# Параметры в роутах в MVC на PHP

В предыдущем уроке наши роуты имели фиксированные адреса. На самом деле механизм роутинга более сложный - можно сделать так, чтобы часть URI страницы попадала в именованные параметры, доступные затем в контроллере.

Пусть, к примеру, наши адреса будут выглядеть вот так: */test/параметр1/параметр2/*, где параметр1 и параметр2 - произвольные строки. При этом мы хотим, чтобы адреса такого вида обрабатывались одним действием контроллера.

Для этого следует придумать имя параметра и перед ним поставить двоеточие, вот так:

```php
<?php
  use \Core\Route;

  return [
    new Route('/test/:var1/:var2/', 'page', 'act'),
  ];
?>
```

В нашем случае получится, что все запросы вида */test/параметр1/параметр2/* будут попадать на действие act. При этом в первый параметр этого действия будет попадать ассоциативный массив с параметрами: текст, который будет на месте первого параметра, попадает в элемент массива с ключом **var1**, а текст второго параметра - в **var2**.

Пусть, к примеру, в адресной строке набрано следующее: */test/eee/bbb/*. Давайте посмотрим, что будет содержать первый параметр действия:

```php
<?php
  namespace Project\Controllers;
  use Core\Controller;

  class PageController extends Controller
  {
    public function act($params)
    {
      var_dump($params); // ['var1' => 'eee', 'va
        r2' => 'bbb']
     }
  }
?>
```

## Задача 94.1

Сделайте контроллер NumController, а в нем - действие sum. Пусть это действие обрабатывает адреса следующего вида: */nums/n1/n2/n3/*, где n1, n2 и n3 - некоторые числа. Сделайте так, чтобы на экран выводилась сумма переданных чисел.

# Применение

Давайте посмотрим применение описанного на более жизненном примере. Пусть наш контроллер PageController содержит массив страниц (эти данные должна отдавать модель, но с моделями мы еще не разобрались, поэтому пусть данные пока просто хранятся в контроллере):

```php
<?php
  namespace Project\Controllers;
```

```
  use Core\Controller;

  class PageController extends Controller
  {
    private $pages;

    public function __construct()
    {
      $this->pages = [
        1 => 'страница 1',
        2 => 'страница 2',
        3 => 'страница 3',
      ];
    }
  }
?>
```

Давайте сделаем действие **show**, которое будет выводить на экран страницу с определенным номером (id):

```
<?php
  namespace Project\Controllers;
  use Core\Controller;

  class PageController extends Controller
  {
    private $pages;

    public function __construct()
    {
      $this->pages = [
        1 => 'страница 1',
        2 => 'страница 2',
        3 => 'страница 3',
      ];
    }

    public function show()
    {
      // здесь выведем страницу с определенным ном
        ером
     }

  }
?>
```

Пусть при обращении к адресу */page/1/* будет выводится текст первой страницы, при обращении к адресу */page/2/* - текст второй страницы и так далее. Сделаем соответствующий роут:

```
<?php
  use \Core\Route;

  return [
    new Route('/page/:id/', 'page', 'show'),
  ];
?>
```

Реализуем описанный метод show:

```
<?php
  namespace Project\Controllers;
  use Core\Controller;

  class PageController extends Controller
  {
    private $pages;

    public function __construct()
    {
      $this->pages = [
        1 => 'страница 1',
```

```
            2 => 'страница 2',
            3 => 'страница 3',
        ];
    }

    public function show($params)
    {
        echo $this->pages[ $params['id'] ]; // выв
            одим страницу по номеру
    }
}
?>
```

## Задача 94.2

Реализуйте контроллер UserController, содержащий следующий массив:

```
<?php
  $this->users = [
    1 => ['name'=>'user1', 'age'=>'23', 'sal
         ary' => 1000],
    2 => ['name'=>'user2', 'age'=>'24', 'sal
         ary' => 2000],
    3 => ['name'=>'user3', 'age'=>'25', 'sal
         ary' => 3000],
    4 => ['name'=>'user4', 'age'=>'26', 'sal
         ary' => 4000],
    5 => ['name'=>'user5', 'age'=>'27', 'sal
         ary' => 5000],
  ] ;
?>
```

## Задача 94.3

В контроллере UserController, сделайте действие **show**, которое будет выводить юзера по определенному id. Пусть оно будет доступно по адресу следующего вида: */user/1/*, где вместо 1 может быть любое число.

## Задача 94.4

В контроллере UserController, сделайте действие **info**, которое будет выводить имя или возраст заданного юзера. Пусть это действие будет доступно по адресу следующего вида: */user/1/key/*, где вместо 1 будет id юзера, а вместо key - текст name, age или salary.

## Задача 94.5

В контроллере UserController, сделайте действие **all**, которое будет выводить список всех юзеров на экран. Пусть это действие будет доступно по адресу следующего вида: */user/all/* (параметров тут никаких не будет).

# Задача 94.6

В контроллере UserController, сделайте действие **first**, которое будет выводить список N первых юзеров на экран. Пусть это действие будет доступно по адресу следующего вида: */user/first/n/*, где вместо n будет количество юзеров, которые следует вывести на экран.