第４回　フォーム処理②

４－１．POSTでのデータ送信

■POSTデータを送信

<form>開始タグ

|  |  |
| --- | --- |
| methodプロパティ | POST |
| actionプロパティ | データの送信先を記載 |

■SampleCode（テキストボックスの内容をPOSTで送信）

「name="name"」　「name="tel"」　「name="address"」３件をPOSTで送信

<form action="sample01\_4.php" method="POST">

  <label for="name" class="form-label">氏名</label>

<input type="text" id="" class="form-control form-control-lg" name="name">

  <label for="tel" class="form-label">電話番号</label>

  <input type="text" id="" class="form-control form-control-lg" name="tel">

  <label for="address" class="form-label">住所</label>

  <input type="text" id="" class="form-control form-control-lg" name="address">

  <div class="p-5 d-grid gap-2 d-md-flex justify-content-md-end">

    <button type="submit" class="btn btn-danger btn-lg">登録</button>

  </div>

</form>

■Sample（テキストボックスの内容をPOSTで送信・ブラウザ表示）　※入力画面



Sample画面の[登録]ボタン押下時、POST形式でデータが送信されます。

GET形式と違って、遷移先（今回のSampleではsample01\_4.php）のURLにパラメータは表示されません。（下記、赤枠部分）



４－２．POSTデータの受け取り

GETデータ同様に、スーパーグローバル変数で受け取ることができます。

■POSTデータを取得できるスーパーグローバル変数

|  |  |
| --- | --- |
| $\_POST | HTTP POST変数  HTTP POST メソッドから現在のスクリプトに渡された変数の連想配列です。 |

■SampleCode　１　（テキストボックスの内容を受け取り、変数に格納）

<?php

//パターン１

//グローバル変数を各変数にそのまま取得

$name = $\_POST["name"];

$tel = $\_POST["tel"];

$address = $\_POST["address"];

?>

■SampleCode　２　（テキストボックスの内容を受け取り、連想配列に格納）

<?php

//パターン２

//グローバル変数を連想配列にそのまま取得

$result["name"] = $\_POST["name"];

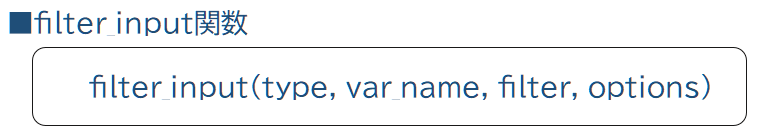
$result["tel"] = $\_POST["tel"];

$result["address"] = $\_POST["address"];

?>

スーパーグローバル変数では、送信されたデータをそのまま受け取るだけですので、続いて、値のフィルタリングを行ってみます。

■filter\_input関数



テキスト

中程度の精度で自動的に生成された説明

■引数

|  |  |
| --- | --- |
| type | 下記のいずれか  INPUT\_GET  INPUT\_POST  INPUT\_COOKIE  INPUT\_SERVER  INPUT\_ENV |
| var\_name | 取得する変数の名前 |
| filter | 適用するフィルタのID。省略した場合はFILTER\_DEFAULT。デフォルトでは、結果として、何もフィルタリングしません。  ■一部抜粋・概要   |  |  |  | | --- | --- | --- | | フィルタID | オプション | 説明 | | FILTER\_VALIDATE\_INT | default,  min\_range,  max\_range | 値が整数であるかどうか、オプションで指定した範囲内にあるかどうかを検証し、成功した場合は整数に変換します。 | | FILTER\_VALIDATE\_EMAIL | default | 値が妥当な e-mail アドレスであるかどうかを検証します。 |   **フィルタIDや型の詳細は公式HP参照。**  <https://www.php.net/manual/ja/filter.filters.php> |
| options | オプション指定が可能なフィルタの場合、フィールドフラグを指定します。  filterの値による。 |

■SampleCode　３　（filter\_input関数使用。テキストボックスの内容を受け取り、変数に格納）

※「name」と「address」は文字列が入ってくることが想定されるためフィルタなし。

※「tel」のみint型でフィルタ。（ハイフンは許容しない仕様と仮定。）

電話番号なので最小値と最大値などのオプション指定なし。

<?php

//パターン３

//filter\_input関数を用いて、変数に取得

$name = filter\_input(INPUT\_POST, "name");

$tel = filter\_input(INPUT\_POST, "tel", FILTER\_VALIDATE\_INT);

$address = filter\_input(INPUT\_POST, "address");

?>

■SampleCode　４　（filter\_input関数使用。テキストボックスの内容を受け取り、連想配列に格納）

<?php

//パターン４

//filter\_input関数を用いて、連想配列に取得

$result["name"] = filter\_input(INPUT\_POST, "name");

$result["tel"] =　filter\_input(INPUT\_POST, "tel", FILTER\_VALIDATE\_INT);

$result["address"] = filter\_input(INPUT\_POST, "address");

?>

■SampleCode　（HTML部分）　※一部抜粋

　※パターン１ or　パターン３　それぞれ独立した変数に値を格納している場合

<p class="form-control form-control-lg"><?= $name ?></p>

<p class="form-control form-control-lg"><?= $tel ?></p>

<p class="form-control form-control-lg"><?= $address ?></p>

■SampleCode　（HTML部分）　※一部抜粋

　※パターン２ or　パターン４　連想配列に値を格納している場合

<p class="form-control form-control-lg"><?= $result["name"] ?></p>

<p class="form-control form-control-lg"><?= $result["tel"]　?></p>

<p class="form-control form-control-lg"><?= $result["address"] ?></p>

■Sample（テキストボックスの内容をPOSTで受信・ブラウザ表示）　結果画面

※スーパーグローバル変数で値をそのまま取得した場合

※**filter\_input関数を使用していても、「tel」に数値のみ入力**されていた場合



■Sample（テキストボックスの内容をPOSTで受信・ブラウザ表示）　結果画面

※**filter\_input関数を使用し、「tel」に数値以外が入力**されていた場合

・入力「電話番号」　　　　　　　　　　　　　　　　　　　　　・結果

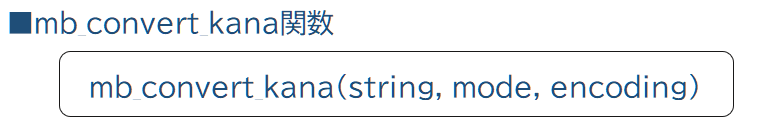
**FILTER\_VALIDATE\_INTでフィルタリング失敗=falseが返っている。**

**falseは表示すると空文字になるので何も表示されていない。**

続いて、文字列をプログラム上で扱いやすいように変換してみます。

■mb\_convert\_kana関数

文字列の値をmb\_convert\_kana関数を用いて、スペースは半角、カタカナは全角、濁点付きの文字は1文字に変換します。





■引数

|  |  |
| --- | --- |
| string | 変換される文字列 |
| mode | 変換オプション   |  |  | | --- | --- | | r | 英字「全角」→「半角」 | | R | 英字「半角」→「全角」 | | n | 数字「全角」→「半角」 | | N | 数字「半角」→「全角」 | | a | 英数字「全角」→「半角」 | | A | 英数字「半角」→「全角」 | | s | スペース「全角」→「半角」 | | S | スペース「半角」→「全角」 | | k | カタカナ「全角」→「半角」 | | K | カタカナ「半角」→「全角」 | | h | ひらがな「全角」→「半角」 | | H | ひらがな「半角」→「全角」 | | c | 「全角カタカナ」→「全角ひらがな」 | | C | 「全角ひらがな」→「全角カタカナ」 | | V | 濁点付きの文字を1文字に変換。"K", "H" と共に使用。 | |
| encoding | 文字列のエンコーディングを指定。省略可。省略した場合は、内部文字のエンコーディングを使用。 |

■SampleCode　※抜粋。

※スペースを半角、カナは全角に変換

<?php

　 //～（＊前述のコードは中略。）～

//mb\_convert\_kanaで「名前」「住所」を変換。

// s→スペース「全角」→「半角」

// K→カタカナ「半角」→「全角」

// V→濁点付きの文字を1文字に変換。

$result["name"] = mb\_convert\_kana($result["name"], "sKV", "UTF-8");

$result["address"]

= mb\_convert\_kana($result["address"], "sKV", "UTF-8");

?>

■Sample（mb\_convert\_kana使用）

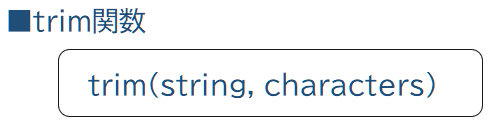
・入力　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　・結果

続いて、文字列の前後の空白の除去を行います。「　大阪市中崎町１－１０　」　のように前後に空白が入っていた場合、不要なので除去しておきます。「大阪市　中崎町　１－１０」のような文字と文字の間にある空白は除去されません。

■trim関数

1. 文字列に関しては、trim関数を用いて、前後の空白を除去してください。





■引数

|  |  |
| --- | --- |
| string | 空白を取り除く文字列 |
| characters | 削除する文字列を指定することも可能。省略可。 |

■SampleCode　※抜粋。

<?php

　 //～（＊前述のコードは中略。）～

　 //mb\_convert\_kanaの後に記述。

//trimで「名前」「住所」の前後空白を除去。

$result["name"] = trim($result["name"]);

$result["address"] = trim($result["address"]);

?>

■Sample（trim使用）

・入力　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　・結果

　グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明