# 序

## 1、适用用户

本框架适合：

有面向对象的原生编程经验，至少使用过thinkphp，对mvc模式不陌生的程序猿，有基本的linux 指令和shell脚本操作能力。

目前实现大部分实用的框架功能:

如：控制器、基本路由、数据模型、模版引擎等。

另外有重构的微信授权登录、微信二级api、redis增删查改、Pdo防注入功能。

## 2、环境

建议：php5.5、redis2.8(api环境)

## 3、待开发部分

0、除了以下内容,一般框架有的功能都有。

1、模版引擎中 未实现 if else 条件标签。

2、虽然模版引擎（view.class.php）处理后能默认压缩html和显示，但跳过了对html页面内的js css 压缩。

3、模版引擎 未实现往静态目录输出静态页。

4、模版引擎 未实现智能转换路径的路由标签。

# 1、目录说明

## 【1公用目录】

weixin\_api/

微信的基本授权目录，作为二级api，用redis做缓存，单独放在独立域名里，由计时器刷新7200秒失效的token，通过shell脚本实现。

weixin\_api/shell/

计时器刷新api/bangju/weixin/index.php 的shell 脚本。

webtest/Library/system.php

公用文件，定义了路径变量，每个控制器入口页面都需要引入。

webtest/Api/index.php

webtest/Admin/index.php

webtest/Home/index.php

webtest/ReHome/index.php

以上入口文件，都引入了system.php，并做进一步设置，控制器也在这里初始化。

webtest/Library/Commons/config.class.php

公用设置文件，所有设置的基础类，通过继承修改。

webtest/Api/Commons/apiConfig.class.php

webtest/Admin/Commons/adminConfig.class.php

webtest/Home/Commons/homeConfig.class.php

webtest/ReHome/Commons/rehomeConfig.class.php

以上设置文件 在对应的入口文件内 初始化，传递设置对象 给 控制器，也可在其他文件中调用。

webtest/Test/

基类测试目录，改写功能或功能异常时使用。

webtest/Test/database.php 数据库综合测试

webtest/Test/encrypt.php 加解密综合测试

webtest/Test/nosql.php 负载缓存综合测试

webtest/Test/third.php 第三方工具综合测试

webtest/Test/tool.php 测试工具综合测试

## 【2库类目录】

webtest/Library/Commons/ 公用类

webtest/Library/Commons/

webtest/Library/Debugs/ 测试类 包含框架测试 和 一般测试 webtest/indexTest.php 和控制器基类 中使用

webtest/Library/Https/ 调用 api/bangju/weixin/ 二级微信api 的请求类

webtest/Library/Thirds/ 第三方库类

webtest/Library/Controlers/ 控制器类

webtest/Library/Modelers/ 数据模型类 wpdb

webtest/Library/Views/ 视图类

## 【3系统目录】

webtest/Api/ 管理后台 ( 使用该框架重构的二级微信api，由于企业号只能指定和企业号应用相同的信任域名，所以api放在项目内；如果是服务号，可以单独放一个域名，特别是有支付证书的场景。)

webtest/Admin/ 管理后台 ( 使用boostrap ui 基本展示了 控制器类 和 视图类的使用 )

webtest/Home/ 前台 ( 无ui 基本展示了微信授权登录 和 创建用户的 基本功能，使用了模型类 和 二级微信api请求类 )

webtest/ReHome/ 前台 ( 与以上相同，主要是在此基础上使用了 重构的redis+pdo类 ,原理可在webtest/Test/nosql.php 测试文件中了解)

# 2、使用方法

## 【1访问入口】

如：http://wx.host.com/Admin/

或：http://wx.host.com/Admin/index.php

对应文件：(webtest为网站根目录)/Admin/index.php ，该文件加载了控制器类 和 选择补充的公用类，还有一些基本设置。

## 【2控制器】

如：[http://wx.host.com/Admin/?/base/welcome](http://xxxx.com/Admin/?/base/welcome)

或：[http://wx.host.com/Admin/index.php?/base/welcome](http://xxxx.com/Admin/index.php?/base/welcome)

对应目录：(webtest为网站根目录)/Admin/Controler/base.class.php ，(welcome)是其中的方法。

## 【3控制器类方法 中 使用视图类】

如：以上base.class.php 中的 admin 方法

先在该类头部 use \Views\view as V ; 前提是视图类在 system.php中已经加载 才能调用命名空间。

路径：/Admin/Controler/base.class.php –> admin

function admin(){

V::tamplate('index'); //路径：/Admin/Tamplate/index.html

$cArr['content']='change\_admin'; //路径：/Admin/Tamplate/change\_admin.html

V::asChangeArr($cArr);

$lArr[0]=array('aid'=>'1','admin'=>'admin1','auth'=>'超级','read'=>'说明');

$lArr[1]=array('aid'=>'2','admin'=>'admin2','auth'=>'普通','read'=>'说明');

$lArr[2]=array('aid'=>'3','admin'=>'admin3','auth'=>'普通','read'=>'说明');

$lArr[3]=array('aid'=>'4','admin'=>'admin4','auth'=>'普通','read'=>'说明');

$lArr[4]=array('aid'=>'5','admin'=>'admin5','auth'=>'普通','read'=>'说明');

V::forList('admin',$lArr); //路径：/Admin/Tamplate/change\_admin.html 替换列表标签

V::show();

}

路径：/Admin/Tamplate/index.html

加载index模版时，自动合并 include标签 指定的模版。

{{include::"include\_head"}}

{{include::"include\_nav"}}

{{include::"include\_foot"}}

路径：/Admin/Tamplate/change\_admin.html

按需要进行内嵌替换，把用不同”content”值区分的 change 标签 内嵌替换成指定模版

{{change::"content"}}

路径：/Admin/Tamplate/change\_admin.html 替换列表标签

{{forList::"admin"}}和{{/forList::"admin"}}之间的“[[key]]”将被php数组中同key的值替换。

{{forList::"admin"}}

<tr>

<th class="hidden-xs" id="[[aid]]" scope="row">[[aid]]</th>

<td>[[admin]]</td>

<td>[[auth]]</td>

<td class="hidden-xs">[[read]]</td>

<td>

<button type="button" class="btn btn-primary edit">编辑</button>

</td>

<td>

<button type="button" class="btn btn-danger">删除</button>

</td>

</tr>

{{/forList::"admin"}}

## 【4控制器类方法 中 使用数据模型类】

如：[http://wx.host.com/Home/?/weixin/index/](http://xxxx.com/Home/?/weixin/index/)

>加载Controler的文件：(webtest为网站根目录)/Home/Controler/weixin.class.php -> index

——>

use \Modelers\model as model; //model装载器

class weixin extends homeBase { //该目录下所有的控制器 都继承同目录下的homeBase.Class.php

protected static $MS;

function \_\_construct(){

parent::\_\_construct();//初始化 父类的构造函数

self::$MS=model::model('weixin');//装载对应model时，自动连接数据库

}

function test(){

//路由节点

$uri=R::get('path\_info');//获取路由结构数组 //需要在父类初始化

//序列get

$uid= I::have('uid'); //获取序列值//例子：./Home/?/weixin/test/u-[PASS\_WORD]/ts-1/uid-0/ //需要在父类初始化

//查询

$MS=self::$MS;

$whereArray['uid']=$uid;

$user=$MS->rowSelect(self::$TB->USR,'\*',$whereArray); //查询数据 //self::$TB->USR 已在父类初始化

}

}

<——

>Controler/weixin.class.php 继承的同目录下的homeBase.Class.php

——>

use \Commons\homeConfig as con; //继承 config基类 的设置类

use \Modelers\model as model; //model装载器

use \Controlers\urlSerial as I; //序列get

use \Controlers\urlRoute as R; //路由数组

class homeBase extends baseControler {

protected static $TB; //数据库表 对象：给继承类用

protected static $PA; //接口用的公用路径 对象：给继承类用

protected static $DB; //数据库库连接资料 对象 ：给继承类用

protected static $FRAME; //框架设置 对象：给继承类用

function \_\_construct(){

//公用设置 初始化

$con= new con();

self::$TB=$con->data('TABLE');

self::$PA=$con->data('PATH');

self::$DB=$con->data('DB');

self::$FRAME=$con->data('FRAME');

//model目录节点设置 和 数据库设置 初始化

model::init(self::$FRAME,self::$DB);

//序列模式设置 初始化

I::urlType(self::$FRAME);

//路由节点 初始化

R::init(self::$FRAME);

}

<——

> model::model('weixin')装载对应model时加载的Modeler文件：(webtest为网站根目录)/Home/Modeler/weixin.class.php -> index

——>

namespace Modelers;

use Modelers\baseModel;

class weixin extends homeBase {

function \_\_construct($DB){ //model::model('weixin')装载对应model时 已经传入数据库连接资料对象

parent::\_\_construct($DB);

}

}

<——

> Modeler/weixin.class.php 继承的同目录下的 homeBase.class.php

——>

namespace Modelers;

use \Commons\homeConfig as con;

class homeBase extends WpBaseModel { //可继承WpBaseModel、PdoBaseModel、RePdoModel

protected static $TB;

protected static $PA;

protected static $DB;

function \_\_construct($DB){

$con= new con();

self::$TB=$con->data('TABLE');

self::$PA=$con->data('PATH');

self::$DB=$con->data('DB');

parent::\_\_construct($DB);//初始化父数据库类

}

}

<——

# 3、其他

**其他功能可参考 webtest/Test/ 基类测试目录中 使用的技巧：**

webtest/Test/database.php 数据库综合测试

webtest/Test/encrypt.php 加解密综合测试

webtest/Test/nosql.php 负载缓存综合测试

webtest/Test/third.php 第三方工具综合测试

webtest/Test/tool.php 测试工具综合测试

**具体项目源码参考：**

https://github.com/Dencit/Act

https://github.com/Dencit/SmartRestaurant