Pentalog

PHP Training

Episode 7

by Andrei Kojusko, 2018

Lesson contents

- SPL
  - Iterators
  - Datastructures
- Loops in templates
- Conditions in templates
- Generators
- Variadic functions
- Anonymous Classes
- Garbage collection

Iteration

```php
$fruits = [
    "apple" => "yummy",
    "orange" => "ah ya, nice",
    "grape" => "wow, I love it!",
    "plum" => "nah, not me",
];

$obj = new ArrayObject($fruits);
$it = $obj->getIterator();

while ($it->valid()) {
    echo $it->key() . "=" . $it->current() . "\n";
    $it->next();
}

// OR
foreach ($it as $key=>$val) {
    echo $key.":".$val."\n";
}
```

```php
class UserFilter extends FilterIterator {
    private $userFilter;

    public function __construct(Iterator $iterator , $filter) {
        parent::__construct($iterator);
        $this->userFilter = $filter;
    }

    public function accept() {
        $user = $this->getInnerIterator()->current();
        return $user['name'] === $this->userFilter;
    }
}

$iterator = new UserFilter($object->getIterator(), 'andrei');
```

Iterators

- IteratorIterator
  - AppendIterator
  - InfiniteIterator
  - LimitIterator
  - NoRewindIterator
  - CachingIterator
    - RecursiveCachingIterator
  - FilterIterator
    - CallbackFilterIterator
      - RecursiveCallbackFilterIterator
    - RecursiveFilterIterator
      - ParentIterator
    - RegexIterator
      - RecursiveRegexIterator

- ArrayIterator
  - RecursiveArrayIterator

- EmptyIterator

- MultipleIterator

- RecursiveIteratorIterator
  - RecursiveTreeIterator

- DirectoryIterator (extends SplFileInfo)
  - FilesystemIterator
    - GlobIterator
    - RecursiveDirectoryIterator

SplDoublyLinkedList::setIteratorMode

- The direction of the iteration (either one or the other):
  - **SplDoublyLinkedList::IT_MODE_LIFO** (Stack style)
  - **SplDoublyLinkedList::IT_MODE_FIFO** (Queue style)
- The behavior of the iterator (either one or the other):
  - **SplDoublyLinkedList::IT_MODE_DELETE** (Elements are deleted by the iterator)
  - **SplDoublyLinkedList::IT_MODE_KEEP** (Elements are traversed by the iterator)

SplFixedArray

```php
// Initialize the array with a fixed length
$array = new SplFixedArray(5);

$array[1] = 2;
$array[4] = "foo";

var_dump($array[0]); // NULL
var_dump($array[1]); // int(2)
var_dump($array["4"]); // string(3) "foo"

// Increase the size of the array to 10
$array->setSize(10);
```

SplQueue / SplPriorityQueue

```php
$q = new SplQueue();

$q->enqueue(new Task(1));
$q->enqueue(new Task(77));

foreach ($q as $task) {

    $task = $q->dequeue();

    $q[] = $someNewTask;

}
```

```php
class PQtest extends SplPriorityQueue {
    public function compare($priority1, $priority2) {
        if ($priority1 === $priority2) return 0;
        return $priority1 < $priority2 ? -1 : 1;
    }
}

$objPQ = new PQtest();
$objPQ->insert('A',3);
$objPQ->insert('B',6);

$objPQ->top();

while($objPQ->valid()) {
    $item = $objPQ->current();

    $objPQ->next();
}
```

SplMaxHeap / SplMinHeap

```
class SortPeopleByAgeDescending extends SplMaxHeap {
      function compare($a, $b) { // optional!
            return $a->age - $b->age;
      }
}

$heap = new SortPeopleByAgeDescending();
$heap->insert($person1); // age 25
$heap->insert($person2); // age 30

foreach($heap as $person) {
   echo $person->age; // 30, 25
}
echo $heap->count(); // 0
```

Templates - loops

```
{% foreach list as item %}
        {{ item.name }}
{% endforeach %}
```

1. Detect foreach block, get iterable, element name, block contents
2. Create a loop in PHP with iterable
3. In loop, replace all element variables in content, concatenate results
4. Replace whole block in template with concatenated results

Templates - conditions

```
{% if userIsAuthenticated %}
    <a href="/logout">logout</a>
{% endif %}
```

1. Detect if block, get condition and block contents
2. Parse and verify condition
3. On true, replace whole block in template with content
4. On false, remove whole block from template

Generators

```
function getLinesFromFile($fileName) {
    if (!$fileHandle = fopen($fileName, 'r')) {
        return;
    }

    while (false !== $line = fgets($fileHandle)) {
        yield $line;
    }

    fclose($fileHandle);
}

foreach(getLinesFromFile('text.txt') as $line) {

        // do something with $line

}
```

- yield
- yield from
- yield $key => $value
- $data = yield $value;
    - Generator::send (mixed $value)

Generators - sending data

```
function myGenerator() {
        $data = yield 5;
        if ($data === 'stop') return;
        yield 8;
}

$a = myGenerator();

foreach ($a as $val) {
        echo ($val);
        $a->send('stop');
}
```

Variadic functions

```php
function sum(...$numbers) {

    $acc = 0;

    foreach ($numbers as $n) {

        $acc += $n;

    }

    return $acc;

}

echo sum(1, 2, 3, 4);
```

```php
function add($a, $b) {
    return $a + $b;
}

$a = [1, 2];
echo add(...$a);
```

```php
function total_intervals($unit, DateInterval ...$intervals) {
    $time = 0;
    foreach ($intervals as $interval) {
        $time += $interval->$unit;
    }

    return $time;
}
```

Anonymous Classes

```
$util->setLogger(new class {
    public function log($msg) {
        echo $msg;
    }
});
```

```
$someVar = new class(10)
        extends SomeClass
        implements SomeInterface
{
    private $num;
    public function __construct($num) {
        $this->num = $num;
    }

    use SomeTrait;
}
```

Garbage collection

```
$a = 'test';  // 4 bytes
$b = $a;     // still 4 bytes

$a = [1, 2, 3];        // 3
$b = $a;               // 3
$b[1] = 5;             // 6

$a = [1, 2, 3];        // 3
$b = &$a;              // 3
$b[1] = 5;             // 3

[1, 2, 3] <<< ['a' => 1, 'b' => 1, 'c' => 1]
```

- memory_get_usage
- memory_get_peak_usage

```
while (true) {
  //do memory intensive code
}

function intensive($parameters) {
  //do memory intensive code
}

while (true) {
  intensive($parameters);
}
```

Links

http://php.net/manual/en/spl.datastructures.php

http://php.net/manual/en/spl.iterators.php

http://php.net/manual/en/language.generators.syntax.php

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

http://php.net/manual/en/features.gc.php

Homework

1. Get user’s Git repositories
2. Show list of repositories with name, updated date, link, number of commits
3. List column header should be a link for sorting the list by the column