PHP
「超」入門

DBの一歩進んだ利用方法

今までのDB操作するプログラム

DB利用時、毎回下記の手順を・・・

1. データベースに接続
2. SQLを生成
3. prepareを実行
4. 条件の値を設定
5. SQLを実行
6. 検索結果を読み込む

今までのDB操作するプログラム

長い！メンドイ！間違える！

今までのDB操作するプログラム

同じ事を何度も何度も・・・

こういう時ってプログラミングでは・・・？

関数？クラス？

今までのDB操作するプログラム

もっと具体的方法あるよ

今までのDB操作するプログラム

DAO

（略語・・・　顔文字か！？）

※今話題のブロックチェーン絡みのDAOとは別物です

DAOとは

（DataAccessObject）

データへのアクセス担当者を作る

今までのDB操作するプログラム

オブジェクト指向では
機能ごとにクラス化が当たり前

▼

DAOは定番のパターン

これで煩わしさから解放されます

DAOパターン

DAOを作る

手順を追って説明

1．DAOのクラスを作成

・DAOとなるクラスを新規作成
・例として「DBManager.php」というファイルとします

2. DBManagerクラスを定義

・DBManager.php内にクラスを定義する

```
<?php
class DBManager {


}
?>
```

3. DB接続の関数を作る

・DBManager内にDB接続をする関数を作る（private）

```
<?php
class DBManager {
      //接続のメソッド
      private  function  dbConnect(){
         $pdo = new PDO('mysql:host=localhost;dbname=webdb;charset=utf8',
                              'webuser', 'abccsd2');
          return $pdo;
      }
}
?>
```

4. やりたいこと毎の関数を作る

・取得したい情報毎、書き込む情報毎で関数を作る
（取得の関数の場合、戻り値でfetchAllした結果を返す）

```
//user_tblをID指定で取得するメソッド
public function getUserTblById($getid){
        $pdo = $this->dbConnect();    //ここで先ほどの接続関数を利用
        $sql = "SELECT * FROM user_tbl WHERE id = ?";
        $ps = $pdo->prepare($sql);
        $ps->bindValue(1, $getid, PDO::PARAM_INT);
        $ps->execute();
        $searchArray = $ps->fetchAll();
        return $searchArray;
}
```

4. やりたいこと毎の関数を作る

・新規追加のメソッドの場合

```
//user_tblに新規登録するメソッド
public function insertUserTbl($id, $pass, $username, $usermail, $address){
    $pdo = $this->dbConnect();
    $sql = "INSERT INTO user_tbl(id, pass, username, usermail, address)
                VALUES (?,?,?,?,?)";
    $ps = $pdo->prepare($sql);
    $ps->bindValue(1, $id, PDO::PARAM_INT);
    $ps->bindValue(2, $pass, PDO::PARAM_STR);
    $ps->bindValue(3, $username, PDO::PARAM_STR);
    $ps->bindValue(4, $usermail, PDO::PARAM_STR);
    $ps->bindValue(5, $address, PDO::PARAM_STR);
    $ps->execute();
}
```

5. 利用する画面でクラス読込

・DBのデータを利用したい画面で、DBManagerクラスを読み込む

```
<?php
require 'DBManager.php';


?>
```

6. クラスをnewする

・DBManagerを使うので、newする

```
<?php
require 'DBManager.php';
$dbmng = new DBManager();

?>
```

7.関数を呼び出す

・必要な操作ができる関数を呼び出す

```
<?php
require 'DBManager.php';
$dbmng = new DBManager();

$searchArray = $dbmng->getUserTblById($_POST['id']);
?>
```

8. 取得した情報を利用する

・検索の場合などは、検索結果の配列が返るので、それを利用

```
require 'DBManager.php';
$dbmng = new DBManager();
$searchArray = $dbmng->getUserTblById($_POST['id']);

foreach($searchArray as $row){
        echo "ユーザー名:$row[username]";
}
```

DAOを利用する場面

・DBを利用する場面があれば、すべての箇所でDAOを利用する

```php
<?php
require 'DBManager.php';
$dbmng = new DBManager();
$searchArray = $dbmng->getUserTblById($_POST['id']);
?>
```

```php
<?php
require 'DBManager.php';
$dbmng = new DBManager();
$dbmng->insertUserTbl($_POST['id'], $_POST['ps'], $_POST['nm'], $_POST['ml'], $_POST['ad']);
?>
```

DAOに実装するメソッド

- DAOのメソッドはいろんな箇所で使われる
- 「何ができるメソッドなのか」が分かる名前に！
- 「動詞」「取得できるデータ」「条件」を組み合わせて

例）getUsertblByID（取得_Usertbl_IDで）
　　insertTweetTbl（新規追加_TweetTbl）
　　getTweetTblByKeyword（取得_TweetTbl_キーワードで）

DAOを利用するメリット

- DBの接続先が変わった場合、
  DAOクラスだけを書き換えれば済む
- テーブル構成が変わったときに、
  DAOクラスだけを修正すれば済む
- 毎回DB接続したり、SQLを生成したり、
  バインドしたりということをしなくて済む

デザインパターン

- オブジェクト指向にてクラスを利用した
  よく考えられたパターン→デザインパターン
- DAOは「Facade（窓口）パターン」の形式

サンプルソース

- 配付資料にDAOクラスや利用側のソースを配置
- 内容を確認してみましょう

今後のクラス設計

- DBにアクセスする機能以外にも
  複数画面から使う可能性がある機能はあるはず
  （ログイン管理機能、画面表示機能などなど）
- 積極的にクラス化していきましょう

- **補足**
  たくさんの機能をクラス化した場合、クラスとクラスが絡み合って、
  requireの際に同じファイルを何回も読み込んでしまうエラーが発生
  することがあります。
  今後はファイルの読み込みは「 require_once 」という一回のみ
  読み込みを利用しましょう

演習にチャレンジ！

TRAINING

演習4

DAOの利用

PHP復習３

- XAMPPの中、htdocsのWebフォルダ内に
- 「lesson4」フォルダを作成
- 以降の演習はこのフォルダに保存

user_tblを確認

- 事前に作成していない場合は下記の様に

名前	データ型	長さ/値	デフォルト値	照合順序	属性	NULL	インデックス
id Pick from Central Columns	INT		なし				PRIMARY PRIMARY
pass Pick from Central Columns	VARCHAR	50	なし				---
username Pick from Central Columns	VARCHAR	50	なし				---
usermail Pick from Central Columns	VARCHAR	100	なし				---
address Pick from Central Columns	VARCHAR	100	なし				---

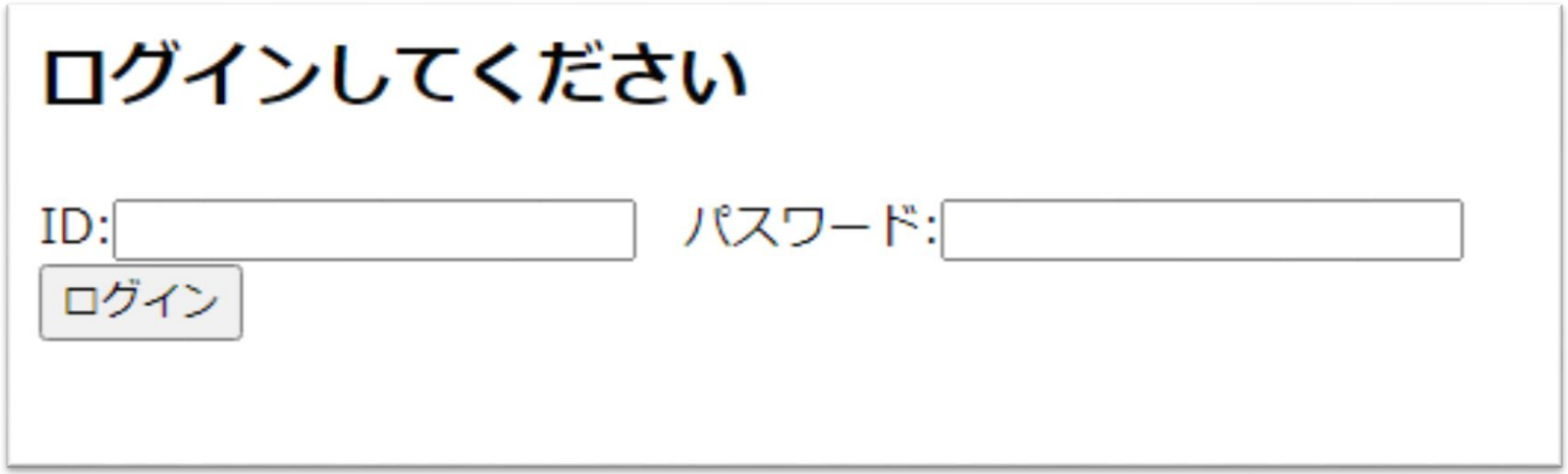
PHP「超」入門

レッスン４－１

DAOのクラスを用意する（DBManager.php）
クラス名：DBManager
下記の様なメソッドを用意する
・DBに接続し、PDOをリターンするメソッド（プライベート）
・IDとパスワードを渡すと、user_tbl検索した結果を返すメソッド

レッスン４－１

下記の様な入力画面を作成する（lesson4_1_input.php）



レッスン４－１

下記の様な出力画面を作成する（lesson4_1_output.php）
この画面内ではDAOのクラスを利用すること
正しいIDとパスワードを入れた場合（user_tblに登録されている内容）

検索結果の１件目のusernameを表示

誤ったIDとパスワードを入れた場合

PHP「超」入門

レッスン４－２

先ほどのDAOのクラスを修正する（DBManager.php）

クラス名：DBManager

下記の様なメソッドを追加する

・地名を渡すと、その地名が「address」に含まれる情報を
　user_tblから検索し、結果を返す

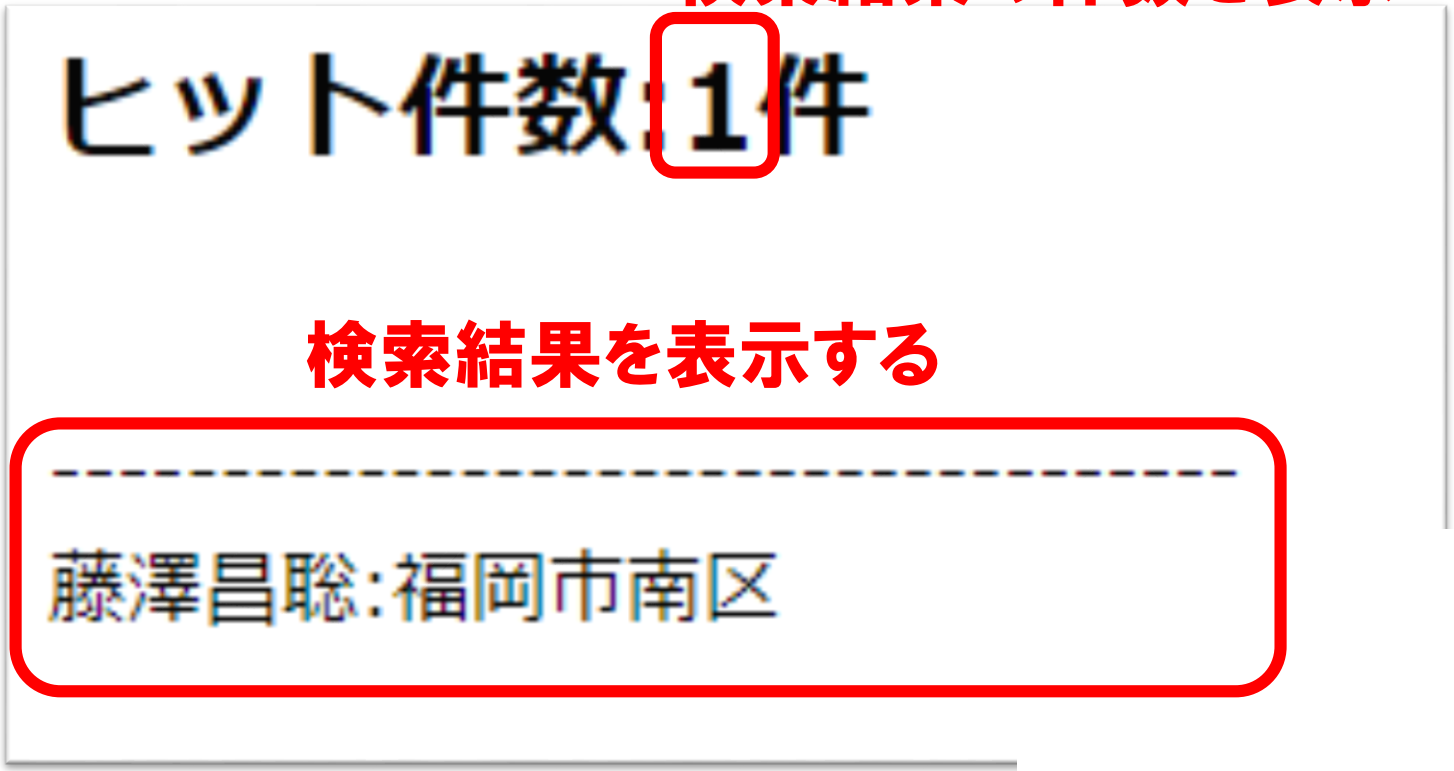
レッスン４－２

下記の様な入力画面を作成する（lesson4_2_input.php）

PHP「超」入門 レッスン４－２

下記の様な出力画面を作成する（lesson4_2_output.php）
この画面内ではDAOのクラスを利用すること

検索結果の件数を表示



自信がある方はこの演習にチャレンジ（成績評価で加点対象）

lesson3（DBの復習）で作成した つぶやき投稿の機能を DAOを使う方式で作り直し

ここまでに作ったDBManagerに追加の機能で作成

提出はTeamsのフォームで

・Teamsにフォームを発行しています。
指示に従ってファイルを提出してください。
（それぞれフォームを用意）