Разработка многостраничного сайта на PHP

АРХИТЕКТУРА ПРИЛОЖЕНИЙ И ШАБЛОНЫ ПРОЕКТИРОВАНИЯ
ЗАНЯТИЕ № 2 - ПРАКТИКА

Тема занятия – Архитектура приложений и шаблоны проектирования

Цель занятия –

Систематизация и обобщение знаний по шаблонам проектирования и архитектуре приложений в PHP

Актуализация

На прошлом занятии было изучено,

1) Что такое «Архитектура приложений»;
2) Что такое «Шаблоны проектирования»;
3) Какие бывают шаблоны проектирования;
4) Как эти понятия применяются в PHP.

Содержание занятия

1) Задача 1
2) Задача 2
3) Задача 3
4) Задача 4
5) Задача 5
6) Задача 6
7) Задача 7

Задача 1

Создайте класс, который будет гарантировать, что в приложении существует только один экземпляр этого класса.

Решение

```php
<?php
class Singleton {
    private static $instance = null;

    private function __construct() {
        // Приватный конструктор, чтобы предотвратить создание экземпляров через оператор new
    }

    public static function getInstance() {
        if (self::$instance == null) {
            self::$instance = new Singleton();
        }

        return self::$instance;
    }
}

$singleton = Singleton::getInstance();

?>
```

Задача 2

Создайте интерфейс для фабрики и два класса - для его реализации.

Решение

```php
<?php
// Интерфейс фабрики
interface FactoryInterface {
    public function createProduct();
}

// Класс, который реализует интерфейс фабрики и создает продукт 1
class FactoryProduct1 implements FactoryInterface {
    public function createProduct() {
        return new Product1();
    }
}

// Класс, который реализует интерфейс фабрики и создает продукт 2
class FactoryProduct2 implements FactoryInterface {
    public function createProduct() {
        return new Product2();
    }
}

// Пример использования
$factory1 = new FactoryProduct1();
$product1 = $factory1->createProduct(); // Создает продукт 1

$factory2 = new FactoryProduct2();
$product2 = $factory2->createProduct(); // Создает продукт 2

?>
```

Задача 3

Создайте класс, который будет хранить и обрабатывать информацию о количестве пользователей, которые используют приложение.

Решение

```php
<?php
class UserCounter {
    private $count;

    public function __construct() {
        // При создании объекта класса UserCounter,
        // мы инициализируем счётчик нулём
        $this->count = 0;
    }

    public function increment() {
        // Увеличиваем счётчик на 1
        $this->count++;
    }

    public function decrement() {
        // Уменьшаем счётчик на 1
        $this->count--;
    }

    public function getCount() {
        // Возвращаем текущее значение счётчика
        return $this->count;
    }
}
?>
```

Задача 4

Напишите калькулятор на PHP, используя шаблон проектирования Singleton.

Решение

```php
<?php

class Calculator {
    private static $instance;

    private function __construct()
    {
    }

    public static function getInstance()
    {
        if (!self::$instance) {
            self::$instance = new self();
        }
        return self::$instance;
    }

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

    public function subtract($a, $b)
    {
        return $a - $b;
    }

    public function multiply($a, $b)
    {
        return $a * $b;
    }

    public function divide($a, $b)
    {
        if ($b == 0) {
            throw new Exception('Division by zero.');
        }
        return $a / $b;
    }
}

// использование калькулятора
$calculator = Calculator::getInstance();
echo $calculator->add(5, 3); // выводит 8
echo $calculator->subtract(5, 3); // выводит 2
echo $calculator->multiply(5, 3); // выводит 15
echo $calculator->divide(6, 3); // выводит 2
echo $calculator->divide(6, 0); // выводит исключение "Division by zero."
```

Задача 5

Создайте абстрактный класс **AbstractAnimal** с абстрактным методом **makeSound**().
Создать два наследника этого класса: **Cat** и **Dog**.
Для каждого животного реализовать метод **makeSound**(), который будет выводить соответствующее звуковое сообщение.

Решение

```php
<?php

abstract class AbstractAnimal {
    abstract public function makeSound();
}
class Cat extends AbstractAnimal {
    public function makeSound() {
    echo "Мяу-мяу";
    }
}

class Dog extends AbstractAnimal {
    public function makeSound() {
        echo "Гав-гав";
    }
}
?>
```

Задача 6

Создайте страницу для отображения списка пользователей. Каждый пользователь должен иметь имя, фамилию и адрес электронной почты.

Решение

```html
<!DOCTYPE html>
<html>
<head>
    <title>Список пользователей</title>
    <style>
        table {
            border-collapse: collapse;
            width: 100%;
        }
        th, td {
            text-align: left;
            padding: 8px;
            border-bottom: 1px solid #ddd;
        }
        th {
            background-color: #f2f2f2;
        }
    </style>
</head>
<body>
    <h1>Список пользователей</h1>
    <table>
        <thead>
            <tr>
                <th>Имя</th>
                <th>Фамилия</th>
                <th>Адрес электронной почты</th>
            </tr>
        </thead>
```

```php
        <tbody>
            <?php
            // Массив с данными пользователей
            $users = array(
                array("Иван", "Иванов", "ivan.ivanov@example.com"),
                array("Петр", "Петров", "petr.petrov@example.com"),
                array("Мария", "Сидорова", "maria.sidorova@example.com")
            );

            // Цикл для вывода каждого пользователя в виде строки таблицы
            foreach ($users as $user) {
                echo "<tr>";
                echo "<td>" . $user[0] . "</td>";
                echo "<td>" . $user[1] . "</td>";
                echo "<td>" . $user[2] . "</td>";
                echo "</tr>";
            }
            ?>
        </tbody>
    </table>
</body>
</html>
```

Задача 7

Создайте страницу для отображения списка товаров. Каждый товар должен иметь название, описание и цену.

Решение

```php
<?php
// Массив с товарами
$products = [
    ['name' => 'Товар 1', 'description' => 'Описание товара 1', 'price' => 100],
    ['name' => 'Товар 2', 'description' => 'Описание товара 2', 'price' => 200],
    ['name' => 'Товар 3', 'description' => 'Описание товара 3', 'price' => 300],
    ['name' => 'Товар 4', 'description' => 'Описание товара 4', 'price' => 400],
    ['name' => 'Товар 5', 'description' => 'Описание товара 5', 'price' => 500]
];

// Выводим каждый товар из массива
foreach ($products as $product) {
    echo '<div>';
    echo '<h2>' . $product['name'] . '</h2>';
    echo '<p>' . $product['description'] . '</p>';
    echo '<p>Цена: ' . $product['price'] . ' рублей</p>';
    echo '</div>';
}
?>
```

Домашнее задание

Создайте простой пример для реализации MVC-архитектуры на PHP для веб-приложений, включающий список пользователей.

Разбор задачи будет в начале следующего урока.

Рефлексия

Сегодня были решены задачи по теме: «Архитектура приложений и шаблоны проектирования в PHP»

Ответьте на несколько вопросов:

1. Какая задача была самая интересная?
2. Какая задача показалась наиболее сложной?
3. Какую задачу вы поняли?

Спасибо за внимание