PHP Уровень 2

Урок 5

Построение моделей. PDO. Часть 1.

О чем речь?

Рассмотрим один из компонентов архитектуры MVC – модель (Model)

- Как правильно работать с Model
- Как применить ООП
- Что такое позднее статическое связывание?
- Исключения в PHP
- Концепции AR, CRUD, ORM
- Переходим на PDO

MVC: Model

Модель должна выполнять следующие функции

- Инкапсулировать в себе подключение к БД
- Выборка нужных данных из таблиц
- Операции CRUD:
  - create
  - read
  - update
  - delete
- Оптимально: быть реализацией концепций Active Record и ORM

MVC: Model. ORM.

Object-Relational Mapping.

Основные принципы:

- Класс – это отражение таблицы в БД
- Методы класса (статические методы) работают с таблицей в целом. Например – выбирают из нее данные
- Объект – это отражение записи в таблице
- Методы объекта (динамические) работают с отдельными записями. Например – модифицируют и сохраняют данные.

MVC: Model. ORM.

Object-Relational Mapping.

```
abstract class Model {
    static protected $table;
    static public function getTableName() {
        return self::$table;
    }
}
class Article extends Model {
    static protected $table = 'articles';
}
echo Article::getTableName();
```

Ничего не выводит. В чем же проблема?

MVC: Model. ORM.

В PHP self всегда указывает на тот класс, в котором написано это ключевое слово

- это называется «раннее связывание»
- **self** связывается с именем класса на этапе разбора кода, а не его выполнения

Позднее статическое связывание

- используем **static** вместо self
- в результате получим «позднюю связь» – на этапе выполнения
- с тем классом, который реально вызывает наш код

MVC: Model. ORM.

PDO: PHP Data Objects

- это объектный интерфейс к базе данных
- позволяет пользоваться подготовленными запросами
- избавляет от необходимости экранирования данных
- возвращает данные в объектном виде

MVC: Model. ORM.

PDO: PHP Data Objects

```
// Подключение к базе данных
$dsn = 'mysql:dbname=test;host=localhost';
$dbh = new Pdo($dsn, $user, $password);

// Подготовка запроса
$sth = $dbh->prepare(
    “SELECT * FROM news WHERE id=:id”
    );

// Выполнение запроса с подстановкой
$sth->execute([‘:id’ => 1]);

// Получение результата запроса (все строки)
$result = $sth->fetchAll()
```

MVC: Model. ORM.

Исключения

- Исключение – это исключительная ситуация, о которой мы предполагаем, что она может произойти
- Исключение – это объект специального класса, наследника класса Exception
- Исключение может быть «выброшено» оператором **throw**
- Исключение после «выбрасывания» начинает «всплывать» вверх по иерархии кода, пока не встретится блок **try … catch**
- В этом блоке исключение можно «поймать» и предпринять какие-то действия
- Правило хорошего тона – свои классы исключений, образующие иерархию

А что дальше?

Перепишите предыдущее ДЗ, используя полученные знания

- Создайте класс AbsractModel, от него унаследуйте модели ваших данных. Используйте LSB для получения имени таблицы
- Создайте класс DbConnection, поручите ему соединение с БД, запросы к ней и возврат результатов запросов в виде объектов нужного Вам класса. Используйте PDO.
- Определите в моделях методы ::findAll(), ::findByPk($id), используйте класс DbConnection для реализации этих методов.
- Определите места, где могут возникать исключения, предусмотрите их обработку и вывод ошибок.