自己搭建框架

学习框架的意义

1. 强化面向对象的知识
2. 掌握框架的原理,能够快速入门其他框架

框架有什么用?

- 数据库封装的很方便
- 常用功能如上传,缩略图很方便
- 强大的调试功能(方便的提示错误,丰富的日志)

2.0 目录搭建

```
/Base/
     App.php
     Controller.php
XBase.php
X.php
```

2.1 调试处理

App.php 接管错误处理和异常处理

```
namespace X\Base;
class App {
    public function __construct() {
        $this->initSystemHandlers();
    }

    public function initSystemHandlers() {;
        set_error_handler([$this,'handleError']);
        set_exception_handler([$this,'handleException'])
    }

    /**
    * 接管错误处理
    * 系统出错时将调用并传递4个参数,错误号,代码,行,及文件
    * 为统一处理,把错误包装成异常抛出.
    */
    public function handleError($errno,$errstr,$errfile,$errline) {
        throw new \ErrorException($errstr,$errno,$errno,$errfile,$errline);
    }

    /**
    * 接管异常处理
    *
    */
    public function handleException($exception) {
        // 禁止再处理错误或异常,防止递归
        restore_error_handler();
        restore_exception_handler();

        /* 再把交给另一个方法输出
        */
        $this->handler($exception);

    }

    /**
    * 输出异常
    */
    public function handler($exception) {
```

```
        $fileName=$exception->getFile();
        $errorLine=$exception->getLine();
        $trace = $exception->getTrace();

        if($exception instanceof \ErrorException) {
            array_shift($trace);
        }

        foreach($trace as $i=>$t) {
            if(!isset($t['file']))
                $trace[$i]['file']='unknown';

            if(!isset($t['line']))
                $trace[$i]['line']=0;

            if(!isset($t['function']))
                $trace[$i]['function']='unknown';

            unset($trace[$i]['object']);
        }

        echo '<pre>';
        print_r($trace);
        echo '</pre>';
    }
}
```

定义框架基类,调用APP

```
namespace X;
/**
 * 定义框架根目录.
 */
defined('X_PATH') or define('X_PATH', __DIR__);
/**
 * 定义DEBUG模式
 */
defined('X_DEBUG') or define('X_DEBUG', false);
class XBase {
    public static $classMap = [];
    protected static $_app = null;

    /**
    * 创建App对象
    */
    public static function app() {
        if(self::$_app === null) {
            self::$_app = new \X\base\App();
        }

        return self::$_app;
    }

    /**
    * 自动加载
    */
    public static function autoload($className) {
        if (isset(static::$classMap[$className])) {
            $classFile = static::$classMap[$className];
        }
        include($classFile);
    }
}
```

定义X类,供用户直接调用

```
define('XPATH' , __DIR__);
require(XPATH . '/XBase.php');

class X extends \X\XBase {
```

```
}
spl_autoload_register(['X','autoload']);

X::$classMap = [
'X\XBase'=>X_PATH.'/XBase.php',
'X\Base\App'=>X_PATH.'/Base/App.php',
'X\Base\Controller'=>X_PATH.'/Base/Controller.php',
];
```

2.2 路由封装

在APP中增加resolve()方法分析pathinfo

```
    /**
    * 分析pathinfo
    */
    public function resolve() {
        $path = isset($_SERVER['PATH_INFO'])?$_SERVER['PATH_INFO']:'';
        if(trim($path,'/') === '') {
            $path = [];
        } else {
            $path = explode('/', trim($path,'/'));
        }

        $path += ['Index', 'index']; // 默认控制器与默认方法
        $ac = [$path[0] , $path[1]];

        $params = array_slice($path, 2); // 切掉前2个单元,即controller/action参数

        if(count($params) & 1) { // 如果只剩下单数个参数,则补个空参数
            $params[] = '';
        }

        if($cnt = count($params)) {
            for($i = 0; $i<$cnt ; $i+=2) {
                $_GET[$params[$i]] = $params[$i+1];
            }
        }

        return $ac;
    }
```

在App中增加runController(),调用控制器

```
/**
* 调用控制器
*/
public function runController() {
    list($className,$Action) = $this->resolve();
    $c = $this->createController($className);
    var_dump($c);
    print_r($_GET);
}
```

在App中增加createController()方法,创建控制器

```
/**
* 创建控制器
*/
public function createController($className) {
    $classFile = APP_PATH . '/Controller/' . $className.'.php';
    $className = 'App\Controller\\' . $className;

    if(is_file($classFile) && !class_exists($className,false)) {
        \X::$classMap[$className] = $classFile;
    }

    if(is_subclass_of($className,'X\Base\Controller')) {
        return new $className();
```

```
    }
}
```

2.3 controller封装

```
namespace X\Base;

/**
* 框架controller的父类
*/
class Controller {
    /**
    * 存储assign过来的值
    */
    protected $_data = [];

    public function assign($key,$value) {
        $this->_data[$key] = $value;
    }

    /**
    * 把$this->_data的数据展示在模板处
    */
    public function display($file) {
        extract($this->_data);
        include(APP_PATH . '/view/' . $file);
    }

    /**
    * 创建符合框架格式的URL
    * @param $ca String 控制器/方法
    * @param $param Array ,如(cat_id=>3,page=>4)
    */
    public function createUrl($ca = 'Index/index' , $param = []) {
        $ca = explode('/' , trim($ca , '/')) + ['Index','index'];
        foreach($param as $k=>$v) {
            $ca[] = $k;
            $ca[] = $v;
        }

        return $_SERVER['SCRIPT_NAME'] . '/' . implode('/', $ca);
    }
}
```

2.4 DB类封装

model与db功能更清晰

```
namespace X\Base;
class Db extends \PDO {
    /**
    * 构造方法
    */
    public function __construct() {
        // 读配置文件
        $cfg = include(APP_PATH . '/config.php');

        $dsn = "$cfg[dbtype]:host=$cfg[host]";
        parent::__construct($dsn,$cfg['user'],$cfg['password']);

        $this->getDb($cfg['db']);
        $this->charset($cfg['charset']);
    }

    /**
    * 切库
    */
    public function getDb($db='') {
        $this->exec('use ' . $db);
```

```
    }

    /**
    * 设置字符集
    */
    public function charset($charset) {
        $sql = 'set names ' . $charset;
        $this->exec($sql);
    }

    /**
    * 获取1行数据
    */
    public function getRow($sql , $params = []) {
        return $this->cmd($sql,$params)->fetch(self::FETCH_ASSOC);
    }

    /**
    * 获取多行数据
    */
    public function getAll($sql , $params=[]) {
        return $this->cmd($sql,$params)->fetchAll(self::FETCH_ASSOC);
    }

    /**
    * 增加1条数据,并返回其主键值
    */
    public function insert($sql , $params=[]) {
        $this->cmd($sql,$params);
        return $this->lastInsertId();
    }

    /**
    * 修改1条数据,并返回影响行数
    */
    public function update($sql,$params=[]) {
        return $this->cmd($sql,$params)->rowCount();
    }

    /**
    * 删除1条数据,并返回影响行数
    */
    public function delete($sql,$params=[]) {
        return $this->cmd($sql,$params)->rowCount();
    }

    /**
    * 执行命令
    */
    protected function cmd($sql,$params) {
        $st = $this->prepare($sql);
        if($st->execute($params)) {
            return $st;
        } else {
            list(,$code,$msg) = $st->errorInfo();
            throw new \Exception($msg,$code);
        }
    }
}
```

model类修改效果

```
public function __construct() {
    $this->getTable();
    $this->db = new \X\Base\Db();
    $this->parseTable();
}
```

2.4 Model封装

- new Model时自动分析出表名
- 连接数据库
- 分析表字段+主键

```
namespace X\Base;
class Model {
    protected $tableName = '';
    protected $db = null;
    protected $pk = '';
    protected $fields = [];

    public function __construct() {
        $this->getTable();
        $this->db = new \X\Base\Db();
        $this->parseTable();
    }

    /**
    * 解析自身的表名
    */
    public function getTable() {
        $className = get_called_class();
        $this->tableName = strtolower(substr(strrchr($className, '\\') , 1,-5));
    }

    /**
    * 分析表结构,包括字段和主键
    */
    public function parseTable() {
        $sql = 'desc ' . $this->tableName;
        $struct = $this->db->getAll($sql);
        foreach($struct as $v) {
            $this->fields[] = $v['Field'];

            if($v['Key'] === 'PRI') {
                $this->pk = $v['Field'];
            }
        }
    }
```

2.5 Model完善

- add()方法 , 添加数据
- save()方法,修改数据
- delete()方法,删除数据
- find()方法, 查询数据

```
/**
* 查询一条信息
* @param $id mixed,允许传int型,理解为按主键查询
*/
public function find($id) {
    $sql = 'select * from ' . $this->tableName . ' where ' . $this->pk . ' = ?';
    return $this->data = $this->db->getRow($sql,[$id]);
}

/**
* 添加1条数据
*/
public function add($data=[]) {
    if(!empty($data) && is_array($data)) {
        $this->data = $this->_facade($data);
    }

    if(empty($this->data)) {
        throw new \Exception("Invalid data object", 500);
    }
```

```
    $sql = 'insert into %s (%s) values (%s)';
    $fields = implode("," , array_keys($this->data));
    $value = trim( str_repeat('?,', count($this->data)) , ',');
    $sql = sprintf($sql , $this->tableName , $fields , $value);

    return $this->db->insert($sql,array_values($this->data));
}

/**
* 修改数据的方法
*/
public function save($data=[]) {
    if(!empty($data) && is_array($data)) {
        $this->data = $this->_facade($data);
    }

    if(empty($this->data)) {
        throw new \Exception("Invalid data object", 500);
    }

    $sql = 'update %s set ';
    $newvalue = [];
    foreach($this->data as $k=>$v) {
        $sql .= $k . '=?,';
        $newvalue[] = $v;
    }
    $sql = trim($sql,',');
    $sql .= ' where %s=?';

    $newvalue[] = $this->data[$this->pk];

    $sql = sprintf($sql , $this->tableName , $this->pk);
    return $this->db->update($sql,$newvalue);
}

/**
* 按主键删除1条数据
*/
public function delete($id) {
    $sql = 'delete from ' . $this->tableName . ' where ' . $this->pk . '=?';
    return $this->db->delete($sql,[$id]);
}


/**
* 过滤无关的列
*/
protected function _facade($data) {
    foreach($data as $k=>$v) {
        if(!in_array($k , $this->fields)) {
            unset($data[$k]);
        }
    }

    return $data;
}
```

2.6 ActiveRecor优化Model操作

ActiveRecord

活动对象就是把:
Model类--->1张表
对象------>1行数据
对象的属性--->1行中字段的值

比如:
UserModel-->User表
$user = new UserModel--->User的1行数据

$user->email = 'xxxx'; // 即修改email列的值

$user->save();

```
    /**
    * 魔术方法,重载属性操作
    */
    public function __set($key,$value) {
        $this->data[$key] = $value;
    }

    /**
    * get操作,获取对象的属性
    */
    public function __get($key) {
        return isset($this->data[$key]) ? $this->data[$key] : NULL;
    }
```

model功能完善
添加如下方法

- field()
- where()
- group()
- having()
- order()
- limit()

```
/**
* 查询多条数据
*/
public function select() {
    $sql = $this->parseSql();
    $this->reset();
    return $this->db->getAll($sql);
}

/**
* 指定要查询的列
* @param $fields string
*/
public function field($fields) {
    $this->options['fields'] = $fields;
    return $this;
}

/**
* 拼接where条件,如果是字符串,直接拼接
* 如果是数组,则k=v and k2=v2
* @param $cond mixed string/array
*/
public function where($cond='') {
    $this->options['where'] .= ' and ' . $cond;
    return $this;
}

/**
*
*/
public function group($col='') {
    $this->options['group'] = ' group by ' . $col;
    return $this;
}

/**
*
*/
public function having($cond='') {
    $this->options['having'] = ' having ' . $cond;
    return $this;
```

```
}

/**
*
*/
public function order($col='') {
    $this->options['order'] = ' order by ' . $col;
    return $this;
}

/**
*
*/
public function limit($offset=0,$n=null) {
    if($n === null) {
        $this->options['limit'] = ' limit 0 ,' . $offset;
    } else {
        $this->options['limit'] = ' limit ' . $offset . ',' . $n;
    }
    return $this;
}

/**
* 拼接sql
*/
public function parseSql() {
    $sql = 'select %s from %s where %s %s %s %s %s';
    $sql = sprintf($sql , $this->options['fields'] , $this->tableName , $this->options['where'] , $this->options['group'] ,$th
    //echo $sql,'<br />';
    return $sql;
}


/**
* 重置options属性
*/
public function reset() {
    $this->options = array(
                    'fields'=>'*',
                    'where'=>'1',
                    'group'=>'',
                    'having'=>'',
                    'order'=>'',
                    'limit'=>''
                    );
}
```

封装常用功能类

文件上传
图片处理
分页类
验证码