

PHP 04



G's ACADEMY
FUKUOKA



本日の内容

講義 + 作業 : 2.5 h程度
演習 : 1.5 h程度

アジェンダ

- セッション機能
- ログイン処理
- ログアウト処理
- 課題発表→チュータリング(演習)タイム

授業のルール

- 授業中は常にエディタを起動！
- 隣の人と相談するときは周りの迷惑にならない大きさで.
- 周りで困ってそうな人がいたらおしえてあげましょう！
- まずは**打ち間違い**を疑おう！

{ } ' " ; など

- 書いたら保存しよう！

command + s

ctrl + s

前回の課題

ユーザ管理機能の作成

■ユーザ管理テーブル（←必ず作成，DBはこれまでのものを使用）

- ・ テーブル名： user_table

■カラム名

- ・ id (int 12 , PRIMARY, AutoIncrement)
- ・ name (varChar 64) ← ユーザ名
- ・ lid (varChar 128) ← ログインID
- ・ lpw (varChar 64) ← パスワード
- ・ kanri_flg (int 1) ← 一般：0, 管理者：1
- ・ life_flg (int 1) ← アクティブ：0, 非アクティブ：1

ユーザ管理機能の作成

■下記ファイル（処理）を作成

- ・ user_index.php （登録処理）
- ・ user_select.php （表示処理）
- ・ user_detail.php （更新画面）
- ・ user_update.php （更新処理）
- ・ user_delete.php （削除処理）

■ブックマーク機能とユーザ管理機能はそれぞれ独立でOK！

準備

- MAMPの起動確認
- <http://localhost/>のアクセス確認
- サンプルフォルダを「htdocs」フォルダに入れる

webの仕組み（復習）

URL

■URLとは

- ・ web上にある情報の場所を指し示す住所のようなもの.
- ・ Uniform Resource Locatorの略.

■例

サーバ名

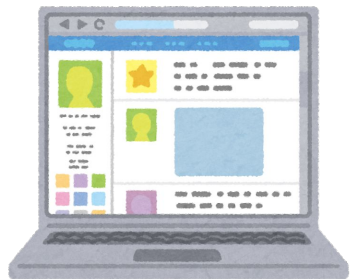
ファイル名

<http://www.〇〇〇〇.jp/△△△△/index.html>

ディレクトリ名

サーバサイド言語の仕組み

※ 言語によらず、ファイル（プログラム）はサーバ上に存在



送られてきたhtmlを実行

- ・ こういう情報がほしい
- ・ こういう処理をしたい
- ・ 例：index.phpにアクセス

http通信

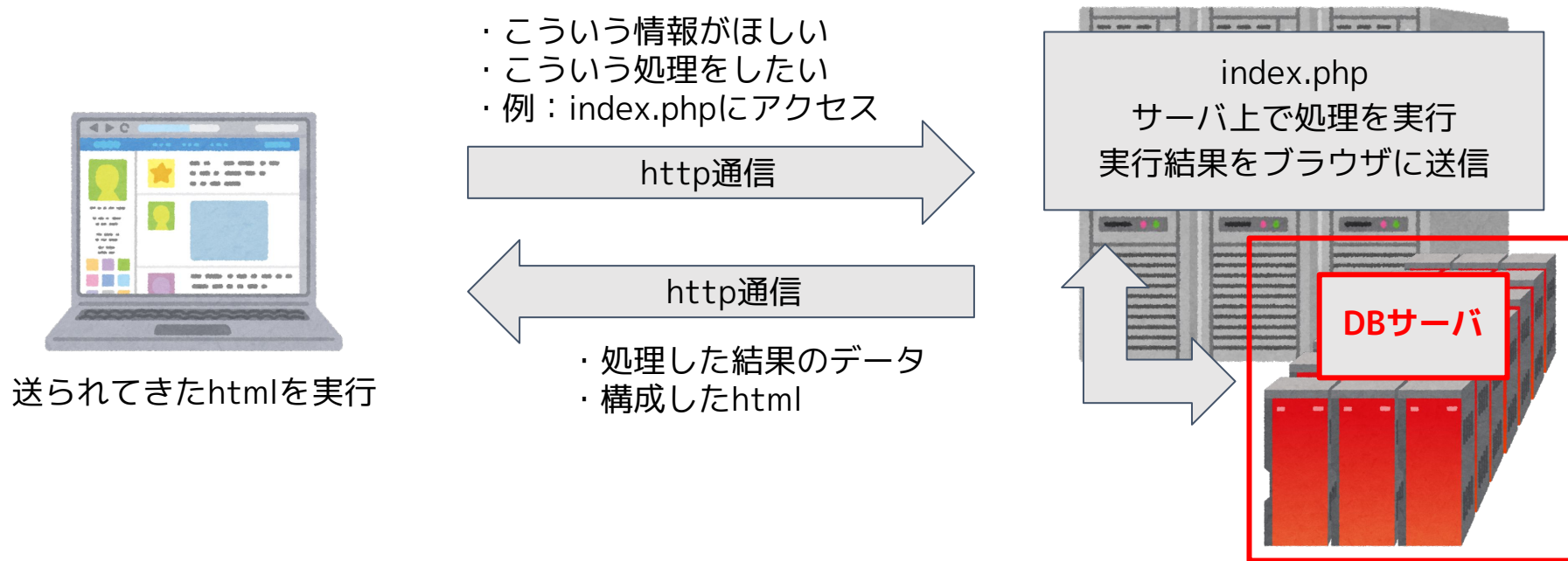
http通信

- ・ 処理した結果のデータ
- ・ 構成したhtml

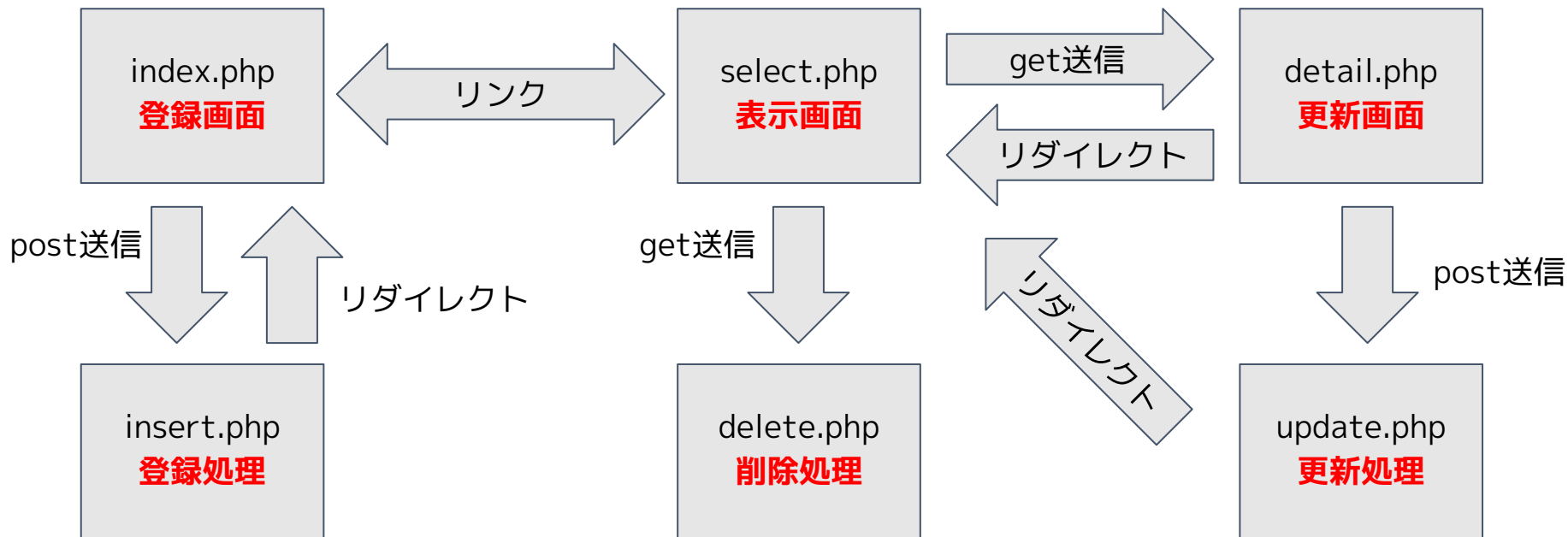


DBの動き方

サーバ上のプログラムがDBにアクセスして処理を実行！



タスク管理アプリの全体像



session

sessionとは

session_id.php

■何か？？

- ・ サーバに変数などを保存できる仕組み.

. . . 以上である！

■補足

- ・ サーバ自体に変数を定義する.
- ・ サーバ上にあるどのファイルからでも値を取り出せる！

使うときの流れ

session_id.php

■まずはsessionを宣言

```
session_start();
```

- ・ 上記を行うことで, idが発行されてブラウザにidが保存される.
- ・ sessionを使用するための「鍵」的なイメージ.
- ・ **使用するファイルでは必ず最初に記述する！**

■idの確認

- ①PHPファイル上でsession_id();で取得可能.
- ②ブラウザで「検証→Application→Cookies→localhost」

■session_regenerate_id();

- ・ sessionのidがバレるとヤバイ. . . !
- ・ 他の人にsessionの中身をいじられてしまう. . . !
- ・ session_regenerate_id();を使用するとidを再生成できる.
- ・ （保存されているデータ自体は変更なし）

■使い所

- ・ ログインしたらid発行してログイン情報を管理.
- ・ ページ移動したタイミングで再生成！

■session_regenerate_id();の例

```
<?php
session_start();           // セッション開始
$old_session_id = session_id(); // idの取得
session_regenerate_id(true); // id再生成&旧idを破棄
$new_session_id = session_id(); // 新idの取得
echo '<p>旧id' . $old_session_id . '</p>';
echo '<p>新id' . $new_session_id . '</p>';
?>
```

idの管理

```
session_regenerate_id.php
```

■練習

- ・ idを発行して確認しよう！
- ・ 再生成して旧idと新idを表示しよう！

session変数を使う！

session01.php

■サーバに変数を保存する！

`$_SESSION['変数名']`で宣言.

【例】

```
<?php
```

```
session_start();
```

```
$_SESSION['num'] = 100; // session変数の宣言
```

```
echo $_SESSION['num'];
```

```
?>
```

session変数を使う！

session02.php

- サーバに保存されている変数を取り出す.

```
<?php
```

```
session_start();
```

```
$_SESSION['num'] += 1;    // session変数を取り出して+1する
```

```
echo $_SESSION['num']; // 結果を出力
```

```
?>
```

session変数を使う！

session02.php

■練習

- ・ session01.phpでsession変数を定義しよう！
- ・ session02.phpで定義した変数を呼び出して出力しよう！

sessionの終了

■session変数の削除

```
unset($_SESSION[key]); // 該当するものを削除
```

■session情報の全削除（ログアウト処理など）

```
$_SESSION = array(); // 情報を全て削除
```

```
// ↓ブラウザに保存されている情報をクリア
```

```
setcookie(session_name(), "", time() - 3600, '/');
```

```
session_destroy(); // sessionを破壊
```

ログイン&ログアウト

login&logoutの全体像

■必要なファイル

- ・ login.php ログイン情報（id, pass）を入力して送信
- ・ login_act.php 送信されたデータを受け取り，DB関連の処理
- ・ logout.php セッション，ログイン情報の破棄

login処理の流れ

■ログイン

- ① ログインフォーム情報を入力して送信 (login.php)
- ② 送信されたデータを受け取る (login_act.php)
- ③ 受け取ったデータがDBにあるかどうかチェック (login_act.php)

■成功時 (DBにユーザのデータが存在した場合)

- ① DBにログイン情報があればセッション変数に格納 (login_act.php)
- ② セッション変数にログイン情報を保持してselect.phpに移動

■失敗時 (DBにユーザのデータが存在しなかった場合)

- ① DBにログイン情報がなければlogin.phpに戻る (ログイン失敗)

login処理の流れ

login.php

■ログイン情報の送信（前回までと同じ要領）

```
<form method="post" action="login_act.php">
  <div class="form-group">
    <label for="lid">LoginID</label>
    <input type="text" id="lid" name="lid">
  </div>
  ...
```

login処理の流れ

login_act.php

■セッションの開始→情報の受け取り

<?php

```
session_start();           // セッションの開始
include('functions.php');   // 関数ファイル読み込み
$pdo = db_conn();          // DB接続
$lid = $_POST['lid'];       // データ受け取り→変数に入れる
$lpw = $_POST['lpw'];
...
```

■DBにデータがあるかどうか検索

```
$sql = 'SELECT * FROM user_table WHERE lid=:lid AND lpw=:lpw  
AND life_flg=0';
```

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->bindValue(':lid', $lid, PDO::PARAM_STR);
```

```
$stmt->bindValue(':lpw', $lpw, PDO::PARAM_STR);
```

```
$res = $stmt->execute();
```

【復習】SQL構文：SELECT文のオプション

■演算子の使用

SELECT * FROM php02_table **WHERE** id = 1; -- 「==」ではない！

SELECT * FROM php02_table **WHERE** id >= 1;

SELECT * FROM php02_table **WHERE** id >= 1 **AND** id <= 3;

■あいまい検索

SELECT * FROM php02_table **WHERE** email **LIKE** 'gs%';

SELECT * FROM php02_table **WHERE** email **LIKE** '%gmail.com';

SELECT * FROM php02_table **WHERE** email **LIKE** '%@%';

■DBにデータがあればセッション変数に格納

```
$val = $stmt->fetch();           //1レコードだけ取得
if ($val['id'] != '') {
    $_SESSION = array();
    $_SESSION['chk_ssid'] = session_id();
    $_SESSION['kanri_flg'] = $val['kanri_flg'];
    $_SESSION['name']     = $val['name'];
    header('Location: select.php');
} ...
```

■ログインしているかどうかのチェック→毎回id再生成

```
function chk_ssid () {  
    // 失敗時はログイン画面に戻る（セッションidがないor一致しない）  
    if (!isset($_SESSION['chk_ssid']) || $_SESSION['chk_ssid']!=session_id()) {  
        header('Location: login.php');  
    } else {  
        session_regenerate_id(true);           // セッションidの再生成  
        $_SESSION['chk_ssid'] = session_id();   // セッション変数に格納  
    }  
}
```


login状態のチェック

■各ページ読み込み時にログインチェック

<?php

```
session_start();           // セッションの開始
include('functions.php');   // 関数ファイル読み込み
chk_ssuid();               // idチェック関数の実行
...
```

■下記ファイルで実施

index.php, select.php, detail.php

各ページでログインしていない状態だとログインページへ移動する.

■セッションの破棄→ログイン画面へ移動

<?php

```
session_start();           // セッションの開始
$_SESSION = array();       // セッション変数を空にする
if (isset($_COOKIE[session_name()])) {
    setcookie(session_name(), '', time()-42000, '/');
}                           // クッキーの保持期限を過去にする
session_destroy();          // セッションの破棄
header('Location: login.php'); // ログインページへ移動
```

課題

【課題】これまでのアプリにログイン機能を追加

■下記の処理を作成（授業と同様！）

- ・ login.php （ログイン画面）
- ・ login_act.php （ログイン処理）
- ・ logout.php （ログアウト処理）

■ユーザの状況によって画面を分ける

- ①ログインしていないユーザ
- ②一般ユーザ
- ③管理者ユーザ

ログイン&ログアウト処理の作成

■授業で扱った内容と同じ！

Login ログインページ 一覧ページ

LoginID

Enter LoginID

Pass

Submit

ログインしていないユーザの画面

■ ログインしていないユーザは情報を見るだけ！

todo一覧 ログインページ 一覧ページ

| | |
|-------------------|--------|
| 2019-02-20-高級な焼き肉 | detail |
| 2019-02-13-水炊き | detail |
| 2019-02-14-とりかわ | detail |
| 2019-02-13-ハイボール | detail |

select_nologin.php
(新しく作成)

todo詳細 ログイン 一覧ページ

Task

高級な焼き肉

Deadline

2019/02/20

Comment

奢って！！！！

detail_nologin.php
(新しく作成)

編集できない

削除できない

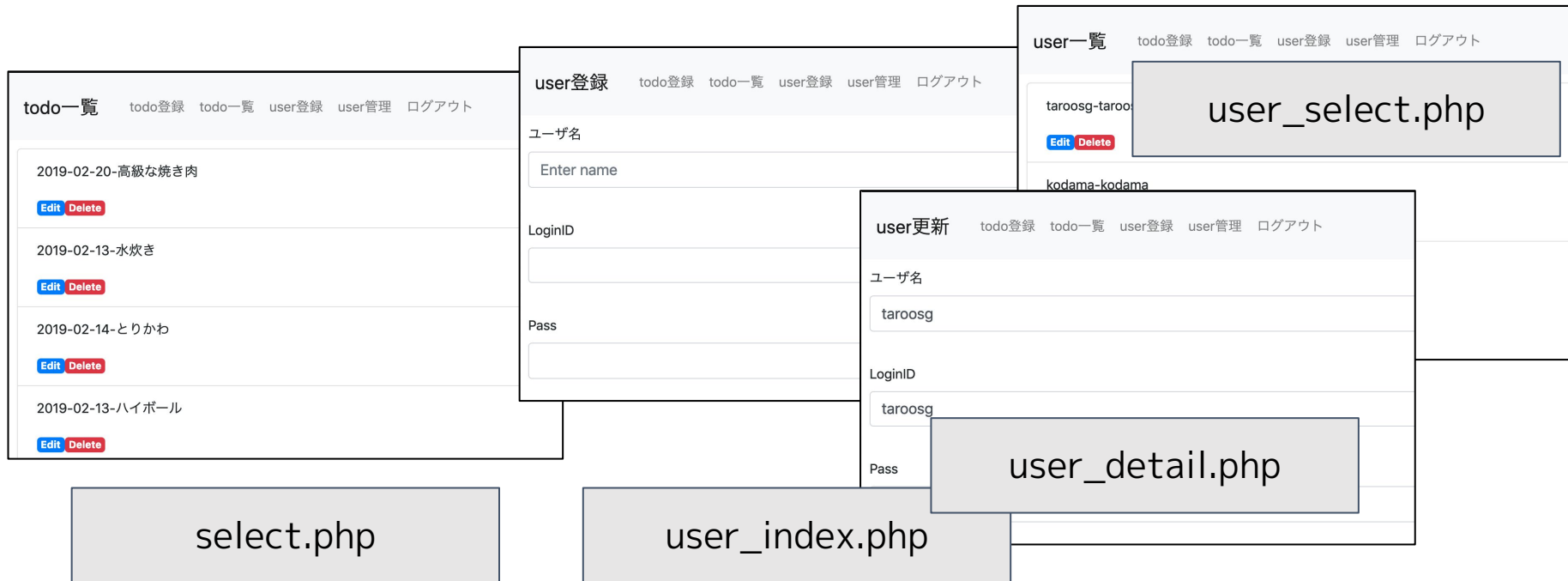
一般ユーザ（ログイン中）の画面

■これまでと同様だが，ログインしないと表示しないように！



管理者ユーザ（ログイン中）の画面

■ 前回課題2で作成したユーザ管理ファイルへのリンクを表示！



提出は次週木曜日「23:59:59」まで！！

チュータリングタイム

17:00までは一人でもくもく
後半は近くのメンバーで教え合おう！