

# PHP入門

①フォームのデータの送信と受信



# PHPの基本の基本

(2) PHPが処理を行って

項目名

担当者  
--選択してください--

期限  
年 / 月 / 日

☐ 完了

PHP

(1) フォームから送信したデータ（値）を

(3) HTMLとして出力します

項目名	担当者	登録日	期限日	完了日	操作
レバニラ炒め作る	がみ	2019-12-05	2019-12-26		<input type="button" value="完了"/> <input type="button" value="更新"/> <input type="button" value="削除"/>
初詣に行く	さかがみ	2020-01-04	2020-01-01	2020-01-04	<input type="button" value="完了"/> <input type="button" value="更新"/> <input type="button" value="削除"/>
東京オリンピック	さかがみ	2019-12-05	2020-07-24		<input type="button" value="完了"/> <input type="button" value="更新"/> <input type="button" value="削除"/>

# HTMLのフォーム

データを入力し、送信するための仕組みです。



項目名

担当者

期限

☐ 完了

<form>タグでくくります。

<form>

</form>

# フォームの部品いろいろ (1)

- テキストボックス

```
<input type="text">  
<input type="password">  
<textarea></textarea>
```

- セレクトボックス

```
<select>  
  <option>りんご</option>  
  <option>バナナ</option>  
  <option>みかん</option>  
</select>
```

## フォームの部品いろいろ (2)

- ラジオボタン  
`<input type="radio">`
- チェックボックス  
`<input type="checkbox">`
- ボタン  
`<input type="submit">`  
`<input type="reset">`  
`<button></button>`

# フォームをつくってみよう (1)

```
<form>
  <p><input type="text"></p>
  <p><input type="password"></p>
  <p><input type="radio"></p>
  <p>
    <select>
      <option>大阪</option>
      <option>兵庫</option>
      <option>奈良</option>
    </select>
  </p>
  <p><input type="checkbox"></p>
  <p>
    <input type="submit">
    <input type="reset">
  </p>
</form>
```

# フォームをつくってみよう (2)

- 部品には「name」属性をつけます。

```
<form>
  <p><input type="text" name="mane"></p>
  <p><input type="radio" name="gender"></p>
  <p>
    <select name="pref">
      <option>大阪</option>
      <option>兵庫</option>
      <option>奈良</option>
    </select>
  </p>
  <p><input type="checkbox" name="is_admin"></p>
  <p>
    <input type="submit">
    <input type="reset">
  </p>
</form>
```

# フォームをつくってみよう (3)

- 部品に「value」属性を指定することで、値を設定することができます。

```
<input type="radio" name="gender" value="1">
<select name="pref">
  <option value="大阪">大阪</option>
  <option value="兵庫">兵庫</option>
  <option value="奈良">奈良</option>
</select>
<input type="checkbox" name="is_admin" value="0">
```

- 複数のラジオボタンに同じ「name」属性をつけることで、同じグループの中の  
一つだけ選択できるようになります。

```
<input type="radio" name="gender" value="1">
<input type="radio" name="gender" value="2">
<input type="radio" name="gender" value="9">
```



# フォームをつくってみよう (4)

- ラジオボタンとチェックボックスは、「checked」を設定することで、初期状態でチェックを入れることができます。

```
<input type="radio" name="gender" value="1" checked>  
<input type="radio" name="gender" value="2">  
<input type="radio" name="gender" value="9">
```

```
<input type="checkbox" name="is_admin" checked>
```

- セレクトボックスは、「selected」を設定することで、初期状態で選択状態にすることができます。

```
<select>  
  <option selected>大阪</option>  
  <option>兵庫</option>  
  <option>奈良</option>  
</select>
```

# フォームをつくってみよう (5)

- フォームの送信方法と送信先を指定します。
  - 送信方法（メソッド）を指定します。

```
method="post"  
method="get"
```

- 送信先のアドレスを指定します。絶対パス、相対パスで指定できます。

```
action="./display.php"  
action="https://www.sample.jp/hoge/foo/index.php"
```

```
<form method="post" action="./display.php">  
  
</form>
```

# フォームを完成しよう (1)

- 1/index.html

```
<!DOCTYPE html>
<html lang="jp">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>PHPのワーク</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
  integrity="sha384-
  Vkoo8x4CGsO3+Hh xv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
  crossorigin="anonymous">
</head>
```

※ スタイルにBootstrapを使っています。

<https://getbootstrap.com/>

# フォームを完成しよう (2)

```
<body>
  <div class="container">
    <form method="post" action="./display.php" class="my-3">
      <div class="form-group">
        <label for="name">お名前</label>
        <input type="text" name="name" id="name" class="form-control">
      </div>

      <div class="form-group">
        <label for="email">メールアドレス</label>
        <input type="text" name="email" id="email" class="form-control">
      </div>

      <div class="form-group">
        <label for="password">パスワード</label>
        <input type="password" id="password" name="password"
          class="form-control">
      </div>
    </form>
  </div>
</body>
```

# フォームを完成しよう (3)

```
<div class="form-group">
  <div class="form-check form-check-inline">
    <input type="radio" name="gender" id="gender1" value="1" class="form-check-input"
      checked>
    <label for="gender1">男性</label>
  </div>
  <div class="form-check form-check-inline">
    <input type="radio" name="gender" id="gender2" value="2" class="form-check-
      input">
    <label for="gender2">女性</label>
  </div>
  <div class="form-check form-check-inline">
    <input type="radio" name="gender" id="gender9" value="9" class="form-check-
      input">
    <label for="gender9">その他</label>
  </div>
</div>
```

# フォームを完成しよう (4)

```
<div class="form-group">
  <label for="pref">事業所所在地</label>
  <select name="pref" id="pref" class="form-control">
    <option value="大阪" selected>大阪</option>
    <option value="兵庫">兵庫</option>
    <option value="京都">京都</option>
    <option value="奈良">奈良</option>
    <option value="和歌山">和歌山</option>
  </select>
</div>
```

```
<div class="form-group">
  <label for="tel">電話番号</label>
  <input type="text" name="tel" id="tel" class="form-control">
</div>
```

# フォームを完成しよう (5)

```
<div class="form-check">
  <input type="checkbox" name="is_admin" id="is_admin" value="1"
  class="form-check-input">
  <label for="is_admin">管理者の方はチェックを入れてください。</label>
</div>
```

```
<div class="my-2">
  <input type="submit" value="送信" class="btn btn-primary">
  <input type="reset" value="リセット" class="btn btn-outline-primary">
</div>
```

```
</form>
```

```
</div>
```

```
</body>
```

```
</html>
```

# フォームを完成しよう (6) 完成！

お名前

メールアドレス

パスワード

☒ 男性 ☐ 女性 ☐ その他

事業所所在地

大阪

電話番号

☐ 管理者の方はチェックを入れてください。

送信

リセット



# フォームから送信されたデータをPHPで表示しよう (1)

method="post" で送信されたとき

**`$_POST`**

method="get" で送信されたとき

**`$_GET`**

で受け取ります。

# フォームから送信されたデータをPHPで表示しよう (2)

name="〇〇" の「〇〇」の値を取り出すときは、

<code>\$_POST['〇〇']</code>	→ method="post"のとき
<code>\$_GET['〇〇']</code>	→ method="get"のとき

とします。

例)

HTML

```
<input type="text" name="foo">
```

PHP

```
$_POST['foo']  
$_GET['foo']
```

# PHPでフォームから受け取ったデータを表示するには

- PHPのコードは、`<?php ?>`でくくります。  
値を表示するには「`echo`」を使います。

```
<?php
    echo $_POST['foo'];
?>
<?php echo $_POST['foo']; ?>
```

- 1行で書くときは、下記のように省略できます

```
<?= $_POST['memo'] ?>
```

# PHPでフォームから受け取ったデータを表示しよう (1)

- 1/display.php

```
<!DOCTYPE html>
<html lang="jp">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>PHPのワーク</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
  integrity="sha384-
  Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
  crossorigin="anonymous">
</head>
```

※ スタイルにBootstrapを使っています。

<https://getbootstrap.com/>

# PHPでフォームから受け取ったデータを表示しよう (2)

```
<body>
  <div class="container-md">

    <table class="table table-striped table-hover my-3">
      <tr>
        <th>お名前</th>
        <td><?= $_POST['name'] ?></td>
      </tr>
      <tr>
        <th>メールアドレス</th>
        <td><?= $_POST['email'] ?></td>
      </tr>
      <tr>
        <th>パスワード</th>
        <td><?= $_POST['password'] ?></td>
      </tr>
    </table>
  </div>
</body>
```

# PHPでフォームから受け取ったデータを表示しよう (3)

```
<tr>
  <th>性別</th>
  <td>
    <?php if ($_POST['gender'] == '1') : ?>
      男性
    <?php elseif ($_POST['gender'] == '2') : ?>
      女性
    <?php else : ?>
      その他
    <?php endif ?>
  </td>
</tr>
<tr>
  <th>事業所所在地</th>
  <td><?= $_POST['pref'] ?></td>
</tr>
<tr>
  <th>電話番号</th>
  <td><?= $_POST['tel'] ?></td>
</tr>
```

# PHPでフォームから受け取ったデータを表示しよう (4)

```
<tr>
<th>管理者</th>
<td>
    <?php if (isset($_POST['is_admin']) && $_POST['is_admin'] == 1) : ?>
        はい
    <?php else : ?>
        いいえ
    <?php endif ?>
</td>
</tr>
</table>

</div>

</body>

</html>
```

- ※ **isset()** は、変数が定義されているか、値が存在するか、調べます。
- 変数が定義されていて、値がある (**null**でない) ときは、**true**
  - そうでないときは、**false**
- を返却します。

## PHPでフォームから受け取ったデータを表示しよう (5) 完成！

お名前	本上まなみ
メールアドレス	ayase@sample.jp
パスワード	password
性別	女性
事業所所在地	兵庫
電話番号	070-1234-5678
管理者	はい



# PHP入門

②データベースに接続して  
データを表示しよう！

※ 「データベース入門」のワークの資料も  
見てね！



# データベースに接続してデータを表示しよう！

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000
4	順子	junko@sample.co.jp	password		女性	兵庫	000-000-0000
5	桃子	momoko@sample.co.jp	password		女性	和歌山	000-000-0000
6	司	tsukasa@sample.co.jp	password		男性	京都	000-000-0000
7	涼子	ryoko@sample.co.jp	password		女性	兵庫	000-000-0000
8	小百合	sayuri@sample.co.jp	password		女性	大阪	06-0000-0000
9	典子	noriko@sample.co.jp	password		女性	大阪	06-0000-0000
10	武蔵	musashi@sample.jp	password		女性	奈良	000-000-0000
11	小次郎	kojiro@sample.jp	password		男性	和歌山	03-0000-0000
12	ニャース	nyasu@sample.jp	password		男性	京都	000-000-0000
13	今日子	kyoko@sample.jp	password		女性	兵庫	000-000-0000
14	はるか	haruka@sample.jp	password		女性	和歌山	03-0000-0000
15	森高千里	chisato@sample.jp	password	はい	女性	奈良	000-123-4567

# PHPの基本の基本

(1) PHPがデータベース接続を行って



(2) PHPが受け取ったデータの処理を行って

(3) HTMLとして出力します

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000

# データベースのテーブルの確認

データベース名：php\_work

テーブル名：users

name	type	NN	PK	AI	Default	Check
id	INTEGER	✓	✓	✓		
name	VARCHAR(50)	✓				
email	VARCHAR(256)	✓				
password	VAECHAR(256)	✓				
is_admin	TINYINT	✓			0	
gender	TINYINT	✓			9	
pref	VARCHAR(10)	✓				
tel	VARCHAR(20)	✓				

✓ is\_admin：1のとき管理者、0のとき通常ユーザー

✓ gender：1のとき男性、2のとき女性、9のときその他

※ 詳細は、「データベース入門」のワークの資料を確認してください。

# PHPからデータベースに接続する

- SQLiteを使う場合

```
<?php
// データベースに接続
// データベースに接続するための文字列（データソースネーム）を作る
// データベースの種類がSQLiteの場合は、下記のように記載します。
$dsn = 'sqlite:../work.db';

// PDOクラスのインスタンスを作る
$dbh = new PDO($dsn);
```

- MySQLを使う場合

```
<?php
// データベースに接続するための文字列（DSN・接続文字列）
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';

// PDOクラスのインスタンスを作る
$user = 'root';
$password = ''; // XAMPPのパスワードはデフォルトで空文字です
$dbh = new PDO($dsn, $user, $password);
```

# PHPでSQLを実行する

- ここからは、データベースの種類に関わらず、共通→PDOクラスを使う利点

```
// データを表示するためのSQL文
$sql = 'select * from users';

// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文を実行
$stmt->execute();

// 実行結果をすべて取り出し、変数に代入
$list = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

# PHPでSQLを実行した結果を表示するには (1)

- SQL文を実行した結果は「**連想配列**」という形になっています。

```
$list = [  
    0 => [  
        'id'=> 1,  
        'name'=> "ちえみ",  
        'email'=> "chiemi@sample.jp",  
        'password'=> "password",  
        'is_admin'=> 0,  
        'gender'=> 2,  
        'pref'=> "大阪",  
        'tel'=> "000-123-4568"  
    ],  
    1 => [  
        'id'=> 2,  
        'name'=> "花子",  
        'email'=> "hanako@sample.jp",  
        'password'=> "password",  
        'is_admin'=> 0,  
        'gender'=> 2,  
        'pref'=> "京都",  
        'tel'=> "000-000-0000"  
    ]  
];  
  
echo $list[0]['name'];           // 「ちえみ」と表示される  
echo $list[1]['email'];         // 「hanako@sample.jp」と表示される
```

# PHPでSQLを実行した結果を表示するには (2)

- 連想配列を順番に取り出して表示するには「foreach」を使います。

`foreach ($list as $v)`

自動的に連想配列`$list`の行（要素）を最初から最後までぐるぐる回してくれます。  
1行（要素）ずつ取り出して、`$v`に代入してくれます。

例1)

PHPのコードの中で書くとき

```
foreach ($list as $v) {  
    echo $v['id'].'<br>';  
    echo $v['name'].'<br>';  
    echo $v['email'].'<br>';  
}
```

例2)

HTMLの中に混ぜて書くとき

```
<?php foreach ($list as $v) : ?>  
    <?= $v['id'] ?><br>  
    <?= $v['name'] ?><br>  
    <?= $v['email'] ?><br>  
<?php endforeach ?>
```



# PHPでSQLを実行した結果を表示しよう！ (1)

- 2/index.php

```
<?php
// データベースに接続するための文字列 (DSN・接続文字列)
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';

// PDOクラスのインスタンスを作る
$dbh = new PDO($dsn, 'root', '');

// データを表示するためのSQL文
$sql = 'select * from users';

// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文を実行
$stmt->execute();

// 実行結果をすべて取り出し、変数に代入
$list = $stmt->fetchAll(PDO::FETCH_ASSOC);
?>
```

# PHPでSQLを実行した結果を表示しよう！ (2)

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>検索結果</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
</head>

<body>
  <div class="container">
```

# PHPでSQLを実行した結果を表示しよう！ (3)

```
<table class="table table-striped table-hover my-3">
  <thead class="bg-primary text-white">
    <tr>
      <th>id</th>
      <th>お名前</th>
      <th>email</th>
      <th>パスワード</th>
      <th>管理者</th>
      <th>性別</th>
      <th>勤務地</th>
      <th>勤電話番号</th>
    </tr>
  </thead>
```

# PHPでSQLを実行した結果を表示しよう！ (4)

```
<tbody>
  <?php foreach ($list as $v) : ?>
    <tr>
      <td><?= $v['id'] ?></td>
      <td><?= $v['name'] ?></td>
      <td><?= $v['email'] ?></td>
      <td><?= $v['password'] ?></td>
      <!-- is_adminが1のときは「はい」と表示する（1でないときは何も表示しない） -->
      <td><?php if ($v['is_admin'] == 1) echo 'はい' ?></td>
      <td>
        <!-- genderが1だったら「男性」と表示 -->
        <?php if ($v['gender'] == 1) : ?>
          男性
        <!-- genderが2だったら「女性」と表示 -->
        <?php elseif ($v['gender'] == 2) : ?>
          女性
        <!-- genderが3だったら「その他」と表示 -->
        <?php else : ?>
          その他
        <?php endif ?>
      </td>
```

# PHPでSQLを実行した結果を表示しよう！ (5)

```
        <td><?= $v['pref'] ?></td>
        <td><?= $v['tel'] ?></td>
    </tr>
    <?php endforeach ?>
</tbody>
</table>
</div>
</body>

</html>
```

# PHPでSQLを実行した結果を表示しよう！ (6)

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000
4	順子	junko@sample.co.jp	password		女性	兵庫	000-000-0000
5	桃子	momoko@sample.co.jp	password		女性	和歌山	000-000-0000
6	司	tsukasa@sample.co.jp	password		男性	京都	000-000-0000
7	涼子	ryoko@sample.co.jp	password		女性	兵庫	000-000-0000
8	小百合	sayuri@sample.co.jp	password		女性	大阪	06-0000-0000
9	典子	noriko@sample.co.jp	password		女性	大阪	06-0000-0000
10	武蔵	musashi@sample.jp	password		女性	奈良	000-000-0000
11	小次郎	kojiro@sample.jp	password		男性	和歌山	03-0000-0000
12	ニャース	nyasu@sample.jp	password		男性	京都	000-000-0000
13	今日子	kyoko@sample.jp	password		女性	兵庫	000-000-0000
14	はるか	haruka@sample.jp	password		女性	和歌山	03-0000-0000
15	森高千里	chisato@sample.jp	password	はい	女性	奈良	000-123-4567

# PHPでレコードを検索して表示しよう！

事業所所在地

大阪

検索

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568
8	小百合	sayuri@sample.co.jp	password		女性	大阪	06-0000-0000
9	典子	noriko@sample.co.jp	password		女性	大阪	06-0000-0000

# PHPでレコードを検索して結果を表示する (1)

- フォームからデータを送信して、検索条件に応じたレコードを抽出し、表示します。
- 2/index.phpに検索フォームを追加します。
- 今回はGETで送信してみます。

```
<form method="get" action="." class="my-4">
  <div class="form-group">
    <label for="pref">事業所所在地</label>
    <select name="pref" id="pref" class="form-control">
      <option value="">全件表示</option>
      <option value="大阪">大阪</option>
      <option value="兵庫">兵庫</option>
      <option value="京都">京都</option>
      <option value="奈良">奈良</option>
      <option value="和歌山">和歌山</option>
    </select>
  </div>
  <div class="my-2">
    <input type="submit" value="検索" class="btn btn-primary">
  </div>
</form>
```



# PHPでレコードを検索して結果を表示する (2)

- フォームから送信された（POSTされた）値から、該当のレコードを検索します。
- 2/index.phpの該当箇所を修正します。

```
<?php
// 検索条件
$pref = '';
// <form method="get">で送信したデータは、どうやって受け取るんだったっけ？
if (isset($_GET['pref'])) { // isset()についてはP23参照
    // フォームから送信されたデータがあれば、変数に代入する
    $pref = $_GET['pref'];
}

// データベースに接続するための文字列（DSN・接続文字列）
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';

// PDOクラスのインスタンスを作る
$dbh = new PDO($dsn, 'root', '');
```

# PHPでレコードを検索して結果を表示する (3)

// データを表示するためのSQL文

```
$sql = 'select * from users';
```

```
if ($pref) {
```

// フォームから値が送信されたときは、検索条件を追加

```
$sql .= ' where pref=:pref'
```

```
}
```

// SQLを実行する準備

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

```
if ($pref) {
```

// フォームから値が送信されたときは、SQL文の中のパラメータに値を割り当てる

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

```
}
```

// SQL文を実行

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

// 実行結果をすべて取り出し、変数に代入

```
$list = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

```
?>
```

# PHPでレコードを検索して結果を表示する (4)

- 検索できますか？

事業所所在地

和歌山

検索

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号
5	桃子	momoko@sample.co.jp	password		女性	和歌山	000-000-0000
11	小次郎	kojiro@sample.jp	password		男性	和歌山	03-0000-0000
14	はるか	haruka@sample.jp	password		女性	和歌山	03-0000-0000

# PHP入門

③レコードの  
追加・修正・削除  
を行おう！

※ 「データベース入門」のワークの資料も  
見てね！



# PHPでデータを追加・修正・削除する

PHPワーク

一覧表示

ユーザー追加

Dropdown ▼

Search

Search

事業所所在地

兵庫

検索

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号		
4	順子	junko@sample.co.jp	password		女性	兵庫	000-000-0000	修正	削除
7	涼子	ryoko@sample.co.jp	password		女性	兵庫	000-000-0000	修正	削除
13	今日子	kyoko@sample.jp	password		女性	兵庫	000-000-0000	修正	削除

# PHPでレコードを追加する

PHPワーク

一覧表示

ユーザー追加

Dropdown ▼

Search

Search

お名前

メールアドレス

パスワード

☒ 男性

☐ 女性

☐ その他

事業所所在地

大阪

◆

電話番号

☐ 管理者の方はチェックを入れてください。

送信

リセット

# レコードを追加するフォームを作成しよう (1)

- 1/index.htmlを2/add.phpにコピーします。
- 下記を2/add.phpに追記します。
- Bootstrapの公式サイトソースをコピペして、赤字の部分を書き換えます

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href=".">PHPワーク</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item">
        <a class="nav-link" href=".">一覧表示 <span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item active">
        <a class="nav-link" href="/add.php">ユーザー追加</a>
      </li>
    </ul>
  </div>
</nav>
```

# レコードを追加するフォームを作成しよう (2)

```
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button"
    data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Dropdown
  </a>
  <div class="dropdown-menu" aria-labelledby="navbarDropdown">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="#">Something else here</a>
  </div>
</li>
</ul>
<form class="form-inline my-2 my-lg-0">
  <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-
    label="Search">
  <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
</form>
</div>
</nav>
```



# レコードを追加するフォームを作成しよう (3)

- 下記の赤字の部分を書き換えます。

```
<form method="post" action="./add_action.php" class="my-3">
```

```
<input type="submit" value="追加" class="btn btn-primary">
```

# PHPでレコードを追加するには

```
<?php
// データベースに接続
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

// レコードを追加するSQL文（インサート文）
$sql = 'insert into users (name, email) values (:name, :email)';

// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':name', $_POST['name'], PDO::PARAM_STR);
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();
```

文字列、小数点がある数値のとき  
**PDO::PARAM\_STR**  
整数のとき  
**PDO::PARAM\_INT**  
を指定する。

# PHPでレコードを追加するコードを書こう (1)

- 2/add\_action.php 新規作成します。

```
<?php
// データベースに接続するための文字列 (DSN・接続文字列)
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

// レコードを追加するSQL文
$sql = 'insert into users (';
$sql .= 'name,';
$sql .= 'email,';
$sql .= 'password,';
$sql .= 'gender,';
$sql .= 'is_admin,';
$sql .= 'pref,';
$sql .= 'tel';
$sql .= ') values (';
$sql .= ':name,';
$sql .= ':email,';
$sql .= ':password,';
$sql .= ':gender,';
$sql .= ':is_admin,';
$sql .= ':pref,';
$sql .= ':tel';
$sql .= ')';
```

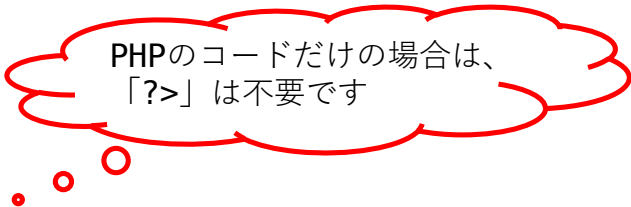
# PHPでレコードを追加するコードを書こう (2)

```
// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':name', $_POST['name'], PDO::PARAM_STR);
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);
$stmt->bindValue(':password', $_POST['password'], PDO::PARAM_STR);
if (isset($_POST['is_admin']) && $_POST['is_admin'] == 1) {
    // isset() って、何でしたっけ? p23参照
    $isAdmin = 1;
} else {
    $isAdmin = 0;
}
$stmt->bindValue(':is_admin', $isAdmin, PDO::PARAM_INT);
$stmt->bindValue(':gender', $_POST['gender'], PDO::PARAM_INT);
$stmt->bindValue(':pref', $_POST['pref'], PDO::PARAM_STR);
$stmt->bindValue(':tel', $_POST['tel'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();

// 処理が終わったらトップページへリダイレクト (ジャンプする)
header('Location: ./');
exit;
```



PHPのコードだけの場合は、  
「?>」は不要です

# PHPでレコードを修正する

PHPワーク

一覧表示

ユーザー追加

Dropdown ▼

Search

Search

お名前

今日子

メールアドレス

kyoko@sample.jp

パスワード

.....

☐ 男性

☒ 女性

☐ その他

事業所所在地

兵庫

電話番号

000-000-0000

☐ 管理者の方はチェックを入れてください。

更新

リセット

# PHPでレコードを修正するには

```
<?php
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

// レコードを修正するSQL文（アップデート文）
$sql = 'update users set name=:name, email=:email where id=:id';

// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':id', $_POST['id'], PDO::PARAM_INT);
$stmt->bindValue(':name', $_POST['name'], PDO::PARAM_STR);
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();
```

文字列、小数点がある数値のとき  
**PDO::PARAM\_STR**  
整数のとき  
**PDO::PARAM\_INT**  
を指定する。

# PHPでレコードを修正しよう！(1)

- 2/index.php を修正する。

```
<!-- テーブルに列を1列追加 -->  
<th></th>
```

```
<!-- レコードのidを送信し、送信先でそのidのレコードを検索して、項目の修正を行う -->  
<form action="./update.php" method="post" style="display: inline">  
    <input type="hidden" name="id" value="<?= $v['id'] ?>">  
    <input type="submit" class="btn btn-primary" value="修正">  
</form>
```

# PHPでレコードを修正しよう！(2)

- 2/index.phpの表示を確認しよう！

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号	
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568	修正
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000	修正
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000	修正
4	順子	junko@sample.co.jp	password		女性	兵庫	000-000-0000	修正
5	桃子	momoko@sample.co.jp	password		女性	和歌山	000-000-0000	修正
6	司	tsukasa@sample.co.jp	password		男性	京都	000-000-0000	修正
7	涼子	ryoko@sample.co.jp	password		女性	兵庫	000-000-0000	修正



# PHPでレコードを修正しよう！(3)

1. 2/add.php → 2/update.php にコピーします。
2. 2/update.php を編集します。

```
<?php
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

$sql = "select * from users where id=:id";
$stmt = $dbh->prepare($sql);
$stmt->bindValue(':id', $_POST['id'], PDO::PARAM_INT);
$stmt->execute();

$ret = $stmt->fetch(PDO::FETCH_ASSOC);
?>
```

# PHPでレコードを修正しよう！(4)

- 2/update.php の下記の箇所を追記・修正します。

```
<form method="post" action="./update_action.php" class="my-3">
```

```
<input type="hidden" name="id" value="<?= $ret['id'] ?>">
```

```
<input type="text" name="name" id="name" class="form-control" value="<?= $ret['name'] ?>">
```

```
<input type="text" name="email" id="email" class="form-control" value="<?= $ret['email'] ?>">
```

```
<input type="password" id="password" name="password" class="form-control" value="<?= $ret['password'] ?>">
```

```
<input type="radio" name="gender" id="gender1" value="1" class="form-check-input"<?php if ($ret['gender'] == 1) echo ' checked' ?>>
```

```
<input type="radio" name="gender" id="gender2" value="2" class="form-check-input"<?php if ($ret['gender'] == 2) echo ' checked' ?>>
```

```
<input type="radio" name="gender" id="gender9" value="9" class="form-check-input"<?php if ($ret['gender'] == 9) echo ' checked' ?>>
```

# PHPでレコードを修正しよう！(5)

- 2/update.php の下記の箇所を追記・修正します

```
<select name="pref" id="pref" class="form-control">
    <option value="大阪"><?php if ($ret['pref'] == '大阪') echo ' selected' ?>>大阪</option>
    <option value="兵庫"><?php if ($ret['pref'] == '兵庫') echo ' selected' ?>>兵庫</option>
    <option value="京都"><?php if ($ret['pref'] == '京都') echo ' selected' ?>>京都</option>
    <option value="奈良"><?php if ($ret['pref'] == '奈良') echo ' selected' ?>>奈良</option>
    <option value="和歌山"><?php if ($ret['pref'] == '和歌山') echo ' selected' ?>>和歌山</option>
</select>

<input type="text" name="tel" id="tel" class="form-control" value="<?= $ret['tel'] ?>">

<input type="checkbox" name="is_admin" id="is_admin" value="1" class="form-check-input"><?php if
($ret['is_admin'] == 1) echo ' checked' ?>>

<input type="submit" value="修正" class="btn btn-primary">
```

# PHPでレコードを修正しよう！(6)

1. 2/add\_action.php を 2/update\_action.php にコピーする。
2. 2/update\_action.php を編集する。

```
<?php
// データベースに接続
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

// レコードを修正するSQL文
$sql = '';
$sql .= 'update users set ';
$sql .= 'name=:name,';
$sql .= 'email=:email,';
$sql .= 'password=:password,';
$sql .= 'gender=:gender,';
$sql .= 'is_admin=:is_qdmin,';
$sql .= 'pref=:pref,';
$sql .= 'tel=:tel ';
$sql .= 'where id=:id';
```

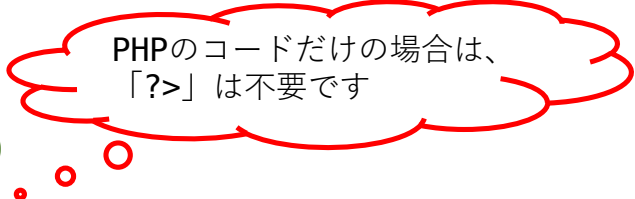
# PHPでレコードを修正しよう！(7)

```
// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':id', $_POST['id'], PDO::PARAM_INT);
$stmt->bindValue(':name', $_POST['name'], PDO::PARAM_STR);
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);
$stmt->bindValue(':password', $_POST['password'], PDO::PARAM_STR);
if (isset($_POST['is_admin']) && $_POST['is_admin'] == 1) {
    // isset() って、何でしたっけ？ p23参照
    $isAdmin = 1;
} else {
    $isAdmin = 0;
}
$stmt->bindValue(':is_admin', $isAdmin, PDO::PARAM_INT);
$stmt->bindValue(':gender', $_POST['gender'], PDO::PARAM_INT);
$stmt->bindValue(':pref', $_POST['pref'], PDO::PARAM_STR);
$stmt->bindValue(':tel', $_POST['tel'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();

// 処理が終わったらトップページへリダイレクト（ジャンプする）
header('Location: ../');
```



PHPのコードだけの場合は、  
「?>」は不要です

# PHPでレコードを削除しよう (1)

- 2/index.php に追記します。

```
<!-- レコードのidを送信し、送信先でそのidのレコードを検索して、削除を行う -->
<form action="./delete_action.php" method="post" style="display: inline">
  <input type="hidden" name="id" value="<?= $v['id'] ?>">
  <input type="submit" class="btn btn-primary" value="削除">
</form>
```

# PHPでレコードを削除しよう (2)

- 2/index.phpの表示を確認しましょう。

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号		
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568	修正	削除
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000	修正	削除
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000	修正	削除
4	順子	junko@sample.co.jp	password		女性	兵庫	000-000-0000	修正	削除
5	桃子	momoko@sample.co.jp	password		女性	和歌山	000-000-0000	修正	削除
6	司	tsukasa@sample.co.jp	password		男性	京都	000-000-0000	修正	削除
7	涼子	ryoko@sample.co.jp	password		女性	兵庫	000-000-0000	修正	削除

# PHPでレコードを削除しよう (3)

- 2/delete\_action.php を新規追加する。

```
<?php
// データベースに接続
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

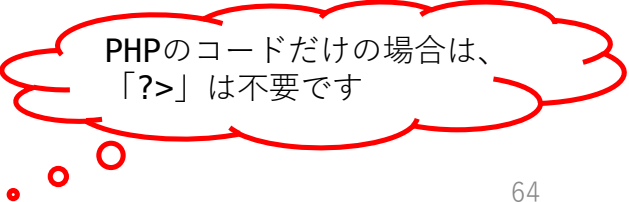
// レコードを修正するSQL文
$sql = delete from users where id=:id';

// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':name', $_POST['id'], PDO::PARAM_INT);

// SQL文を実行
$stmt->execute();

// 処理が終わったらトップページへリダイレクト (ジャンプする)
header('Location: ./');
exit;
```



PHPのコードだけの場合は、  
「?>」は不要です



# PHP入門

④セッションを使って  
ログイン・ログアウト機能を実装しよう！



# ログイン・ログアウト機能を実装しよう！

A UI mockup of a login form. It features a light gray header bar with the text 'ログイン'. Below this, there are two text input fields: the first is labeled 'メールアドレス' and the second is labeled 'パスワード'. At the bottom of the form is a blue button with the text 'ログイン' in white.

ログイン

メールアドレス

パスワード

ログイン

※ ログインしないと使えないようにします。

# ログイン・ログアウト機能を実装しよう！

ログイン

該当するユーザーが見つかりません

メールアドレス

パスワード

ログイン

※ ログインに失敗すると、エラーが表示されます。

# ログイン・ログアウト機能を実装しよう！

PHPワーク

一覧表示

ユーザー追加

森高千里さん ▼

Search

Search

事業所所在地

全件表示

検索

Action

Another action

ログアウト

id	お名前	email	パスワード	管理者	性別	勤務地	勤電話番号	
1	ちえみ	chiemi@sample.jp	password		女性	大阪	000-123-4568	<div>修正</div> <div>削除</div>
2	花子	hanako@sample.jp	password		女性	京都	000-000-0000	<div>修正</div> <div>削除</div>
3	次郎	jiro@sample.co.jp	password		男性	奈良	000-000-0000	<div>修正</div> <div>削除</div>

# セッションとは？ (1)

HTTP通信は

- リクエストに対してレスポンスが返ってきます。
- Webサーバーはレスポンスを返すと、クライアントのことを忘れてしまいます。  
→ Webサーバーはクライアントからのアクセスを特定することができません。

Webサーバーとクライアントを永続的に結びつけるために「セッション」が使われます。

セッションは、

- シンプルな方法で個々のユーザーのデータを格納する仕組みです。
- 個々のクライアントに対して一意なセッション ID を用意します。
- 複数ページにまたがるリクエストの間で状態の情報を永続させることができます。
- 異なるページ間で同じ変数の値を再利用することができます。  
→ 通常の変数は異なるページ間で再利用できませんよね？

セッションには「クッキー」が使われます。

クッキーに保存されるのはセッションIDのみです。



sess\_hrbd5mv2doq152tkofkmevnnie



sess\_hrbd5mv2doq152tkofkmevnnie

# セッションとは？ (2)

セッションを使えるようにするには、ページの先頭で下記のコードを書きます。

```
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);
```

PHPのセッションは

`$_SESSION`

という**連想配列**になっています。

```
$_SESSION['foo'] = 'こんにちは';
```

別のページで

```
echo $_SESSION['foo'];    // 「こんにちは」と表示されます。
```

# ログイン機能を実装するには (1)

1. ログイン機能を実装するには、セッションを使います。

2. メールアドレス、パスワードに一致するユーザーがいるかどうか調べます。

```
$sql = 'select * from users where email=:email and password=:password';  
$stmt = $dbh->prepare($sql);  
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);  
$stmt->bindValue(':password', $_POST['password'], PDO::PARAM_STR);  
$stmt->execute();
```

```
$ret = $stmt->fetch(PDO::FETCH_ASSOC);          // レコードがなかったらfalseが返ります
```

3. \$retが

- falseだったら、ログインページへリダイレクトします。
- trueだったら、
  - **セッション**にユーザー情報を登録します。  
`$_SESSION['user'] = $ret;`
  - 一覧ページへリダイレクトします。

# ログイン機能を実装するには (2)

セッションにユーザー情報が

- あったら、ページを表示します。
- なかったら、ログインページへリダイレクトします。

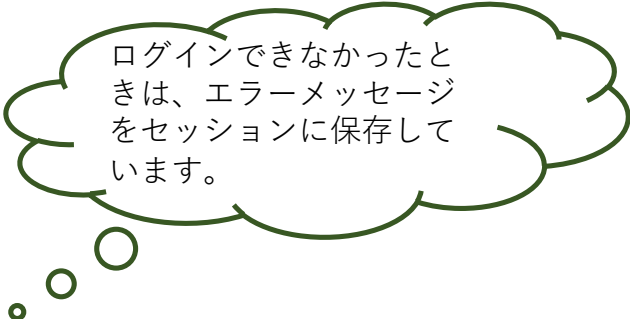
```
// セッションにログインユーザー情報がない場合、ログインページへリダイレクトする
if (!isset($_SESSION['user'])) {
    header('Location: ./login.php');
    exit;
}
```



# ログインページを作ろう！(1)

- 2/login.php

```
<div class="container">
  <div class="row my-5">
    <div class="col-md-4">
    </div>
    <div class="col-md-4">
      <div class="card">
        <div class="card-header">
          ログイン
        </div>
        <div class="card-body">
          <?php if (isset($_SESSION['error']) && $_SESSION['error']) : ?>
            <div class="alert alert-danger" role="alert">
              <?= $_SESSION['error'] ?>
            </div>
          <?php endif ?>
        </div>
      </div>
    </div>
  </div>
</div>
```



ログインできなかったときは、エラーメッセージをセッションに保存しています。

# ログインページを作ろう！ (2)

```
<form method="post" action="./login_action.php">
  <div class="form-group">
    <label for="email">メールアドレス</label>
    <input type="text" class="form-control" id="email" name="email">
  </div>
  <div class="form-group">
    <label for="password">パスワード</label>
    <input type="password" class="form-control" id="password"
      name="password">
  </div>
  <button type="submit" class="btn btn-primary">ログイン</button>
</form>
</div>
<div class="col-md-4">
</div>
</div>
</div>
</div>
</div>
```

# ログイン機能を作ろう！ (1)

- 2/login\_action.php

```
<?php
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);

// データベースに接続
$dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
$dbh = new PDO($dsn, 'root', '');

// ユーザーを検索するSQL文
$sql = '';
$sql .= 'select * from users ';
$sql .= 'where email=:email ';
$sql .= 'and password=:password';
```

# ログイン機能を作ろう！ (2)

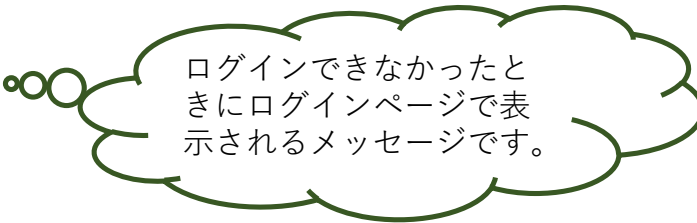
```
// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);
$stmt->bindValue(':password', $_POST['password'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();

// 実行結果を取り出し、変数に代入
$ret = $stmt->fetch(PDO::FETCH_ASSOC);

// 該当のレコードが存在しないときは、エラーメッセージをセッションに保存し、
// ログインページにリダイレクトする
if (!$ret) {
    $_SESSION['error'] = '該当するユーザーが見つかりません';
    header('Location: ./login.php');
    exit;
}
```



ログインできなかったときにログインページで表示されるメッセージです。

# ログイン機能を作ろう！ (3)

```
// 所得したユーザーの情報をセッションに保存する
$_SESSION['user'] = $ret;

// エラーメッセージを削除する
unset($_SESSION['error']);

// トップページへリダイレクト
header('Location: ./');
exit;
```

# ログイン機能を作ろう！ (4)

下記のファイルの先頭に

- 2/index.php
- 2/add.php
- 2/add\_action.php
- 2/update.php
- 2/update\_action.php
- 2/delete\_action.php

下記のコードを貼り付けます。

```
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);

// セッションにログインユーザー情報がない場合、ログインページへリダイレクトする
if (!isset($_SESSION['user'])) {
    header('Location: ../login.php');
    exit;
}
```

# ログイン機能を作ろう！ (5)

- ログインに成功したときは、セッションに保存したエラーメッセージを削除しておきましょう。
- 2/index.php

```
// エラーメッセージがあれば削除する  
if (isset($_SESSION['error'])) {  
    unset($_SESSION['error']);  
}
```



`$_SESSION['error']`  
を削除しなかったら  
どうなります？

# ログアウト機能を作ろう！(1)

- ユーザー情報を保存したセッションを削除したら、ログアウトされます。
- 2/logout.php

```
<?php
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);

// セッションに保存したログインユーザー情報を削除する
unset($_SESSION['user']);

// ログインページへリダイレクト
header('Location: ../login.php');
exit;
```



# ログアウト機能を作ろう！ (2)

下記のファイル

- 2/index.php
- 2/add.php
- 2/update.php

の赤字の部分編集します。

```
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button"
  data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    <?=$_SESSION['user']['name'] ?>さん
  </a>
  <div class="dropdown-menu" aria-labelledby="navbarDropdown">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="./logout.php">ログアウト</a>
  </div>
</li>
```

# ログアウト機能を作ろう！ (3)

下記のファイル

- 2/index.php
- 2/add.php
- 2/update.php

の</body>の前に下記のJSのコードを挿入します。

<https://getbootstrap.com/docs/4.4/getting-started/introduction/>

(**navbar**のドロップダウンを使うために必要です)

```
<script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwjl1yYfoRSJoZ+n"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
integrity="sha384-wfSDF2E50Y2D1uUdj003uMBJnjuUD4Ih7YwaYdliqfktj0Uod8GCExl3Og8ifwB6"
crossorigin="anonymous"></script>
```

# PHP入門

## ⑤例外処理について




# 例外とは！？

- プログラムの実行時に何らかの「**例外**的」なエラーが発生し、プログラムがそれ以上実行できなくなった状態のこと。
    - ✓ 例外が発生した、例外が起きた
    - ✓ 例外がスロー（throw）された、例外が投げられた
- といいます。

今まで作ってきたPHPのプログラムで例外が発生する可能性がある箇所としては、

- データベースの接続時
  - ✓ SQLiteのデータベースファイルのパスに誤りがある
  - ✓ データベースのファイルが壊れている
  - ✓ データベースに接続できない
- SQLの実行時
  - ✓ SQL文に誤りがある
  - ✓ テーブルが壊れている

# 例外が発生したとき



PDOException  
という例外が発生  
しています

- プログラムが止まります。

**Fatal error:** Uncaught PDOException: SQLSTATE[HY000]: General error: 1 no such table: users in /Applications/MAMP/htdocs/php\_work/4/login\_action.php:24

Stack trace:

#0 /Applications/MAMP/htdocs/php\_work/4/login\_action.php(24): PDO->prepare('select \* from u...')

#1 {main} thrown in **/Applications/MAMP/htdocs/php\_work/4/login\_action.php** on line **24**

**Fatal error:** Uncaught PDOException: SQLSTATE[HY000]: General error: 1 near "=": syntax error in /Applications/MAMP/htdocs/php\_work/4/login\_action.php:24

Stack trace:

#0 /Applications/MAMP/htdocs/php\_work/4/login\_action.php(24): PDO->prepare('select \* from u...')

#1 {main} thrown in **/Applications/MAMP/htdocs/php\_work/4/login\_action.php** on line **24**

# 例外とエラーの違い

- プログラムの文法間違い

**Parse error:** syntax error, unexpected '\$dbh' (T\_VARIABLE)  
in **/Applications/MAMP/htdocs/php\_work/4/login\_action.php** on line **9**

- 「注意」の場合は、プログラムが止まらない場合があります

**Notice:** Array to string conversion  
in **/Applications/MAMP/htdocs/php\_work/4/login\_action.php** on line **7**

※ Javaの場合は、すべての「エラー」が「例外」として扱われます。

※ PHPの場合は、「例外」と「エラー」は区別されます。

# 例外をスローする関数とクラス (1)

- PHPの関数は、全てが「例外」をスローするわけではありません。

## date

(PHP 4, PHP 5, PHP 7)

date — ローカルの日付/時刻を書式化する

## 説明

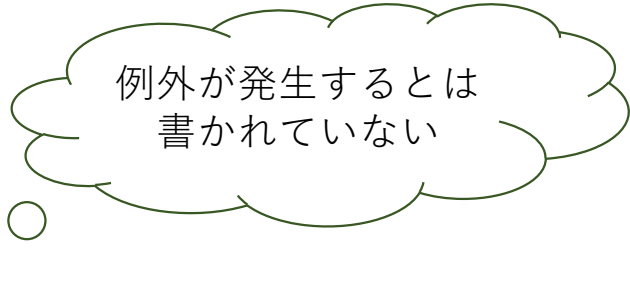
`date ( string $format [, int $timestamp = time() ] ) : string`

## 戻り値

日付を表す文字列を返します。 `timestamp` に数字以外が使用された場合は `FALSE` が返され、`E_WARNING` レベルのエラーが発生します。

## エラー / 例外

すべての日付/時刻関数は、有効なタイムゾーンが設定されていない場合に `E_NOTICE` を発生させます。また、システム設定のタイムゾーンあるいは環境変数 `TZ` を使用した場合には `E_STRICT` あるいは `E_WARNING` を発生させます。 [date default timezone set\(\)](#) も参照ください。



例外が発生するとは  
書かれていない

# 例外をスローする関数とクラス (2)

`DateTime::__construct`

`date_create`

(PHP 5 >= 5.2.0, PHP 7)

`DateTime::__construct` -- `date_create` — 新しい `DateTime` オブジェクトを返す

説明

オブジェクト指向型

`public DateTime::__construct ([ string $time = "now" [, DateTimeZone $timezone = NULL ] ] )`

手続き型

`date_create ([ string $time = "now" [, DateTimeZone $timezone = NULL ] ] ) : DateTime`

新しい `DateTime` オブジェクトを返します。

返り値

新しい `DateTime` のインスタンスを返します。手続き型の場合は、失敗したときに **FALSE** を返します。

エラー / 例外

エラーがあった場合は `Exception` を発生させます。 ○ ○ ○

例外が発生すると  
書かれている



# 例外に対応する方法（例外処理）

- 例外に対応するには、try ~ catch() を使います。

```
try {
```



```
} catch (Exception $e) {
```

- 例外が発生すると、処理が catch() ブロックに移動します。  
→ 例外を「補足（キャッチ）する」といいます。
- 例外の内容が \$e に代入されます。

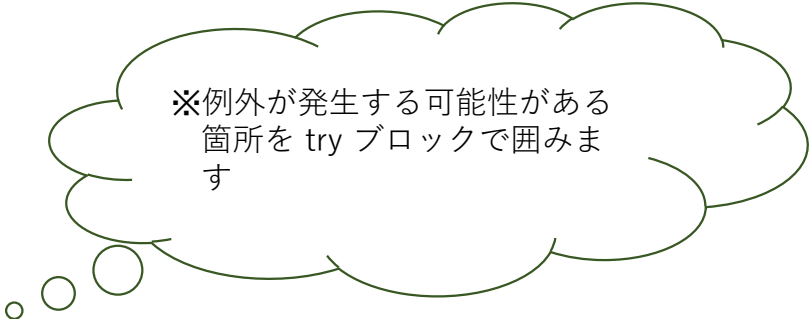
```
}
```

# 例外処理をしてみよう！ (1)

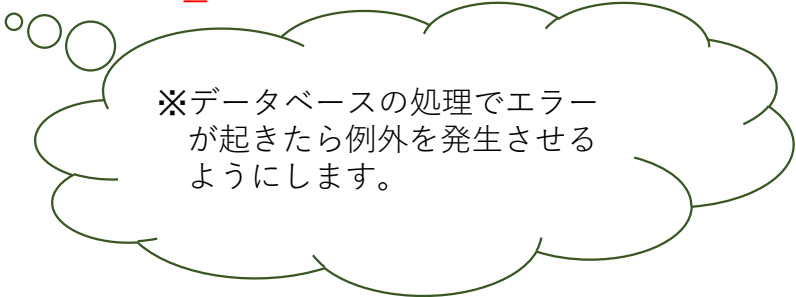
```
<?php
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);
```

```
try {
    // データベースに接続
    $dsn = 'mysql:dbname=php_work;host=localhost;charset=utf8';
    $dbh = new PDO($dsn, 'root', '');
    $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // ユーザーを検索するSQL文
    $sql = '';
    $sql .= 'select * from users ';
    $sql .= 'where email=:email ';
    $sql .= 'and password=:password';
```



※例外が発生する可能性がある  
箇所を try ブロックで囲み  
ます



※データベースの処理でエラー  
が起きたら例外を発生させる  
ようにします。

# 例外処理をしてみよう！ (2)

```
// SQLを実行する準備
$stmt = $dbh->prepare($sql);

// SQL文のパラメーターに値を割り当てる
$stmt->bindValue(':email', $_POST['email'], PDO::PARAM_STR);
$stmt->bindValue(':password', $_POST['password'], PDO::PARAM_STR);

// SQL文を実行
$stmt->execute();

// 実行結果を取り出し、変数に代入
$ret = $stmt->fetch(PDO::FETCH_ASSOC);

// 該当のレコードが存在しないときは、ログインページにリダイレクトする
if (!$ret) {
    $_SESSION['error'] = '該当するユーザーが見つかりません';
    header('Location: ./login.php');
    exit;
}
```

※これ以降は例外が発生する可能性がないので、ここに catch() ブロックを入れてもいいですが、コードの「可読性」が悪くなりますね😓

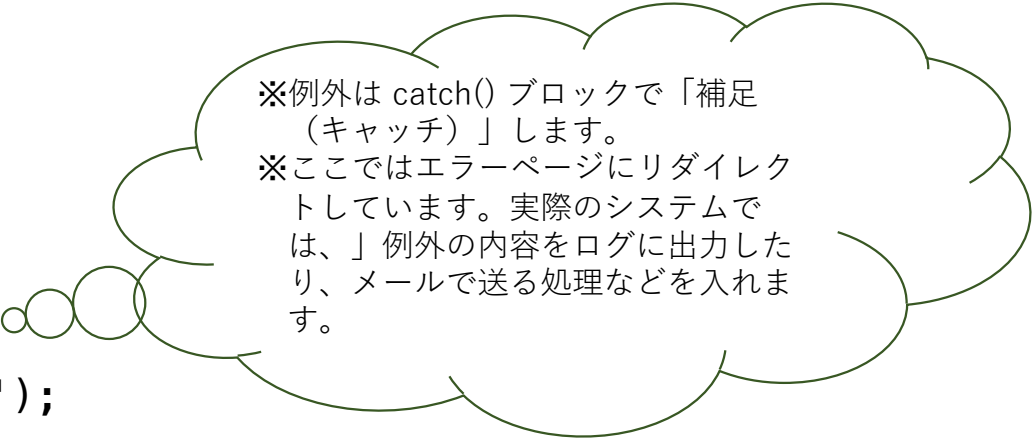
# 例外処理をしてみよう！ (3)

```
// 所得したユーザーの情報をセッションに保存する
$_SESSION['user'] = $ret;

// エラーメッセージを削除する
unset($_SESSION['error']);

// トップページへリダイレクト
header('Location: ./');
exit;

} catch (Exception $e) {
    // 例外発生時の処理
    // エラーページへリダイレクト
    header('Location: ./error.php');
    exit;
}
```



※例外は catch() ブロックで「補足（キャッチ）」します。  
※ここではエラーページにリダイレクトしています。実際のシステムでは、「例外の内容をログに出力したり、メールで送る処理などを入れます。」

※ データベースの処理を行っているページに「例外処理」を入れてみてくださいね😊

# エラーページを作ろう！ (1)

```
<?php
// セッションを有効にする
session_start();
// リクエストごとにセッションIDを切り替える
session_regenerate_id(true);

// セッションに保存したログインユーザー情報があれば削除する
if (isset($_SESSION['user'])) {
    unset($_SESSION['user']);
}

// セッションに保存したエラーメッセージ情報があれば削除する
if (isset($_SESSION['error'])) {
    unset($_SESSION['error']);
}
?>
<!DOCTYPE html>
<html lang="jp">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>エラー！</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
        integrity="sha384-Vkoo8x4CGs03+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
</head>
```

# エラーページを作ろう！ (2)

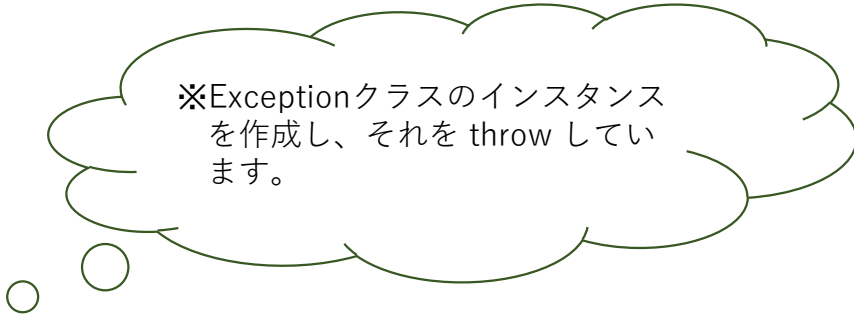
```
<body>
  <div class="container">
    <div class="row my-5">
      <div class="col-md-4"></div>
      <div class="col-md-4">
        <div class="card">
          <div class="card-body">
            <div class="alert alert-danger" role="alert">
              申し訳ございません。エラーが発生しました。
            </div>
            <button class="btn btn-danger" onclick="location.href='./login.php';">ログインページ
              へ戻る</button>
          </div>
        <div class="col-md-4"></div>
      </div>
    </div>
  </div>
</body>

</html>
```

# 番外編：わざと例外を投げる方法

- プログラムのコードの中で、わざと例外をスローすることもできます。

```
try {  
    // 例外をスローします  
    throw new Exception('エラーです');  
} catch (Exception $e) {  
    var_dump($e)  
}
```



※Exceptionクラスのインスタンスを作成し、それを throw しています。

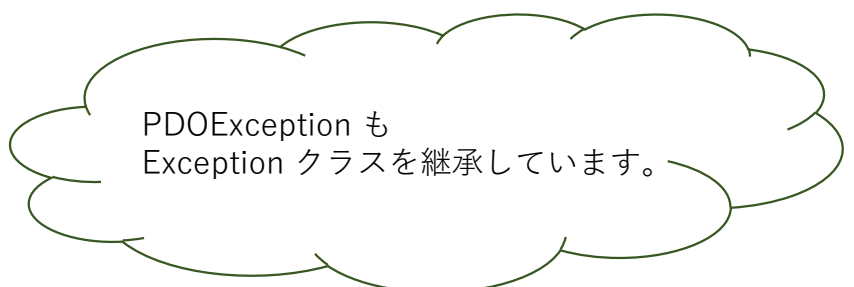
# 番外編：例外を効率的に使う方法 (1)

- 自作の関数（メソッド）を作る時に、例外を投げるようにします。
- メソッドの中でエラーを返すだけの（例外を投げない）メソッドを使う時に有効です。

```
public function checkData(array $arr)
{
    if (!isset($arr['foo'])) {
        throw new Exception('fooがありません！');
    }
}
```

- 例外は「クラス」なので、継承して新しい「例外クラス」を作ることができます。

```
class MyException extend Exception
{
    . . .
}
```



PDOException も  
Exception クラスを継承しています。



# 番外編：例外を効率的に使う方法 (2)

- 例外の種類（例外クラスを継承した例外クラス）ごとに catch() ブロックを設けることができます。

```
try {  
    . . .  
} catch HTTPException ($e) {  
    // HTTPException がスローされたときの処理  
} catch PDOException ($e) {  
    // PDOException がスローされたときの処理  
} catch Exception ($e) {  
    // 上記以外の例外がスローされたときの処理  
}
```