ソフトウェア開発
第 10 回目授業

平野 照比古

2018/12/7

問 1

- `if(変数)` の動作の確認がない、または間違っている内容のものが多かった。
- 基本的には 3 項演算子?を用いればよい。
- `print("0"?"true":"false");` などと書けばよい。
- ルーブリックに挙げた例は JavaScript と PHP で結果が異なる。

問 2

- print_r での表示では同じに見えるものが、var_dump では違いが分かる例を示したものがなかった。
- サーバーを通じた実行結果の場合にはページのソースで確認をしてほしい。

問 3

- pop や unshift では戻り値の確認も必要

```
<?php
$a = [1,2,3,4,5];
print array_splice($a, -1, 1)[0]."\n";//POP
print_r($a);
array_splice($a, count($a), 0, 10);//PUSH
print_r($a);
print array_splice($a, 0, 1)[0]."\n";//SHIFT
print_r($a);
array_splice($a, 0, 0, [-1, -2]);//UNSHIFT
print_r($a);
function pop(&$a) {
    return array_splice($a, -1, 1)[0];
}
print pop($a)."\n";
print_r($a);
```

問 4

4.1

- 実行途中で表示される警告文に対する考察が足りない。
- 関数 sum2 において変数のスコープに関する考察が欲しい

4.2

問題文で文字列をテンプレートリテラルに直すように指示があったのに通常の文字列のままの解答であった。

4.3

- 解答が中たものが多かった。
- 変数の宣言、スコープの違い、関数の定義などに違いがある。

サーバーとのデータ交換の基本

Web ページにおいてサーバーにデータを送る方法には POST と PUT の 2 通りの方法がある。

POST による送信

windows.onload =function() 内に次のコードを追加する。

```
let Form = document.getElementsByTagName("form")[0];
Form.setAttribute("method","POST");
Form.setAttribute("action","09sendData.php");
```

HTML の要素に対しては次のことを行う。

- <select>要素の属性に name="select" を追加する。
- id が"colorName"であるテキストボックスに name="colorName" を追加する。
- 「設定」ボタンの要素の後に次の要素を追加する。

```
<input type="submit" value="送信" id="Send"></input>
```

POST による送信 (解説)

- このページでは「送信」ボタンを押すと<form>の action 属性で指定されたプログラムが呼び出される。
- ここでは Web ペー ジと同じ場所にある 09sendData.php が呼び出される。

サーバープログラムのリスト

```
<?php
print <<<_EOL_
<!DOCTYPE html>
<head>
<meta charset="UTF-8"/>
<title>サーバーに送られたデータ</title>
</head>
<body>
<table>
_EOL_;
foreach($_POST as $key=>$value) {
 print "<tr><td>$key</td><td>$value</td></tr>\n";
}
print <<<_EOL_
</table>
</body>
</html>
_EOL_;
?>
```

サーバープログラムのリスト (解説)

- 2 行目から 10 行目の間はヒアドキュメント形式で HTML 文書の初めの部分を出力させている。
- `method="POST"` で呼び出されたときには `form` 要素内の `name` 属性が指定されたものの値がスーパーグローバル`$_POST` 内の連想配列としてアクセスができる。
- フォームの中のデータを `application/x-www-form-urlencoded` あるいは `multipart/form-data` によりエンコードされたもの
- 11 行目から 13 行目でそれらの値を `table` 要素内の要素として出力している。

サーバープログラムの結果

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8"/>
<title>サーバーに送られたデータ</title>
</head>
<body>
<table><tr><td>select</td><td>yellow</td></tr>
<tr><td>color</td><td>green</td></tr>
<tr><td>colorName</td><td>gray</td></tr>
</table>
</body>
</html>
```

GET による通信

- method="PUT"で呼び出した場合にはスーパーグローバル$_GET を用いる。
- スーパーグローバル$_REQUEST は method="POST"でも method="PUT"で呼び出された場合の$_POST や$_GET の代わりに使用できる (非推奨)。

通信に関する注意

- type="submit"の input 要素は、ボタンが押されたときに直ちに、action 属性で指定された処理が呼び出される。
- サーバーにデータを送る前に最低限のエラーチェックを行い、エラーがない場合にだけサーバーと通信するのが良い。

スパーグローバルの補足

- $_SERVER
  サーバーにアクセスしたときのクライアントの情報などを提供。具体的な内容はクライアントごとに異なる。
- $_COOKIE
  COOKIE とは Web サーバー側からクライアント側に一時的にデータを保存させる仕組み。すでに訪問したことがあるサイトに対して情報を開始時に補填する機能などを実現できる。
- $_SESSION
  セッションとはある作業の一連の流れを指す。たとえば会員制のサイトではログイン後でなければページを見ることができない。情報のページに直接行くことができないような仕組みが必要

セッション

- HTTP 通信はセッションレスな通信である（各ページが独立して存在し、ページ間のデータを直接渡せない）
- セッションを確立するためには、クライアント側から情報を送り、それに基づいてサーバー側が状況を判断するなどの操作を意識的にする必要
- PHP ではセッションを開始するための関数 `session_start()` とセッションを終了させる `session_destroy()` が用意されている。
- セッションを通じで保存させておきたい情報はこの連想配列に保存
- セッションの管理はサーバーが管理
- この機能は COOKIE の機能を利用して実現

Web Storage

- localStoarage と sessionStorage の 2 種類
- localStorage は文字列をキーに、文字列の値を持つ Storage オブジェクト
- 同一の出身 (プロトコルやポート番号も含む) のすべてのドキュメントがおなじ localStorage を共有
- このデータは意識的に消さない限り存在
- sessionStorage はウインドウやブラウザが閉じられると消滅
- セッション間の情報の移動を可能にしている。

Web Storage の補足

- いくつかのサイトではこの機能を用いており、その開発者ツールで見ることが可能
- データの形式は文字列である。構造化されたデータは JSON 形式で保存するのがよい

WebStrage の例 (1)

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>WebStorage --- localStorage</title>
```

WebStrage の例 (2)

```
<script type="text/javascript">
//<![CDATA[
let Storage = window.localStorage;
//let Storage = window.sessionStorage;
window.onload = function() {
  let AccessList, Message = document.getElementById("message");
  let D = new Date();
  if(Storage["access"]) {
    AccessList = JSON.parse(Storage["access"]);
  } else {
    AccessList = [];
    appendMessage(Message, "初めてのアクセスです");
  }
  AccessList.unshift(D.getTime());
  Storage["access"] = JSON.stringify(AccessList);
  appendMessage(Message, "今までのアクセス時刻です");
  AccessList.forEach(function(D, i, A) {
    appendMessage(Message, new Date(D));
  });
}
```

WebStrage の例 (3)

```
function appendMessage(P, Mess) {
  let div = document.createElement("div");
  P.appendChild(div);
  div.appendChild(document.createTextNode(Mess));
}
```

WebStrage の例 (4)

```
//]]
</script>
</head>
<body>
  <form action="10next.html">
    <input type="submit" value="次のページ"></input>
  </form>
  <div  id="message"/>
</body>
</html>
```

WebStrage の例 (5)

```
<!DOCTYPE html>
<head>
  <meta charset="UTF-8"/>
  <script type="text/javascript">
    //<![CDATA[
    let Storage = window.localStorage;
    //let Storage = window.sessionStorage;
```

WebStrage の例 (6)

```
window.onload = function() {
  let AccessList, Message = document.getElementById("message");
  appendMessage(Message, "新しいページです");
  if(Storage["access"]) {
    AccessList = JSON.parse(Storage["access"]);
  } else {
    AccessList = [];
    appendMessage(Message, "初めてのアクセスです");
  }
  appendMessage(Message, "今までのアクセス時刻です");
  AccessList.every(function(D, index, A) {
    appendMessage(Message, new Date(D-0));
    return index < 3;
  });
}
```

WebStrage の例 (7)

```
    function appendMessage(P, Mess) {
      let div = document.createElement("div");
      P.appendChild(div);
      div.appendChild(document.createTextNode(Mess));
    }
    //]]
  </script>
</head>
<body>
  <div  id="message"/>
</body>
</html>
```

Ajax とは

- Ajax とは Asynchronous Javascript+XML の略で、非同期 (Asynchronous) で Web ページとサーバーでデータの交換を行い、クライアント側で得られたデータをもとにその Web ページを書き直す手法
- Google Maps がこの技術を利用したことで一気に認知度が高まった。
- Ajax の機能は `XMLHTTPRequest` というオブジェクトの機能を用いて実現

Ajax の例

- 日付が変わったときに、その日の記念日をメニューの下部に示すもの
- http://ja.wikipedia.org/wiki/日本の記念日一覧の表示画面からコピーして作成

Ajax の例 (2)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
5 <title>今日は何の日</title>
6 <script type="text/ecmascript">
7 //<![CDATA[
8   window.onload = function(){
```

Ajax の例 (3)

```
 9     function makeSelectNumber(from, to, prefix, suffix, id, parent){
10       let Select = makeElm("select", {"id":id}, parent);
11       for(let i=from; i<=to; i++) {
12         let option = makeElm("option",{ "value":i}, Select);
13         makeTextNode(prefix+i+suffix,option);
14       }
15       return Select;
16     };
```

Ajax の例 (4)

```
17    function makeElm(name, attribs, parent) {
18      let elm = document.createElement(name);
19      for(let attrib in attribs) {
20        elm.setAttribute(attrib,attribs[attrib]);
21      };
22      if(parent) parent.appendChild(elm);
23      return elm;
24    }
25    function makeTextNode(text,parent) {
26        parent.appendChild(document.createTextNode(text));
27    };
```

Ajax の例 (5)

```
28    let Form  = document.getElementById("menu");
29    let Year  = makeSelectNumber(2000,2020,"","年","year", Form);
30    let Month = makeSelectNumber(1,12,"","月","month", Form);
31    let Days = [];
32    for(let d=28; d<=31; d++) {
33      Days[d] = makeSelectNumber(1,d,"","日","day");
34    }
35    let today = new Date();
36    Year.value = today.getFullYear();
37    Month.value = today.getMonth()+1;
38    let d = new Date(Year.value, Month.value,0).getDate();
39    Form.appendChild(Days[d]);
40    Form.children[2].value = today.getDate();
```

Ajax の例 (5)

```
28    let Form  = document.getElementById("menu");
29    let Year  = makeSelectNumber(2000,2020,"","年","year", Form);
30    let Month = makeSelectNumber(1,12,"","月","month", Form);
31    let Days = [];
32    for(let d=28; d<=31; d++) {
33      Days[d] = makeSelectNumber(1,d,"","日","day");
34    }
35    let today = new Date();
36    Year.value = today.getFullYear();
37    Month.value = today.getMonth()+1;
38    let d = new Date(Year.value, Month.value,0).getDate();
39    Form.appendChild(Days[d]);
40    Form.children[2].value = today.getDate();
```

Ajax の例 (6)

```
41    let changePulldown = function(){
42      let d2 = Form.children[2].value
43      d = new Date(Year.value, Month.value, 0).getDate();
44      if( d != Form.children[2].children.length) {
45        Form.replaceChild(Days[d],Form.children[2]);
46        d2 = Math.min(Form.children[2].length, d2);
47        Form.children[2].value = d2;
48      }
```

Ajax の例 (7)

```
49      let  xmlHttpObj = new XMLHttpRequest();
50      if(xmlHttpObj) {
51        xmlHttpObj.onreadystatechange = function(){
52          if(xmlHttpObj.readyState == 4 && xmlHttpObj.status == 200) {
53            document.getElementById("details").innerText = xmlHttpObj.respo
54          }
55        }
56        xmlHttpObj.open("GET",
57          `./aniversary.php?month=${Month.value}&day=${d2}`,true);
58        xmlHttpObj.send(null);
59      }
60    }
61    Form.addEventListener("change", changePulldown,false);
62    changePulldown();
63  }
```

Ajax の例 (5)

```
64 //]]>
65 </script>
66 </head>
67 <body>
68   <form id="menu"></form>
69   <p id="details"></p>
70 </body>
71 </html>
```

Ajax で呼び出される PHP のプログラム

```
 1 <?php
 2 mb_internal_encoding("UTF8");
 3 //print mb_internal_encoding();
 4 $m = isset($_GET["month"])?$_GET["month"]: $argv[1];
 5 $d = isset($_GET["day"])?$_GET["day"]:$argv[2];
 6 $data = file("aniversary.txt",FILE_IGNORE_NEW_LINES);
 7 for($i=0;$i<count($data);$i++) {
 8   $data[$i] = mb_convert_encoding($data[$i],"UTF8");
 9   $mm = mb_split("\[",$data[$i]);
10   if(count($mm) >1) {
11     if(mb_convert_encoding($m."月","UTF8") === $mm[0]) break;
12   }
13 }
14 for($i++;$i<count($data);$i++) {
15   $data[$i] = mb_convert_encoding($data[$i],"UTF8");
16   $dd = mb_split("\s-\s",$data[$i]);
17   if(($d."日") === $dd[0]) break;
18 }
19 //  print mb_convert_encoding($dd[1],"SJIS");
20 print $dd[1];
21 ?>
```

Ajax で呼び出される PHP のプログラム–解説

- 2 行目で内部で処理をするエンコーディングを UTF8 にしている。関数、mb_internal_encoding 関数を引数なしで呼び出すと現在採用されているエンコーディングを得ることができる。
- 4 行目と 5 行目では月 ($m) と日 ($d) の値をそれぞれの変数に設定している。
  - ここではコマンドプロンプトからもデバッグできるように、スーパーグローバル$_GET 内に値があれば (isset()) が true になれば、その値を、そうでなければコマンドからの引数を設定している。
  - スーパーグローバル$argv はの先頭は呼び出したファイル名であり、その後に引数が順に入る [1]。

[1] C 言語の main 関数は通常、int main(int argc, char* argv[]) と宣言される。argc は argv の配列の大きさを表し、渡された引数のリストが argv[] に入っている。このとき、argv[0] は実行したときのファイル名が入る。

file() 関数

指定されたファイルを行末文字で区切って配列として返す関数

- 引数には URL も指定できる。
- この関数は 2 番目の引数をとることができる。

FILE_USE_INCLUDE_PATH	include_path のファイルを探す
FILE_IGNORE_NEW_LINES	配列の各要素の最後に改行文字を追加しない
FILE_SKIP_EMPTY_LINES	空行を読み飛ばす

- file_get_contents() はファイルの内容を一つの文字列として読み込む。Web ページの解析にはこちらの関数を使うとよい。

読み込むファイルの一部

1 月 [編集]
1 日 – 鉄腕アトムの日
2 日 – 月ロケットの日
[中略]
31 日 – 生命保険の日、愛妻家の日
2 月 [編集]
1 日 – テレビ放送記念日、ニオイの日
2 日 – 頭痛の日
[以下略]

- 月の部分の後には [がある。
- 日の情報は␣-␣で区切られている (␣は空白を表す)。
- すべての日の情報が入っている。

データの処理—月を探す

- 8 行目で念のためコードを UTF8 に変更
- 関数 mb_split() 関数は第 1 引数に指定された文字列パターンで第 2 引数で指定された文字列を分割して配列として返す関数
- 分割を指定する文字列には正規表現が使えるので、文字 [で分割するために、"\["としている (9 行目)。
- 指定された文字列があれば配列の大きさが 1 より大きくなる。その行に対して求める月と一致しているか判定し、等しければループを抜ける (11 行目)。

- 14 行目から 18 行目までは指定された月での指定された日の情報を探している。日を決定する方法も月と同じである。文字列の分割は"\s-\s"となっている[2]。
- 20 行目で得られた情報をストリームに出力している。

[2] これは"\s"ではうまく行かなかったためである。