

PHP 練習問題. 09 例外処理

(1). 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。

今までの練習問題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
 - ✓ データベースに接続できない（止まっている？）
 - ✓ MySQL のデータベースの接続に誤りがある（接続ユーザー名、パスワードの誤りなど）
 - ✓ データベースが壊れている
- SQL の実行時
 - ✓ SQL 文に誤りがある
 - ✓ テーブルにアクセスできない、テーブルが壊れている

例外が発生したとき

- 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: 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

(2). 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。

- 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。

(例外がスローされない関数)

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

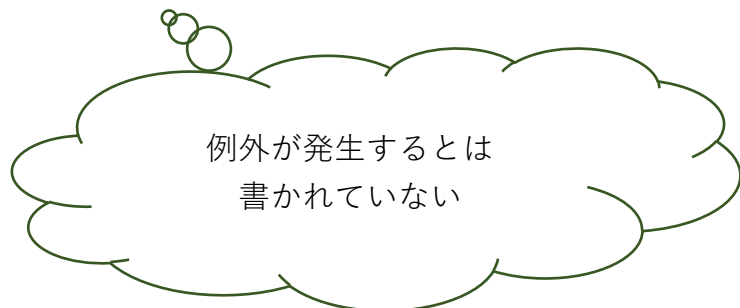
date (string \$format [, int \$timestamp = time()]) : string

返り値

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。



(例外がスローされる関数)

`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
```

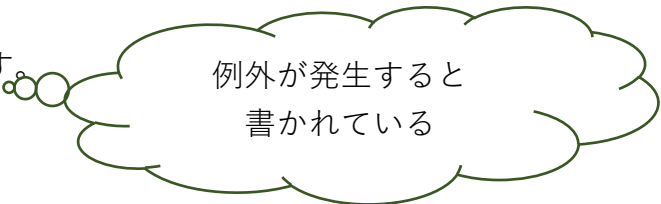
新しい `DateTime` オブジェクトを返します。

戻り値

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

(3). 例外処理

- 例外に対応するには、`try ~ catch()` を使います。

```
try {  
    // 例外が発生する処理  
} catch (Exception $e) {  
    // 例外が発生したときの処理  
}
```

- ✓ 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。

(4). 例外をスローする

- プログラムの中で例外をスローすることができます。

```
try {  
    // 例外をスローします  
    throw new Exception('エラー内容');  
} catch (Exception $e) {  
    var_dump($e->getMessage());  
    exit;  
}
```

~~~~~

設問1. 「練習問題 08 設問 7」の index.php で、例外処理を追加してください。例外が発生したときの処理は、

- ✓ var\_dump()で例外の内容を表示する。
- ✓ exit でプログラムの動作を止める。

にしてください。

(ヒント)

(1). データベース処理でエラーが発生したときに、例外をスローするようにします。

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

```
// エラーが起きたときのモードを指定する  
// 「PDO::ERRMODE_EXCEPTION」を指定すると、エラー発生時に例外がスローされる  
$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

(2). XAMPP の MySQL を STOP して、index.php にアクセスしてみてください。

- add.php、action.php も、同様に例外処理を入れてみましょう。

設問2. index.php で、下記のような form を作成してください。

- index.php

日付を入力してください

- (1) 送信先は action.php にしてください。
- (2) 「/」区切りの日付を入力して「送信」ボタンをクリックします。

- action.php

- (1) POST されてきた日付の文字列が、日付として正しいかどうか確認します。
- (2) 日付が正しいかどうかの判定は、関数を作ってください。下記の関数を使うと実現できます。
  - ✓ checkdate()  
<https://www.php.net/manual/ja/function.checkdate.php>
  - ✓ explode()  
<https://www.php.net/manual/ja/function.explode.php>
- (3) 日付が正しくないときは例外をスローし
  - ✓ セッションにエラーメッセージを代入してください。
  - ✓ index.php にリダイレクトしてください。
- (4) 日付が正しいときは、「正しい日付です」と表示します。

| 判定結果    |
|---------|
| 正しい日付です |

- index.php

(1) POST した日付の形式が正しくなかったときは、セッションに保存したエラーメッセージを表示します。

(2) POST された値をテキストボックスに表示します。

日付の形式が正しくありません

日付を入力してください

2020/02/30

送信