PHP Уровень 2

Урок 4

Построение View. Шаблонизация. Буферизация.

О чем речь?

Рассмотрим один из компонентов архитектуры MVC – представление (View)

- Как правильно работать с View
- Как применить ООП
- Почему PHP – отличный шаблонизатор?
- Что такое буфер вывода и как им пользоваться?

На чем мы остановились прошлый раз?

```
class NewsController {
  public function actionAll() {
    …
    include “all.php”;
  }
}
```

Мы подключаем шаблон как обычный файл PHP

- шаблону видны все переменные actionAll() – нет явной передачи данных в шаблон
- сильная связность, что плохо
- при смене пути к шаблону всё сломается

Как сделать лучше?

- Нужно создать класс View
- Он должен уметь хранить в себе данные для отображения
- И уметь отобразить выбранный файл шаблона с этими данными

Примерно так:

```
$view = new View;
$view->foo = ‘bar’;
$view->display(‘index.php’);
```

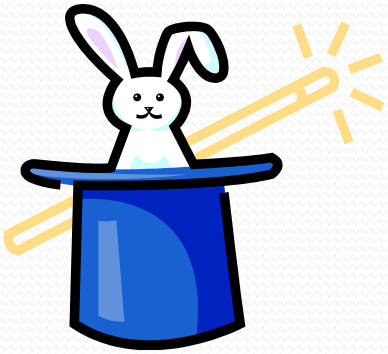
«Магия» в PHP

- Мы хотим создать объект, который сможет устанавливать себе любые свойста
- Разумеется, этот объект должен позволять читать свои свойства
- И он должны быть доступны нам через foreach

Решение:

«Магические» методы

«Магия» в PHP



public function __set($key, $val)

- вызывается при попытке установить недоступное свойство объекта
- $key – имя свойства
- $val – значение, которое мы пытаемся установить

public function __get($key)

- вызывается при попытке прочитать недоступное свойство объекта
- $key – имя этого свойства

«Магия» в PHP

Интерфейс

- это «контракт», который подписавшиеся классы обязаны реализовать
- похож абстрактный класс
- определяется ключевым слово **interface**
- в классе подключается с помощью слова **implements**
- несет в себе описание методов, которые обязан реализовать класс
- один класс может реализовать несколько интерфейсов

«Магия» в PHP

Интерфейс Iterator

- в случае реализации классом позволяет объекту этого класса быть перечисляемым

- public function **current**()
- public function **key**()
- public function **next**()
- public function **rewind**()
- public function **valid**()

v_1	v_2	v_3	…	v_n
k_1	k_2	k_3	…	k_n

Буферизация вывода

Проблема

- то, что мы уже послали в браузер, уже не изменить

Решение

- ob_start() начинает захват вывода в буфер
- ob_get_contents() получает содержимое буфера
- ob_get_clean() получает содержимое и очищает буфер

Выполнение произвольного кода

Проблема

- Как выполнить тот код, который находится в произвольном файле или строке?

Решение

```
eval('?>' . $code . '<?php');
```

ВНИМАНИЕ!

ЭТА КОНСТРУКЦИЯ ПРИВЕДЕНА ТОЛЬКО ДЛЯ ОЗНАКОМЛЕНИЯ! ИСПОЛЬЗОВАНИЕ В РЕАЛЬНЫХ ПРОЕКТАХ МОЖЕТ БЫТЬ ОПАСНО!

А что дальше?

- **Перепишите предыдущее ДЗ используя полученные знания**
  - Создайте класс View
  - Класс должен уметь устанавливать произвольные свойства своих объектов
  - В нем – метод display($path), отображающий заданный шаблон с заданными данными
  - Примените класс View в проекте