Pentalog

PHP Training

Episode 2

by Andrei Kojusko, 2018

Lesson contents

- Constants
- require/include
- Filesystem
- JSON
- Databases (PDO)
- Regular expressions
- Array manipulations

Constants - defining

```php
// Works as of PHP 5.3.0
const CONSTANT = 'Hello World';



// Works as of PHP 5.6.0

const ANOTHER_CONST = CONSTANT.'; Goodbye
World';

const ANIMALS = ['dog', 'cat', 'bird'];
```

```php
// Works as of PHP 7

define('ANIMALS', [
    'dog',
    'cat',
    'bird'
]);
```

Constants - accessing

```php
const CONSTANT = 'Hello World';
echo CONSTANT;

// Hello World!

echo UNDEFINED_CONSTANT;

// notice - undefined constant
// UNDEFINED_CONSTANT


echo SomeClass::UNDEFINED_CONSTANT;

// error!
```

```php
defined(UNDEFINED_CONSTANT)

// false


constant($const)

// constant value
// OR
// null + error if undefined
```

Require / include

```php
// vars.php :
$a = 1;

// other.php :
include 'vars.php';
echo $a; // 1
$a = 2;

function foo()
{
    include 'vars.php';
    echo $a; // 1
}

echo $a; // 2
include 'vars.php';
echo $a; // 1
```

```php
include 'notfound.php'; // warning

require 'notfound.php'; // error
```

```php
include_once 'vars.php';
$a = 2;
```

```php
include_once 'vars.php';
echo $a; // 2
```

Filesystem

```php
file_exists('/path/to/foo.txt'); // true / false
```

```php
$file = file_get_contents('./people.txt'); // string
$page = file_get_contents('http://www.example.com/');
```

```php
file_put_contents('foo.txt', 'some text');
```

```php
mkdir("/path/to/my/dir", 0700);
```

```php
if (is_dir($dir)) {
    if ($dh = opendir($dir)) {
        while (($file = readdir($dh)) !== false) {

            echo 'filename: '.$file."\n";

        }
        closedir($dh);
    }
}
```

Filesystem - links

http://php.net/manual/en/book.filesystem.php

http://php.net/manual/en/book.dir.php

JSON

```php
$arr = ['a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5];

echo json_encode($arr);

// {"a":1,"b":2,"c":3,"d":4,"e":5}
```

```php
$arr = [1, 2, 3, 4, 5];

echo json_encode($arr);

// [1,2,3,4,5]
```

```php
var_dump(json_decode($json));

object(stdClass)#1 (5) {
   ["a"] => int(1)
   ["b"] => int(2)
   ["c"] => int(3)
   ["d"] => int(4)
   ["e"] => int(5)
}
```

```php
var_dump(json_decode($json, true));

array(5) {
   ["a"] => int(1)
   ["b"] => int(2)
   ["c"] => int(3)
   ["d"] => int(4)
   ["e"] => int(5)
}
```

Databases (PDO) - connecting

```php
$dbh = new \PDO('sqlite:example.db');
```

```php
$dsn = 'mysql:dbname=testdb;host=127.0.0.1';
$user = 'dbuser';
$password = 'dbpass';

try {

    $dbh = new \PDO($dsn, $user, $password);

} catch (\PDOException $e) {

    echo 'Connection failed: ' . $e->getMessage();

}
```

Databases (PDO) - fetching data

```
$result = $conn->query('SELECT name, color, calories FROM fruit ORDER BY name');

foreach ($result as $row) {
    print $row['name'] . "\t";
    print $row['color'] . "\t";
    print $row['calories'] . "\n";
}
```

Databases (PDO) - request parameters

```
$calories = 150;

$colour = 'red';

$sth = $dbh->prepare('SELECT name, colour, calories

    FROM fruit

    WHERE calories < :calories AND colour = :colour');

$sth->execute(array(':calories' => $calories, ':colour' => $colour));
```

Databases (PDO) - inserting

```
$sth = $dbh->prepare('INSERT INTO fruit (name, colour, calories)

    VALUES (:name, :colour, :calories)');


$sth->execute([':calories' => 33, ':colour' => 'red', ':name' => 'papple']);
```

Databases (PDO) - links

http://php.net/manual/en/class.pdo.php

http://php.net/manual/en/class.pdostatement.php

http://sqlitebrowser.org/

Regular expressions - match

```php
preg_match("/php/i", "PHP is the web scripting language of choice.");

preg_match('/[^.]+\.[^.]+$/', $host);

preg_match('/(fo+)([a-z]+)baz/', 'foobarbaz', $matches);
print_r($matches);

// 0 => foobarbaz
// 1 => foo
// 2 => bar

preg_match_all("|<[^>]+>(.*)</[^>]+>|U", "<b>example: </b><div align=left>this is a test</div>",  $out);
echo $out[1][0] . ", " . $out[1][1] . "\n";

// example: , this is a test
```

Regular expressions - replace

```
$clean = preg_replace('/[\n\r\t ]+/', ' ', $html);
```

```
$string = 'April 15, 2003';
$pattern = '/(\w+) (\d+), (\d+)/';
$replacement = '${3}-${1}-01';
echo preg_replace($pattern, $replacement, $string);

// 2003-April-01
```

```
$string = 'The quick brown fox jumps over the lazy dog.';
$patterns = ['/quick/', '/brown/', '/fox/'];
$replacements = ['slow', 'black', 'bear'];
echo preg_replace($patterns, $replacements, $string);

// The slow black bear jumps over the lazy dog.
```

Regular expressions - links

http://php.net/manual/en/function.preg-match.php

http://php.net/manual/en/reference.pcre.pattern.syntax.php

http://php.net/manual/en/reference.pcre.pattern.modifiers.php

Array manipulations - filtering/mapping

```php
$arr = ['a' => 1, 'b' => 2, 'c' => 3, 'd' => 4];

var_dump(array_filter($arr, function($v) {

    return $v >= 3;

}));
```

```
array(2) {
  ["c"]=> int(3)
  ["d"]=> int(4)
}
```

```php
function cube($n)
{
    return($n * $n * $n);
}

$a = array(1, 2, 3, 4, 5);
$b = array_map("cube", $a);

print_r($b);
```

```
[0] => 1
[1] => 8
[2] => 27
[3] => 64
[4] => 125
```

```php
array_walk($a, "cube");
```

Array manipulations - sorting

```php
$fruits = array("lemon", "orange", "banana", "apple");

sort($fruits);

// 0 => apple, 1 => banana, 2 => lemon, 3 => orange

asort($fruits);

// 3 => apple, 2 => banana, 0 => lemon, 1 => orange
```

```php
$a = array(3, 2, 5, 6, 1);

usort($a, function($a, $b)
{
   if ($a == $b) {
      return 0;
   }

   return ($a < $b) ? -1 : 1;
});

// 1, 2, 3, 5, 6
```

Array manipulations - more sorting

- ksort
- rsort
- krsort
- uasort
- uksort

Array manipulations - links

http://php.net/manual/en/ref.array.php

http://php.net/manual/en/array.sorting.php

Homework

1. >php import.php ***dirname***
   Finds, processes and imports all ***dirname***/*.json files into SQLite DB.
   Date format must be correct, type must contain only uppercase letter, owner must be lowercase alphanumeric. Incorrect entries are ignored. Files are deleted after processing.

   JSON structure:
   [{"date":"2010-11-24","type":"ERROR","owner":"somename","message":"Strange error has occurred."},{"date":"2010-11-25","type":"INFO","owner":"othername","message":"Just an info."}]

2. >php count.php ***owner***
   Creates or updates report/***owner***/counts.json with a total number of logs by type in DB for this owner.

   JSON structure:
   {"error":5,"info":2,"success":0}