サーバサイドプログラミング

4. PDO (PHPからSQLアクセス)

コンテンツメディアプログラミング演習 Ⅱ

2014年

菊池，斉藤

1. PDO概要

- PDO (PHP Data Object)
  - PHP5.1から採用された＿＿＿SQLの標準クラス.
  - ＿＿＿＿＿＿＿＿＿＿を採用し，オブジェクトからメソッドやクラス変数を操作する.
  - MySQL, SQLiteなどのサーバソフトに依存せず，ほぼ共通のコードでプログラミングできる.

PHPからSQL文の実行

- PDOオブジェクト作成（準備）

```
$変数 = new PDO("ドライバ:
            host=ホスト;
            dbname=データベース名);
```

  - » ドライバ = mysql，sqlite など
  - » hostはローカルの場合は省略可能
  - » "dbname="　を省略可能
  - » $変数がメソッドなどを含むオブジェクト.

- 例）Mtsデータベースにアクセスする

```
$db = new PDO("sqlite:mts.sqlite");
```

SQLコマンドの実行

- query (クエリ)

  $変数->　　(SQL文);

  » 任意のSQL文を実行する. $変数はPDOオブジェクト.

  ❑例）$db->query("INSERT INTO mts(name) VALUES('高尾山')");

  この場合は返り値はなし(VOID)

検索結果の抽出 (1/3)

- fetch (物を取に行く)

```
$検索結果 = $変数->query(SELECT文);
$行 = $検索結果->fetch();
```

  - » SELECT文は複数の行と列を含むテーブルを返す
  - » fetch()はテーブルから1行づつ取り出す.
  - » 返し値 = _____ (もう行がない時)
    列から成る(通常の)配列
    列名をメンバとする連想配列

- 例) $rows = $db->query("SELECT * from mts"); テーブルmtsから 1行 $rowsに取出す.

検索結果の抽出 (2/3)

- 例1） 配列で列を取出す
  $cols = $rows->fetch()
  print $cols[0] . $cols[1] . $cols[2] . $cols[3]. "¥n";
  1谷川岳05
  2丹沢06
- 例2)　配列を "-"で繋いで出力.
  print join($cols,"-") . "¥n";
  1-1-谷川岳-谷川岳-0-0-5-5-1227-1227
- 例3) 連想配列で名前と標高のみ出力.
  print $cols['name'] ."¥t". $cols['height'] . "¥n";
  1谷川岳05
  2丹沢06
- 例4) 全ての行を出力.
  while($cols = $rows->fetch()){
  　print $cols['name'] . ",";
  }
  谷川岳,丹沢,天城山,八ヶ岳,那須岳,駒ヶ岳,燕岳,奥穂岳,

検索結果の抽出 (3/3)

- fetchAll (全ての行を抽出)

```
$検索結果 = $変数->query(SELECT文);
$表 = $検索結果->fetchAll();
```

  - » fetchAll()は全部の行から成るテーブル全体を取出す.
  - » 返し値 =「列名をメンバとする連想配列」の配列

❑ 例）

```
$rows = $db->query(
"SELECT * from mts");
$all = $rows->fetchAll();
print_r($all);
```

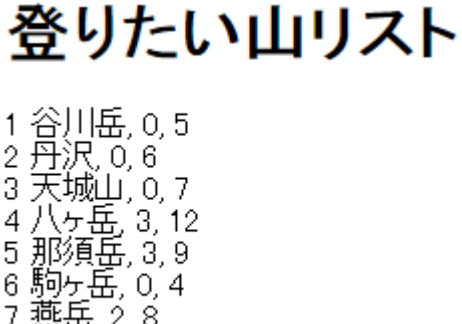
全行をまとめて表示

```
Array
(
    [0] => Array
        (
            [ID] => 1
            [0] => 1
            [name] =>谷川岳
            [1] =>谷川岳
            [day] => 0
            [2] => 0
            [hour] => 5
```

サンプル1

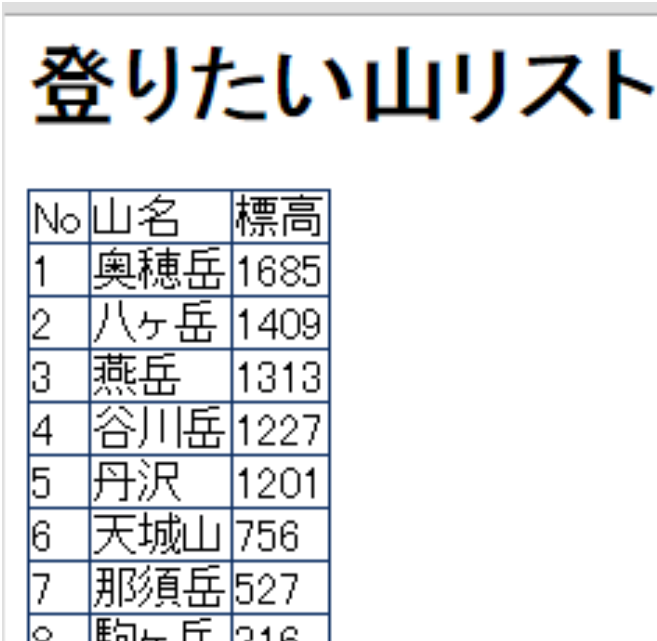
- mts-list.php

```
<html>
<head> <meta charset='UTF-8'> <title> 山登り </title> </head>
<body> <H1> 登りたい山リスト </h1>
<?php
        $db = new PDO("sqlite:Mts-u.sqlite");
        $rows = $db->query('SELECT * FROM mts');
        $i = 0;
        while($cols = $rows->fetch()){
                print   ++$i . " " . $cols[1] .", ". $cols[2] . ", ". $cols[3].
"<br>¥n";
        }
?>
</body> </html>
```



演習1

- Mts.sqlite データベースを用いて，標高順に山名を次の様に出力するmts-top.phpを書き，ブラウザから閲覧せよ.
  - ヒント: select でソートを行う. 必要な項目のみ選択. Tableタグ利用.
  - UTF-8のデータベース mts-u.sqlite



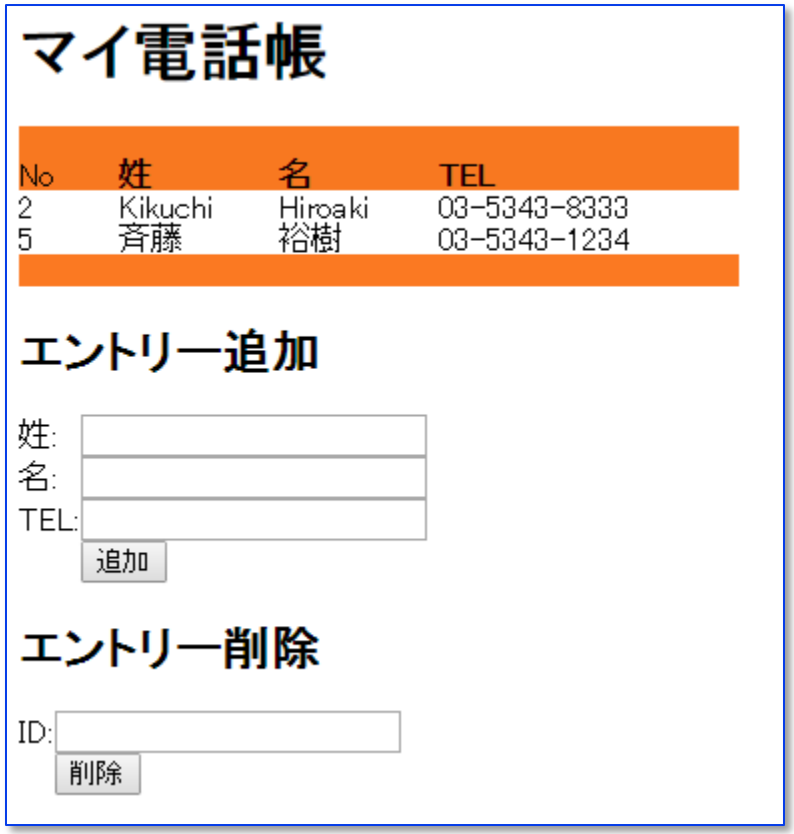
登りたい山リスト

No	山名	標高
1	奥穂岳	1685
2	八ヶ岳	1409
3	燕岳	1313
4	谷川岳	1227
5	丹沢	1201
6	天城山	756
7	那須岳	527
8	駒ヶ岳	216

2. データベースの更新

- 「マイ電話帳」
  - 姓(lastname), 名(firstname), TEL(phone)の列から成る表 phonebook.sqlite

```
CREATE TABLE users
 (id integer primary key
autoincrement,
firstname text,
lastname text,
phone text)
```



全ての行を表示

- phonebook-list.php

```
<html><head>
<title> phonebook</title> <meta charset="UTF-8"></head>
<body>
<h1> マイ電話帳 </h1>
<table border=0 cellpadding=0 cellspacing=0>
<tr bgcolor=#f87820>
<td width=50><br>No</td>
<td width=80><br><b>姓 </b></td>
<td width=80><br><b>名</b></td>
<td width=150><br><b>TEL</b></td>
<?php
$db = new PDO("sqlite:phonebook.sqlite");
$result=$db->query("SELECT * FROM users");
for($i = 0; $row=$result->fetch(); ++$i ){
        echo "<tr valign=center>";
        echo "<td >". $row['id']. "</td>";
        echo "<td >". $row['lastname']. "</td>";
        echo "<td >". $row['firstname']. "</td>";
        echo "<td >". $row['phone']. "</td></tr>";
        }
?>

<tr> <td bgcolor=#fb7922 colspan=6> </td> </tr>
</table>

</body></html>
```

行の追加

- queryメソッドでINSERT文実行

```
$db->query( "INSERT INTO テーブル
                    VALUES（値）");
```

  - » $dbはデータベースを含む_____（new PDO)
  - » 返し値はない．エラーのない様に値を用意．

❑例）

  - » $firstname=$_GET['firstname'];
  - » $db = new PDO("sqlite:phonebook.sqlite");
  - » $db->query("INSERT INTO users VALUES(null, '$firstname',null,null)");

FORMからのデータ受け取り

- phonebook-add.html

```
<html> <head><title>
phonebook</title> </head> <body>
<h2>エントリー追加</h2>
<form action=phonebook-add.php
method=get>
<table >
 <tr><td>姓:</td><td><input
type=text name=lastname>
</td></tr>
 <tr><td>名:</td><td> <input
type=text name=firstname>
</td></tr>
<tr><td>TEL:</td><td> <input
type=text name= phone> </td></tr>
 <tr><td> </td><td><input type=
submit value= "追加"></td></tr>
</table>
</form>
```

- phonebook-add.php

```
<?php
if(isset($_GET['firstname']))
$firstname=$_GET['firstname'];
if(isset($_GET['lastname']))
$lastname=$_GET['lastname'];
if(isset($_GET['phone']))
$phone=$_GET['phone'];

$db = new
PDO("sqlite:phonebook.sqlite");
if(isset($firstname))          {
    $db->query("INSERT INTO
        users VALUES(null,
        '$firstname','$lastname','$ph
one')");
}
```

単一のphpファイルにまとめてしまう

FORMからのデータ受け取り (2/2)

- phonebook-add.html

```
<html> <head><title>
phonebook</title> </head> <body>
<h2>エントリー追加</h2>
<form action=phonebook-add.php
method=get>
<table >
 <tr><td>姓:</td><td><input
type=text name=lastname>
</td></tr>
 <tr><td>名:</td><td> <input
type=text name=firstname>
</td></tr>
<tr><td>TEL:</td><td> <input
type=text name= phone> </td></tr>
 <tr><td> </td><td><input type=
submit value= "追加"></td></tr>
</table>
</form>
```

- phonebook-add.php

```
<?php
if(isset($_GET['firstname']))
$firstname=$_GET['firstname'];
if(isset($_GET['lastname']))
$lastname=$_GET['lastname'];
if(isset($_GET['phone']))
$phone=$_GET['phone'];


$db = new
PDO("sqlite:phonebook.sqlite");
if(isset($firstname))            {
    $db->query("INSERT INTO
          users VALUES(null,
          '$firstname','$lastname','$ph
one')");
}
```

行の削除

- queryメソッドでDELETE文実行

```
$db->query( "DELETE FROM テーブル
                    WHERE 条件");
```

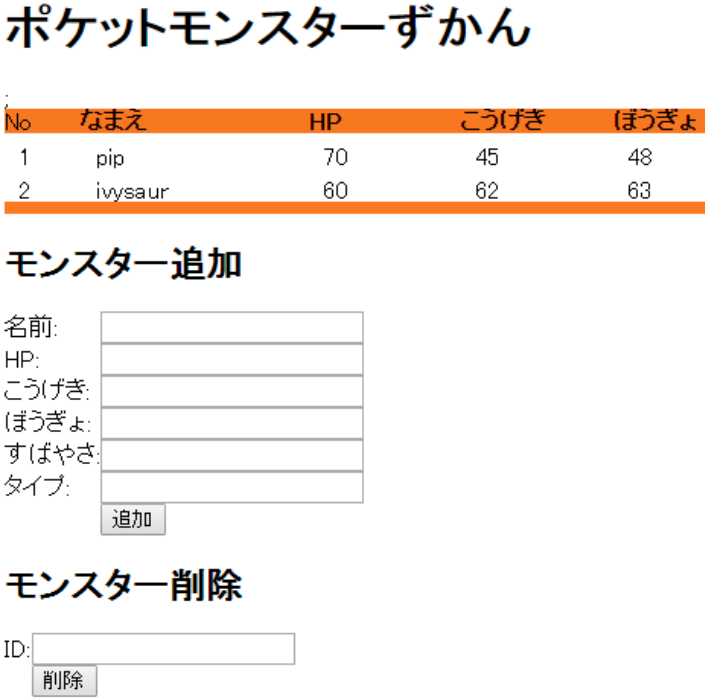
  - » 条件は，id=2 などの行を一意に決める式
  - » 返し値はない.

- ❑例）
  - » $id=$_GET['id'];
  - » $db = new PDO("sqlite:phonebook.sqlite");
  - » $db->query("DELETE FROM users WHERE id='$id' ");

演習2

- phonebook.phpを参考にして，ポケモン図鑑 pokemon.php を作れ.
  - pokemon.sqlite

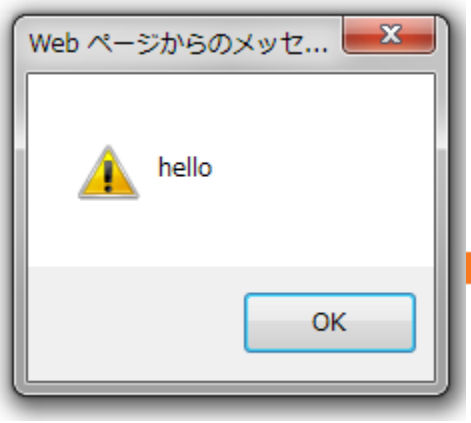
```
CREATE TABLE monsters (
id integer primary key
autoincrement,
name text,
hp integer,
offense integer,
defense integer,
speed integer,
type text);
```



  - 好きなポケモンを5匹追加せよ.
  - http://www.pokemon.co.jp/zukan/

3. セキュリティの考慮

- 演習2のモンスターの名前に次を入力して何が起きるか観察せよ.
  1. <h1> ピカチュー </h1>
  2. <script> alert('Hello') </script>
  3. <body bgcolor=black>

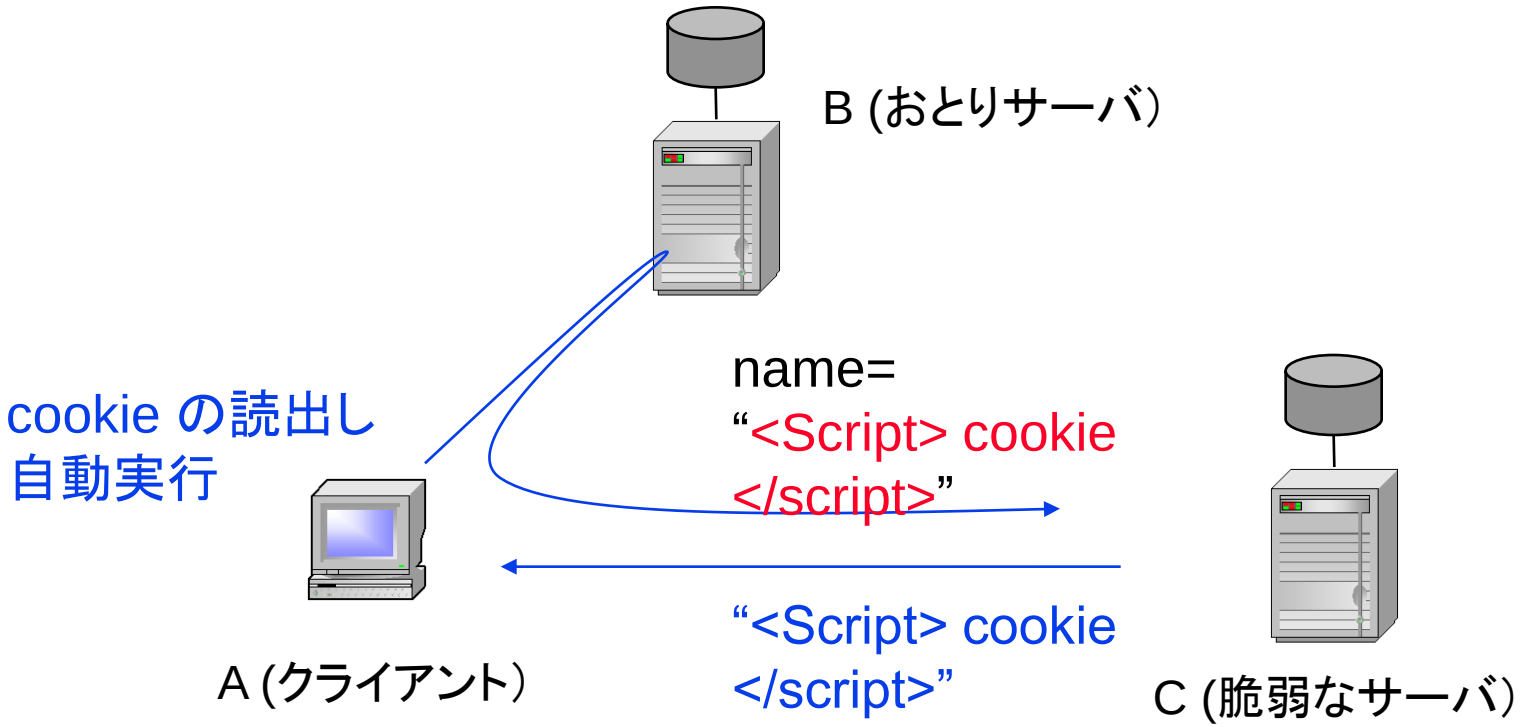


ポケットモンスターずかん

No	なまえ	HP	こうげき	ぼうぎ		
1	pip	70	45	48		
2	ivysaur	60	62	63		
5	ピカチュー	60	30	60		
9	ピカチュー	10	10	10	10	10

クロスサイトスクリプティング

- Cross Site Scripting (　　)



SQLインジェクション

- データベースへの攻撃
  - 意図しないSQL文の注入(　　)
  - 例）演習2のエントリーの削除（注意）
    DELETE　FROM monsters WHERE
    id = *$id*
  - 入力＝「3 OR 1=1」
    DELETE　FROM monsters WHERE
    id = 3 OR 1= 1 ;
  - 全行を削除する. (全データを抽出すると情報漏えい）

対策

- サニタイジング（衛生化）

  $変換後変数 = **htmlspecialchars**($入力変数);

  » 入力変数に含まれるタグをHTMLの特殊文字記号に置き換える.

  » < = <, > = >, &=&, "=" など.

  ❑例)

  <h1>ピカチュー</h1>

  <h1> ピカチュー </h1>

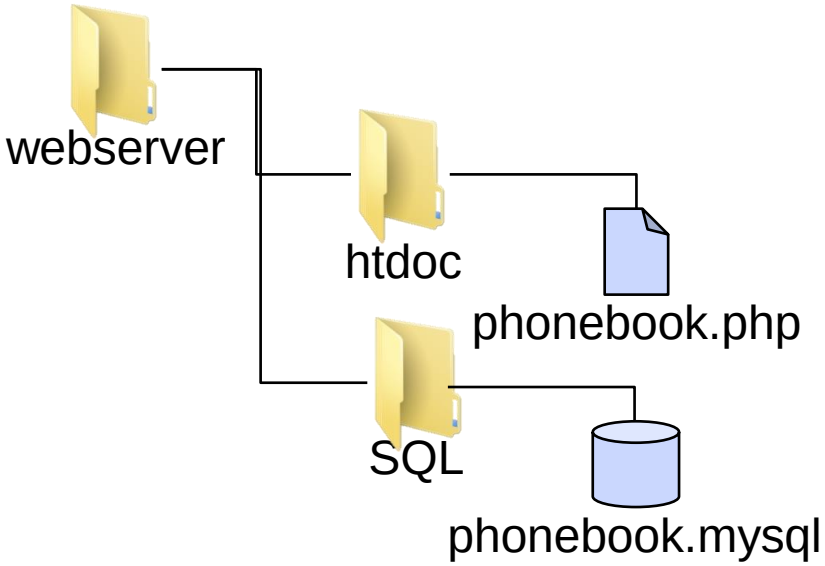
  （見た目は同じ）

その他の対策（参考）

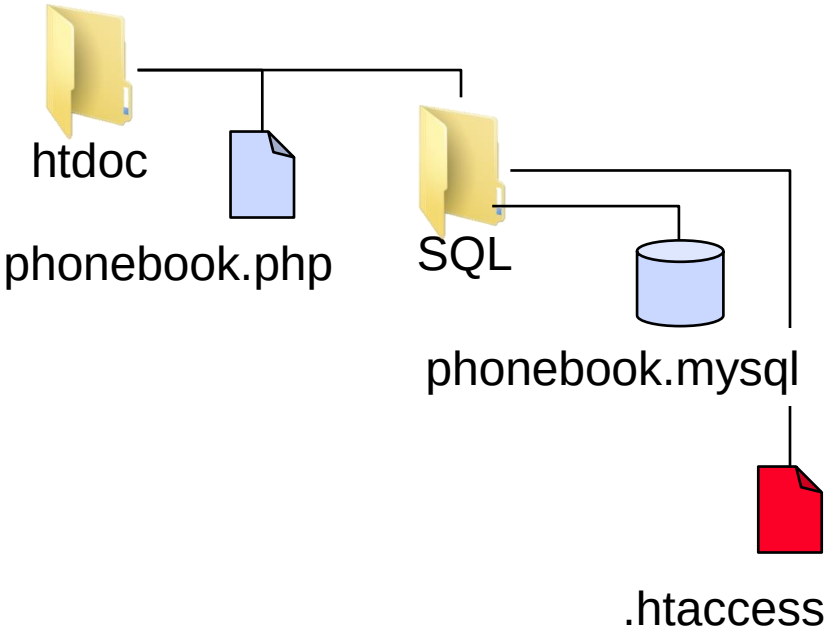
- サニタイジングだけでは完全にSQLインジェクションを防止できない.
  - タグや特殊記号を含まない命令
- 整数化
  - $intid = round($id); 文字列を整数のみに（切り捨て）
- 正規表現
  - preg_match("/[0-9]+/", $入力変数)
    入力が数字のみかどうかを判定する
- Prepared Statement
  - $ps = $db->prepare("select * from tb where id=:A");
  - $ps->bindParam(":A", $a);
  - $ps->execute();
    :Aの位置に$aの整数のみが代入.

対策

- 1. SQL DBを外部のフォルダーに.



- 2. SQL　DBを直下の別フォルダーに.



- 欠点: ファイルの管理

アクセス制御ファイル

- .htaccess（ドットに注意）

```
Order deny,allow
Deny from all
```

  - deny（　　），allow（　　）
  - order は，優先順序を指定.
  - このフォルダでは全てのウェブからのアクセスを禁じる（sqliteコマンドは別）

演習3

- 演習2で作成したpokemon.php を，次の2点を考慮した安全な pokemon-secure.php にせよ.
  - 検査方法

    (1) エントリー追加で名前「 <h1> ピカチュー </h1>」を持つ行を追加し，H1の大きさで表示されないこと.

    (2) データベース本体を http://localhost/pokemon.sqlite で外部からアクセスできないこと

まとめ

- PDOはオブジェクト指向を用いた(　　)のSQLアクセスライブラリ. (　　)により任意のSQL文を実行する.
- PDOから行を追加するには，queryメソッドにより，SQLの(　　)を実行する.
- データベースを操作するコマンドにタグが入っていると(　　)の攻撃を受ける. 防止するには，タグをHTMLの特殊文字に置き換える(　　)がある.