2  
3 $data = array(); // 配列として初期化  
4  
5 $data[] = 'りんご';  
6 $data[] = 'ごりら';  
7 $data[] = 'らっぱのマークの正露丸';  
8  
9 var_dump($data);
```

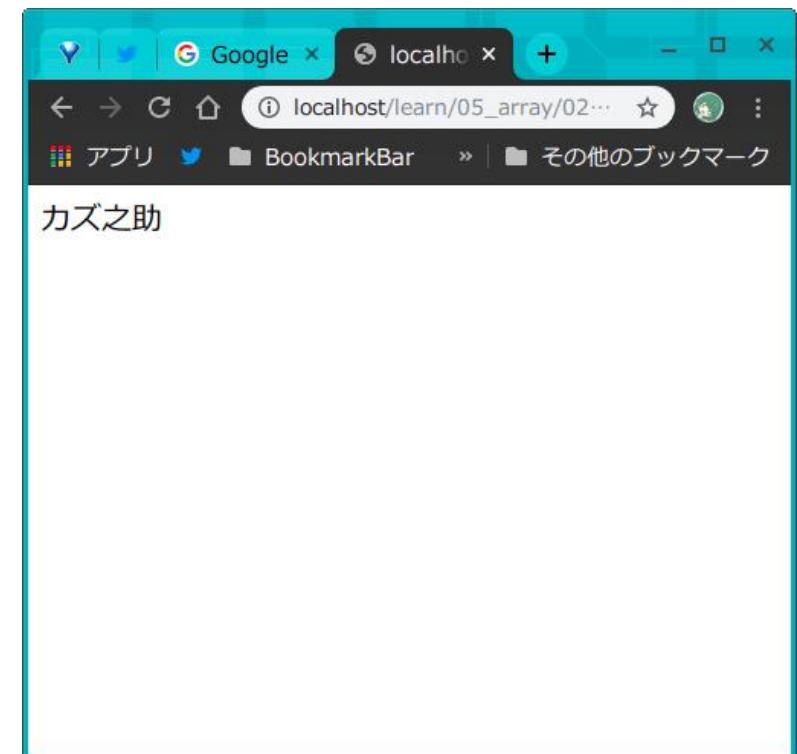


配列はC言語と同じように0から始まる

3-5-2: 連想配列

配列の[]をキーといいます。このキーには添字(自動的に割り振られる値)以外にも、**自分で名付けることができます**。一緒にやってみましょう。

```
02_array_assoc.php ×  
learn > 05_array > 02_array_assoc.php > ...  
1 <?php  
2  
3 $array = array(); // 配列として初期化  
4  
5 $array['name'] = 'カズ之助';  
6 $array['twitter'] = '@Tech_Kazu';  
7 $array['slack'] = 'Kazuki_Kushida - u306065';  
8  
9 echo $array['name'];
```



3-5-2: 連想配列

この配列、キーを見ればなんなく何のデータが入っているかわかりやすいという利点があります。

こんな感じの配列を 「連想配列」 と言います。

データベースから取得したデータはすべてこの連想配列の形になっています。

3-5-3: 配列を効率的に書こう！

実は私、効率厨なんです。せっかくなので
効率的に書く方法を教えてしんぜよう。

3-5-3: 配列を効率的に書こう！

複数のデータをこのように書くことで一気に格納することが可能です。

```
php 03_array_other.php ×  
learn > 05_array > 03_array_other.php > ...  
1 <?php  
2  
3 $array1 = array();  
4 $array2 = array();  
5  
6 // キーが添字になる配列  
7 $array1 = array('りんご', 'みかん', 'バナナ');  
8  
9 // 連想配列  
10 $array2 = array('name' => 'カズ之助', 'age' => 20, 'twitter' => '@Tech_Kazu');
```

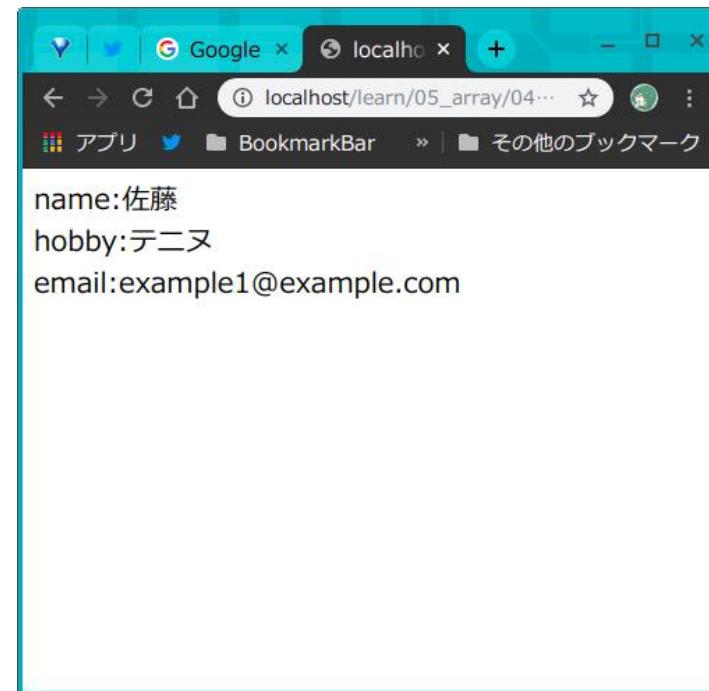
「=>」をダブルアロー演算子といい、連想配列を作るときに使います

3-5-4: 配列専用のループ文！

実はPHPには配列専用の繰返し構文というのが存在します。

やつの名は 「**foreach**」 配列の要素数に応じて繰り返してくれます。
以下のコードを書いてみよう。

```
04_foreach.php ×  
learn > 05_array > 04_foreach.php > ...  
1  <?php  
2  
3  // 配列はキーごとに行を分けておくと見やすいです  
4  $array = array(  
5      'name' => '佐藤',  
6      'hobby' => 'テニス',  
7      'email' => 'example1@example.com'  
8  );  
9  
10 foreach($array as $key => $var){  
11     echo $key. ':' . $var. '<br>';  
12 }
```



3-5-4: 配列専用のループ文！

foreachの構文を確認しましょう。

```
foreach($array as $key => $var) {  
    echo $key . ':' . $var . '<br>';  
}
```

name, hobbyなどのキー名が代入される

「佐藤」, 「テニス」などの値が代入される

文字列, 変数がドットで連結されている

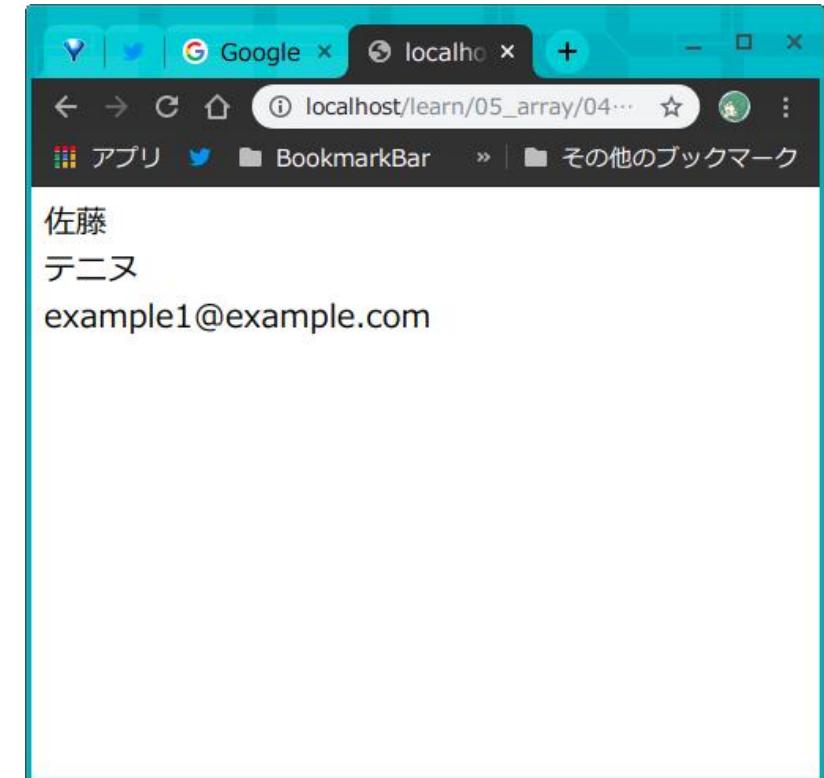
1周目：\$key に name , \$var に 佐藤 が代入

2周目：\$key に hobby , \$var に テニス が代入

3-5-4: 配列専用のループ文！

foreachのキーは省略可能です。試しに先程のforeachをコメントアウトして以下のように書いてみましょう。

```
04_foreach.php ×  
learn > 05_array > 04_foreach.php > ...  
1  <?php  
2  
3 // 配列はキーごとに行を分けておくと見やすいです  
4 $array = array(  
5     'name' => '佐藤',  
6     'hobby' => 'テニス',  
7     'email' => 'example1@example.com'  
8 );  
9  
10 // foreach($array as $key => $var){  
11 //     echo $key.':'.$var.'<br>';  
12 // }  
13  
14 // foreachのキーを省略した書き方  
15 foreach($array as $var){  
16     echo $var.'<br>';  
17 }
```



3-5-5: 二次元配列を理解しよう

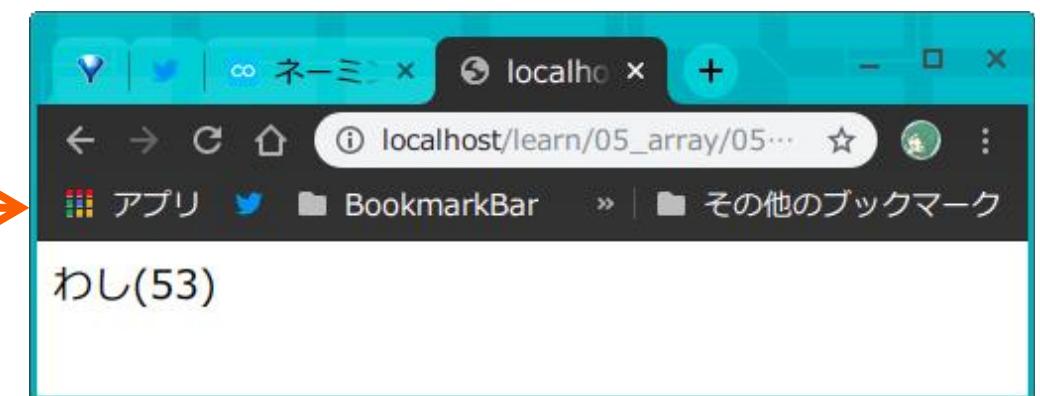
配列の中に配列を入れることで二次元配列を実現することが可能です。

05_two_dimensional_array.php ×

learn > 05_array > 05_two_dimensional_array.php > ...

```
1 <?php
2 $arrays = array(
3
4     0 => array(
5         'name' => '佐藤',
6         'age' => 20,
7         'email' => 'example1@example.com'
8     ),
9
10    1 => array(
11        'name' => '鈴木',
12        'age' => 25,
13        'email' => 'example2@example.com'
14    ),
15
16    2 => array(
17        'name' => 'わし',
18        'age' => 53,
19        'email' => 'example3@example.com'
20    ),
21 );
22
23 echo $arrays[2]['name'].('.'.$arrays[2]['age'].'');
```

配列の中に配列を作る



3-5-6: 二次元配列を表として出力しよう

先程の配列をテーブル(表)として出力してみよう。echo文は消してください。

```
19     'email' => 'example3@example.com'
20   ),
21 );
22 ?>
23 <html>
24   <body>
25     <table border="1">
26       <tr>
27         <th>名前</th><th>趣味</th><th>メアド</th>
28       </tr>
29       <tr>
30         <td>佐藤</td><td>20</td><td>example1@example.com</td>
31       </tr>
32     </table>
33   </body>
34 </html>
```

このコードはタグ内が既に記述されている
静的なページになる

<table>が表
<tr>が行, <th>が列のタイトル
<td>がセルの中身を表す

table		
tr→	th	th
tr→	td	td
tr→	td	td

このコードを変更していきます

3-5-6: 二次元配列を表として出力しよう

foreachで配列の中身を出力していこう

```
21 );  
22 ?>  
23 <html>  
24   <body>  
25     <table border="1">  
26       <tr>  
27         <th>名前</th><th>趣味</th><th>メアド</th>  
28       </tr>  
29       <?php foreach($arrays as $row){ ?>  
30         <tr>  
31           <td><?php echo $row['name']; ?></td>  
32           <td><?php echo $row['age']; ?></td>  
33           <td><?php echo $row['email']; ?></td>  
34         </tr>  
35       <?php } ?>  
36     </table>  
37   </body>  
38 </html>
```

\$rowとして要素を取り出す

\$rowの配列の中身をechoする

3-5-6: 二次元配列を表として出力しよう

chromeで見てみよう下のような感じに出力されれば優秀だ！！



デザイン的な観点から……
echo文でHTMLタグはなるべく出力しないようにしましょう
HTMLとかCSSを書く人が書きやすいようにしうね！

3-5-7: 複数の値を受け取ろう！

実は配列を扱うことで、一度にたくさんのデータを受け取ることができる！
今回はこれで終わりなので頑張ってやってみよう！！

要件定義

- ・checkboxを使って好きな色を選択し、送信できるようにする
- ・取得後のデータはリスト表示とする
- ・出力時はセキュリティを意識してみよう！
→エスケープ処理というものを行います(詳しくは後述)

3-5-7: 複数の値を受け取ろう！

では早速送信ページから作っていこう！これは簡単！HTMLだけでできるよ！

```
checkbox_send.php ×  
learn > 05_array > checkbox_send.php  
1  <html>  
2    <body>  
3      <h1>チェックボックスを使った送信ページ</h1>  
4      <p>好きな色を選択してください(複数選択可能)</p>  
5  
6      <form action="checkbox_receive.php" method="POST">  
7        <p>  
8          <input type="checkbox" name="colors[]" value="青">青  
9          <input type="checkbox" name="colors[]" value="赤">赤  
10         <input type="checkbox" name="colors[]" value="黄">黄  
11         <input type="checkbox" name="colors[]" value="緑">緑  
12         <input type="checkbox" name="colors[]" value="紫">紫  
13         <input type="checkbox" name="colors[]" value="白">白  
14         <input type="checkbox" name="colors[]" value="橙">橙  
15        </p>  
16        <input type="submit">  
17      </form>  
18    </body>  
19  </html>
```

送信するデータが配列であることを示すため
colorsに[]をつける！！

(これを忘れるとなにか送信できない！)

3-5-7: 複数の値を受け取ろう！

続いて、受信ページを組み立てていくよ！まずはHTMLから！

```
php checkbox_receive.php ×  
learn > 05_array > checkbox_receive.php  
1  <?php  
2  // チェックボックスの値を取得する  
3  ?>  
4  
5  <html>  
6      <head>  
7          <meta charset="utf-8">  
8      </head>  
9      <body>  
10         <h1>受信ページ</h1>  
11         <h3>好きな色</h3>  
12  
13     </body>  
14 </html>
```

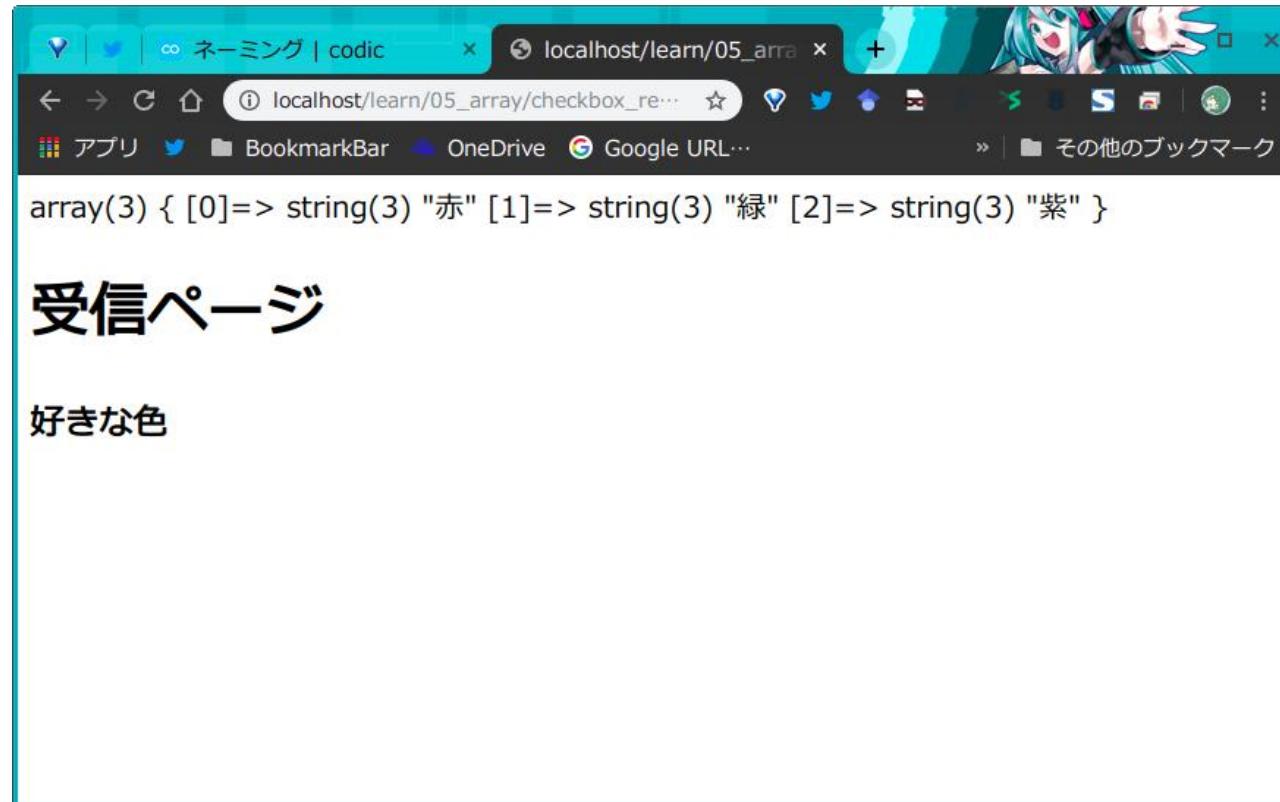
3-5-7: 複数の値を受け取ろう！

データがどのような感じで送られているか確認してみよう

```
checkbox_receive.php ×  
learn > 05_array > checkbox_receive.php > ...  
1  <?php  
2  // チェックボックスの値を取得する  
3  $colors = $_POST['colors'];  
4  
5  // データがどのような形式で送られているか確認してみよう  
6  var_dump($colors);  
7  ?>  
o
```

3-5-7: 複数の値を受け取ろう！

なるほど、データは自動的に添字がつく形で送信されているね。
この場合、個別に値を取り出すには \$colors[0]などと記述するといいよ！
では、\$var_dumpは消しておこう！



3-5-7: 複数の値を受け取ろう！

受け取った値をforeachでリスト表示しよう！

```
10 <body>
11     <h1>受信ページ</h1>
12     <h3>好きな色</h3>
13     <ul>
14         <?php foreach($colors as $row){ ?>
15             <li><?php echo $row; ?></li>
16         <?php } ?>
17     </ul>
18 </body>
```

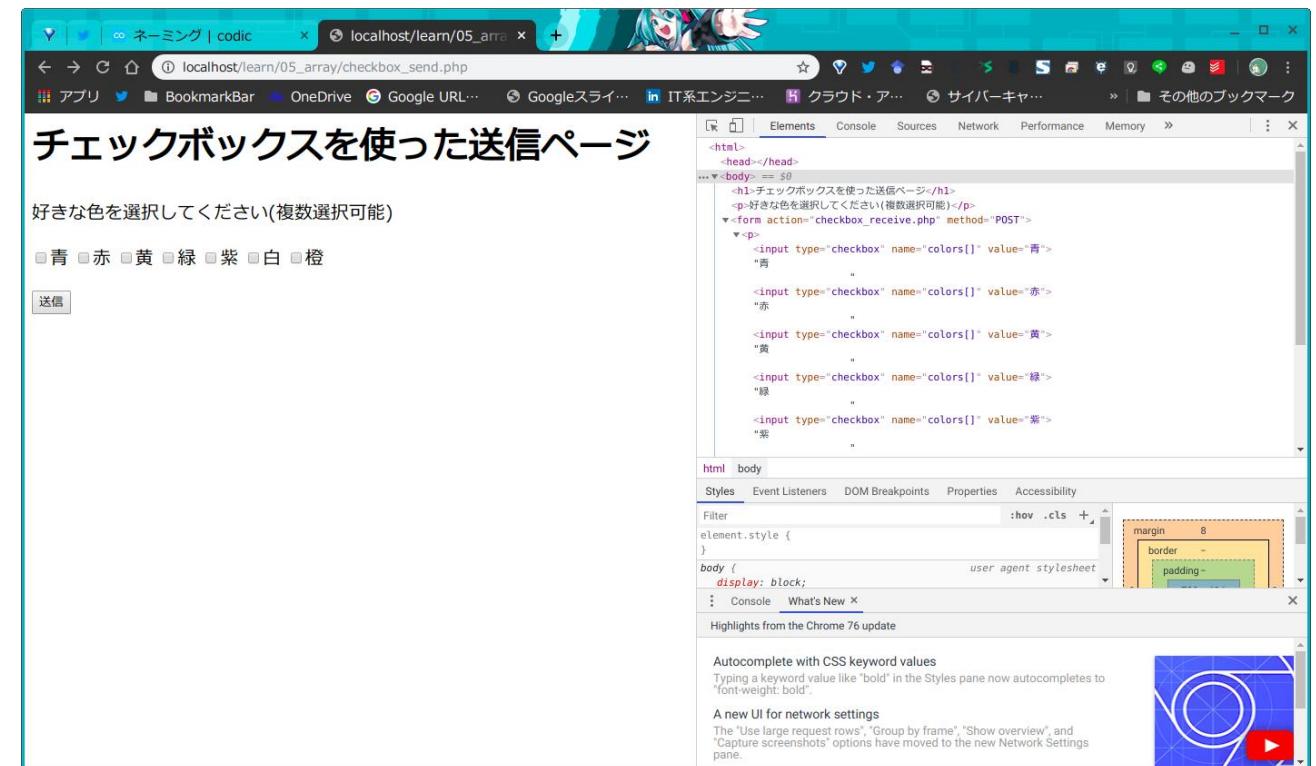
配列の要素数分繰り返す

3-5-7: 複数の値を受け取ろう！

これだけで一応できしたことにはできたけど……

セキュリティ的にはガバガバなのです。

試しに、chromeでcheckbox_send.phpを開いて
F12キーを押してみてください。



3-5-7: 複数の値を受け取ろう！

では、青のチェックボックスのvalueを変更してみましょう。

value = '青'の 「青」 の上でダブルクリックしてみてください。

このように編集可能な状態になります。

```
method="POST">>  
rs[]" value="青"> ==
```

3-5-7: 複数の値を受け取ろう！

私は**悪意を持って編集している**ので、HTMLタグを埋め込んでやりましょう。

```
  ページ</h1>
  数選択可能)</p>
  live.php" method="POST">

    name="colors[]" value="
```

```
    name="colors[]" value="赤">
```

3-5-7: 複数の値を受け取ろう！

このまま青を選択して送信を押すと……？

チェックボックスを使った送信ページ

好きな色を選択してください(複数選択可能)

青 赤 黄 緑 紫 白 橙

送信

3-5-7: 複数の値を受け取ろう！

なんてこった！！リンクになっちまった！！

しかも<a>タグが閉じられていないから悲惨なことになってる！！

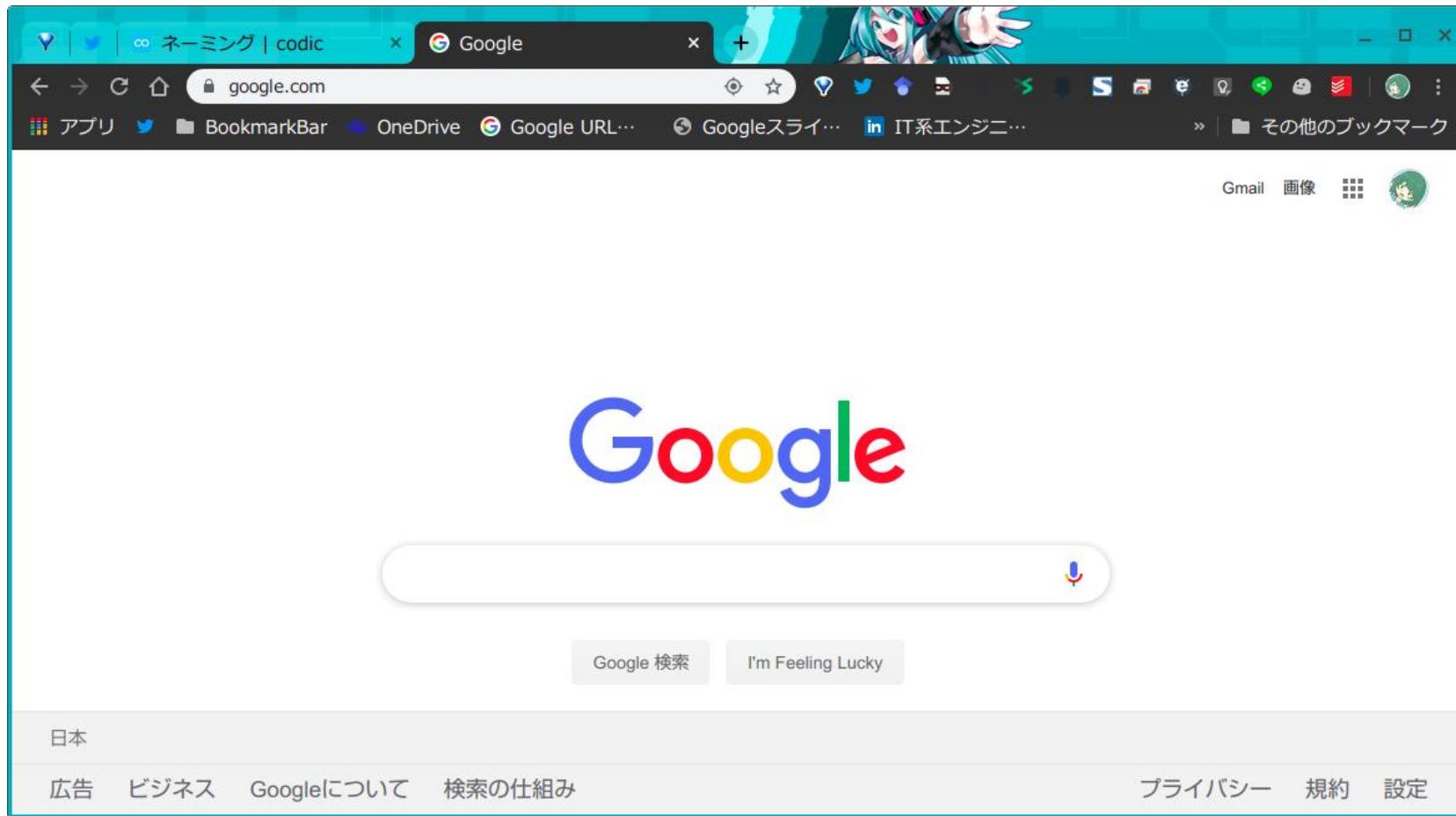
受信ページ

好きな色

- [google](#)
- [黄](#)
- [緑](#)

3-5-7: 複数の値を受け取ろう！

あーあ……



3-5-7: 複数の値を受け取ろう！

……と、HTMLタグが埋め込まれるところなってしまいます。

これではいけません。修正しましょう！

少なくともHTMLタグとして解釈されてしまう<>や
プログラム的な効果のある'などを無効化する必要があります。

3-5-7: 複数の値を受け取ろう！

ここで使用する関数は `htmlspecialchars()` というものです。

`htmlspecialchars(文字列, オプション, 文字コード)`

それでは、早速修正していきましょう！

3-5-7: 複数の値を受け取ろう！

このようにecho の次にhtmlspecialcharsを続けて書こう！

```
12     <h3>好きな色</h3>
13     <ul>
14         <?php foreach($colors as $row){ ?>
15             <li><?php echo htmlspecialchars($row, ENT_QUOTES, 'utf-8'); ?></li>
16         <?php } ?>
17     </ul>
18     </body>
19 </html>
```

3-5-7: 複数の値を受け取ろう！

実行して、valueの中身を先程と同じような手順で書き換えてみると……



タグとして解釈されていない！！安全！！

3-5-7: 複数の値を受け取ろう！

今回はちょっとやりすぎ感のあるプログラムだったけれど、
チェックボックスの値も自由に変更できてしまうから、バリデーションや、
エスケープ処理などで意図したデータが送信されるように
二重三重にセキュリティを施そう！

まあ合宿なんでそこまでガチガチに固めなくても別に問題はないけどね！
楽しくやろう！

というわけで今回はここまで！次回はデータベースから！
次回も頑張っていこう！！