PHP array_map & Lambda Functions

array_map in PHP

array_map(callable $callback, array $array): array

This function applies a callback to each element of an array and returns a new array with the results.

Example 1 - Multiply numbers:

```
$nums = [1, 2, 3];
$double = array_map(function($n) {
    return $n * 2;
}, $nums);
```

Output: [2, 4, 6]

Example 2 - Uppercase names:

```
$names = ['ali', 'sara'];
$upperNames = array_map(function($x) {
    return strtoupper($x);
}, $names);
```

Output: ['ALI', 'SARA']

Example 3 - Transform product prices:

```
$products = [
    ['name' => 'Phone', 'price' => 1000, 'category' => 'Electronics'],
    ['name' => 'T-shirt', 'price' => 25, 'category' => 'Fashion']
];
$filtered = array_filter($products, function($p) {
    return $p['category'] === 'Electronics';
});
```

PHP array_map & Lambda Functions

```
$filtered = array_map(function($p) {
    return [
        'name' => $p['name'],
        'price' => $p['price'] * 1.1
    ];
}, array_values($filtered));

Output: [
    ['name' => 'Phone', 'price' => 1100]
]
```

Lambda / Anonymous Functions

Anonymous (Lambda) functions in PHP are functions without a name.

They are usually assigned to variables or passed directly into functions.

Example:

```
$greet = function($name) {
    return "Salam, $name!";
};

echo $greet("Mohammad"); // Output: Salam, Mohammad!
```

These functions are useful for short, inline logic (especially inside array_map, array_filter, etc).

The '?' symbol before callable (in array_map's definition) means that the callback can be NULL (optional).

PHP uses 'anonymous' and 'lambda' terms interchangeably.