

Power2php 开发指南

Power2php开发指南	1
前言.....	1
1.Power2php简介	1
1.1 框架特点.....	2
1.2 环境要求.....	2
1.3 技能要求.....	2
1.4 目录/文件说明.....	2
2.Power2php运行机制和原理.....	3
3.Hello word 例子.....	4
4. MVC中的控制器.....	6
5. MVC中的模型.....	7
6. MVC中的视图.....	8
7. 高级应用.....	9
7.1 Apache的 AddType的秒用	9
7.2 使用smarty模板做为视图引擎.....	10
7.3 操作数据库.....	12
7.4 划分模块.....	12

前言

现在使用 PHP 开发也有了挺长的时间,接触了不少的开发框架,同时也发现了一些开发框架很多都照搬于 JAVA 或 ROR 的 WEB 开发框架,使不少初学者对某些功能难以理解,比如 ActiveRecord.工作之余根据自己的开发经验和一些 PHP 牛人的指点创建个人认为适合 PHP 的开发框架:Power2php.当然 Power2php 是经过许多项目的考验的,而不是一个测试的 Framework.

1.Power2php 简介

Power2php是一个简单的,开源的,易扩展(升级),高效的,面向对象的轻量级PHP MVC 开发框架.遵循[GNU General Public License \(GPL\)](#) 开源协议发布. Power2php是为了简化企业级应用开发和敏捷WEB 应用开发以及为初学者摆脱传统开发模式而诞生的. Power2php会使开发变得更简单。

对于开发团队而言使用此框架不需要专门的培训。对于初学者,你可以立即尝到到MVC开发的甜头,对于中高级程序员来说,使用Power2php会使你设计出更好Framework。Power2php也适合于不想花时间去研究那些复杂Framework的开发人员。

网址: <http://power2php.sourceforge.net/>

1.1 框架特点

- 简单,是笔者见过最简单的 Framework,核心代码只有十几行,
- MVC 模式
- 基于 URL 和表单驱动的架构
- 模块化开发,很方便于大量代码的维护
- 默认使用 PHP Include,比 smarty 模板引擎快
- 可自由的加入其他类库或模板引擎,如 smarty,pear 等
- 使用 PDO 抽象库(与 pear 相似),可以很方便的操作数据库
- 没有花哨的功能,它只是一个 Framework

1.2 环境要求

Power2php 对环境没有特殊的要求(除非你还在使用 PHP3),只是如果你使用了附带的 PDO 抽象库操作数据库的话,那么你的 PHP 服务器上必须安装有 PDO

1.3 技能要求

- 面向对象基础
- 能理解 PHP Include

Power2php 虽然非常的简单,但你仍然需要有一定面向对象基础,如果你还没有掌握面向对象技术,那么强烈建议您学习此技术。

1.4 目录/文件说明

.....\config.php	开发框架的配置文件,主要一些全局常量
.....\index.php	入口文件
.....\readme.txt	说明文件
.....\core	开发框架的核心文件夹
.....\..\common.php	开发框架的核心文件,用于分发控制器操作

.....\..\..\controller.php	控制器的基类文件
.....\..\..\controller.php	模型的基类文件
.....\application	应用程序文件夹，还可以放置如缓存，日志等
.....\..\app_config.php	应用程序的配置文件，如数据库连接信息
.....\..\controls	控制器文件夹
.....\..\..\index.php	控制器文件
.....\..\..\	
.....\..\views	视图文件夹
.....\..\..\view_index.php	一些视图文件
.....\..\..\	
.....\..\models	模型文件夹
.....\..\..\saveinfo.class.php	模型文件
.....\..\..\	
.....\libs	引入的类库文件夹
.....\..\mypdo.class.php	附带的 PDO 抽象类库
.....\..\validation.php	附带的验证类库
.....\..\..\	

Power2php 的目录非常的清晰和简洁，你也可以更改为你平时所习惯的目录结构，只需更改 config.php 的配置和 core\ common.php 的分发控制代码。

2.Power2php 运行机制和原理

Power2php 运行机制和原理很简单，一句话：通过传入特定的 URL 或表单值以分发给相应的控制器中某个函数进行处理，最后将控制器处理后信息呈现于所包含的视图中。

在 WEB 开发中，主要工作无外乎是显示页面和提交表单这两个工作。因此控制器通过 URL 访问来确定显示页面，得到信息后那么只需 include(“要显示的视图”)。而处理表单只需要处理完后重定向或者输出某个信息。

为了最简单完成 MVC 模式开发的要求，在开发过程中，需要遵循 3 个规则：

- URL 中都必须带有两个参数 ,分别是 control 和 view,用于表明改 URL 将由哪一控制器及函数处理
- 表单提交中必须带有两个隐藏字段,分别是 control 和 action，功能同上
- 控制器类的类名称应该和它的文件名是相同

另外注意 linux 下区分大小写，建议统一使用小写。

下图的例子将会帮助你理解 Power2php 是如何运作的：

访问URL: `Http://localhost/framework/index.php?control=index&view=view_index`

```
<?php
session_start();
require_once(MODEL_DIR."/vote.class.php");

//显示所有的投票类(首页)的控制器类
class index extends controller {

    // 显示 投票列表 (当访问到index.php?control=index&view=view_index时,会执行下面这个函数)
    function view_index(){
        $user=$_SESSION['user'];
        $vote=new Vote();//实例化模型类
        $topic_arr=$vote->getTopic_page($offset,$pagesize,$type,$pagesize);
        if(!empty($_GET['ended'])){
            $topic_arr=$vote->getTopic_page_ByEnd($offset,$pagesize,$type,$pagesize);
        }
        if(!empty($_GET['myself'])){
            $topic_arr=$vote->getTopic_page_Myself($offset,$pagesize,$type,$pagesize,$us
    }
    include(VIEW_DIR."/index.php");//包含视图,以显示数据
}
```

上图中给 `index.php` 传入了两个参数 `control` 和 `view`, 此两个参数会交由 `./core/common.php` 文件以分发到 `index` 控制器中的 `view_index` 函数, 也就是说 `control` 和 `view` 用来指定哪一控制器和控制器中的那个函数处理。最后将处理后的数据显示到 `view` 下 `index.php` 中

同样当表单处理时, 将通过两个隐藏域以指定交由哪一控制器和函数进行处理.如下例子:

```
<FORM name=form1 action="index.php" method=post>
<INPUT type=hidden name=control value="index" >
<INPUT type=hidden name=action value="post_add_vote">
.....其他表单字段
</FORM>
```

这样这个表单将会交由 `index` 控制器中的 `post_add_vote()` 函数处理。

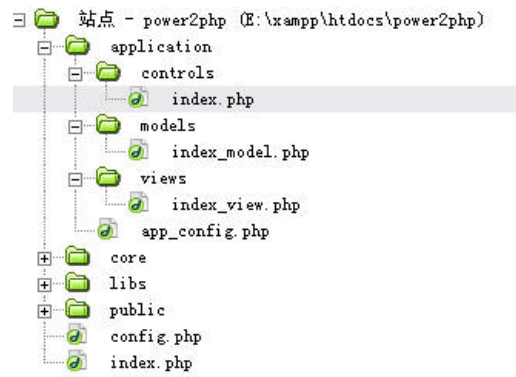
注意: 如果表单是以 `get` 方式提交的话, 那么那个 `action` 的字段名应改用

```
<INPUT type=hidden name=view value="post_add_vote">
```

3.Hello word 例子

下面演示一个最简单的例子:hello word

首先将 `Power2php` 压缩包解压到你的 `PHP` 开发目录中, 如我这里是: `E:\xampp\htdocs\Power2php`。分别在 `application` 下的 `controls` 创建 `index.php` 控制器, `models` 下创建 `index_model.php` 模型类, `views` 下创建 `index_view.php`。目录结构如下图:



其中 `models/index_model.php` 的代码:

[code]

```
<?php
//helloworld的模型类
class index_model extends Model{
    //返回字符串
    function getHelloWord(){
        return "Hello,Phper!";
    }
}
?>
```

[/code]

`controls/index.php`的代码:

[code]

```
<?php
session_start();
require(MODEL_DIR."/index_model.php" );
// helloworld的控制器类
class index extends controller {
    //显示 helloworld, 当访问到index.php?control=index&view=view_index时,
    function view_hello(){
        $model=new index_model();
        $hello=$model->getHelloWord();
        include(VIEW_DIR."/index_view.php" );
    }
}
```

[/code]

`views/index_view.php`的代码:

[code]

```
<HTML xmlns="http://www.w3.org/1999/xhtml">
<HEAD>
<TITLE><?=$hello ?> </TITLE>
<META http-equiv=Content-Type content="text/html; charset=gb2312">
```

```
</HEAD>
<BODY>
    <h1><?=$hello ?></h1>
</BODY>
</HTML>
```

[/code]

最后运行：

http://localhost/Power2php/index.php?control=index&view=view_hello

出现如下图，



提示：你可以在 config.php 设置默认的首页 URL，那么当你访问：

<http://localhost/power2php/> 时将会跳转到此URL

4. MVC 中的控制器

在 MVC 开发框架中控制一般用于处理业务逻辑，表单验证，重定向，权限控制等。在 Power2php 中控制器的角色与上相同。在 Power2php 中创建一个控制器就是创建一个 PHP 类，建议你从 core/controller.php 的控制器基类继承。

如下典型的例子：

index_control.php **注意：**此控制文件名必须和类名称相同：

[code]

```
<?php
session_start();
require(MODEL_DIR."/index_model.php") ;
// helloworld的控制器类
class index_control extends controller {
```

```
//构造函数,通常用于权限认证
```

```
function index_control() {
    if(!empty($_SESSION['user'])) {
```

```

        echo "你还没有登录呢";
        die;
    }
}

//显示首页,当访问到index.php?control=index&view=view_index时,执行以下函数
function view_index(){
    $model=new index_model();
    $hello=$model->getHelloWord();
    include(VIEW_DIR."/index.php");
}
/**
 * 注册一个用户
 * 当如下表单提交时交由此函数处理
 * <form name="form1" acion="index.php" method="POST">
 * <input type="hidden" name="control" value="index_control">
 * <input type="hidden" name="action" value="post_adduser">...
 * </form>
 */
function post_adduser(){
    $username=$_POST['user'];
    $password=$_POST['password'];
    $model=new index_model();
    $flag=$model->addUer($username,$password);
    if($flag>0){
        header("location:index.php?control=index&view=view_succ");
    }else{
        echo "不能注册此用户";
    }
}
}
}

[/code]

```

5. MVC 中的模型

在 MVC 开发框架中模型一般用于进行持久化操作。如操作数据库，写入和读写文件，访问其他接口等。在 Power2php 与控制器相似，创建一个模型其实就是创建一个类。但建议你继承于 core/model.php 中的 Model 类，如果你继承类，并且在 application/app_config.php 正确配置数据库连接，那么操作数据库将非常的简单！

如下典型例子

[code]

```

<?php
// 模型 投票 的数据库操作类
class Vote extends Model
{
    //获得所有或分类的投票主题信息，返回多条记录
    function getTopic_page($start,$end,$type,$pagesize=16 {
$sql="SELECT * FROM topic WHERE top_state !=0 limit $start,$end ";

        if(!empty($type)){
$sql="SELECT * FROM topic WHERE top_state !=0 AND top_catlog='$type'
limit $start,$end ";
        }
        return $this->db->getAll($sql); //返回多条记录

    }

    //通过主题ID获得主题信息 ,只会返回一条记录
    function getTopicByID($top_id) {
        $sql="SELECT * FROM topic WHERE top_id='$top_id' ";
        return $this->db->getRow($sql);
    }

    //条件以个答案
    function addNewSub($top_id,$answer) {
        $sql="INSERT INTO `sub_vote` ( `sub_title` , `top_id` )
            VALUES ( '$answer', '$top_id')";
        return $this->db->query($sql);
    }
}
[/code]

```

6. MVC 中的视图

在 Power2php 中，视图就是普通的 PHP 文件，为什么要使用 PHP Include 文件？个人认为使用 PHP Include 是要比模板引擎要快的，另外视图中可以使用 PHP 脚本，这样会更灵活。但建议你在开发中不要在视图中放入控制代码，控制代码应该在控制器中完成，视图最多只是显示。因此使用 `<?= $变量?>` 和 `foreach, for` 等就可以了.当然视图中也可以使用其他任何模板引擎。

7. 高级应用

7.1 Apache 的 AddType 的秒用

如果你可以配置 Apache,那么打开其配置文件,在

```
AddType application/x-httpd-php .php
```

 下一行添加

```
AddType application/x-httpd-php .htm
```

然后重新启动 apache 服务器.那么 apache 服务器将视.htm 文件做为 php 程序运行.也就是说,apache 会把.htm 进行 PHP 编译然后再输出。下面两个文件的执行结果和过程是一样的

test.htm:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>htm页面</title>
</head>
<body>
<?php
    echo "这是一个.htm文件";
?>
</body>
</html>
```

test.php:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>htm页面</title>
</head>
<body>
<?php
    echo "这是一个.htm文件";
?>
</body>
</html>
```

这样的话,如下两个函数执行结果也是一样的,因此在 Power2php 中默认是使用 PHP Include 显示视图,这样开发更具有灵活性,因为.htm 视图页面中可以执行任何的 PHP 代码:

```
function view_index(){
    $model=new index_model();
    $hello=$model->getHelloWord();
    include(VIEW_DIR."/test.php");
}
```

```

function view_index(){
    $model=new index_model();
    $hello=$model->getHelloWord();
    include(VIEW_DIR."/test.htm");
}

```

7.2 使用 smarty 模板做为视图引擎

下面演示如何使用[Smarty-2.6.16](#)模板引擎作为视图引擎

首先将[Smarty-2.6.16](#)压缩包解压到Power2php目录的libs下,

其次将 Smarty-2.6.16 目录下的 demo 的 `templates`, `configs`, `templates_c` 文件夹复制到在 Power2php 目录的 application/view 下

在 application/controls 下创建控制器名为 `smarty_demo.php`,代码:

[code]

```
<?php
```

```
require PROJECT_DIR.'/libs/Smarty-2.6.16/libs/Smarty.class.php';
```

```
// 演示 smarty 的控制器
```

```
class smarty_demo extends controller {
```

```
    var $smarty=null;
```

```
    //构造函数, 可用初始化数据
```

```
    function smarty_demo() {
```

```
        $smarty = new Smarty;
```

```
        //指定smarty的模板目录以及配置目录, 否则出错
```

```
        $smarty->template_dir=VIEW_DIR."/templates";
```

```
        $smarty->compile_dir=VIEW_DIR."/templates_c/";
```

```
        $smarty->config_dir=VIEW_DIR."/configs/";
```

```
        $this->smarty=$smarty;
```

```
    }
```

```
//显示 smarty DEMO的函数
```

```
function view_smarty(){
```

```
    //指定smarty的模板目录
```

```
    $smarty=$this->smarty;
```

```
    $smarty->compile_check = true;
```

```
    $smarty->debugging = true;
```

```
    $smarty->assign( "Name" , "Fred Irving Johnathan Bradley Peppergill");
```

```
    $smarty->assign( "FirstName" ,array( "John" , "Mry" , "Jaes" , "Henry" ));
```

```

$smartyy->assign("LastName",array("Doe","Smith","Johnon","Case"));
$smartyy->assign("Class",array(array("A","B","C","D"),array("G","H"),
    array("I","J","K","L"),array("M","N","O","P")));

    $smarty->assign("contacts",array(array("phone"=>"1","fax"=>"2",
"cell"=>"3"),
    array("phone"=>"55-4444","fax"=>"5333","cell"=>"760-1234")));

$smartyy->assign("option_values",array("NY","NE","K","IA","OK","TX"));
$smartyy->assign("option_output",
    array("New rk","Nraska","Kaas","Ia","Okoma","Teas"));
$smartyy->assign("option_selected","NE");

$smartyy->display('index.tpl');
    }

}
?>

```

[/code]

最后运行地址 http://localhost/Power2php/index.php?control=smarty_demo&view=view_smart
 结果:

The screenshot shows a web browser window with the following content:

```

phone: 1
fax: 2
cell: 3

phone: 555-4444
fax: 555-3333
cell: 760-1234

testing strip tags
This is a test

This is an example of the html_select_date function:
October 24 2008

This is an example of the html_select_time function:
08 36 32 AM
    
```

The Smarty Debug Console shows the following information:

```

Smarty Debug Console
included templates & config files (load time in seconds)
index.tpl (0.03508) (total)
test.conf [setup] global (0.00094)
  header.tpl (0.00148)
  footer.tpl (0.00371)
assigned template variables
({$Class})
Array (4)
0 => Array (4)
0 => "A"
1 => "B"
2 => "C"
3 => "D"
1 => Array (4)
0 => "E"
1 => "F"
2 => "G"
3 => "H"
2 => Array (4)
0 => "I"
1 => "J"
2 => "K"
3 => "L"
    
```

7.3 操作数据库

Power2php 默认附带了 PDO 的抽象类库，利用此类库可以非常方便操作数据库。将 application/app_config.php 正确配置数据库连接，然后去掉模型基类 core/model.php 中注释。那么当模型类继承于基类后。可以直接使用函数(与 pear 相似)
`$this->db->getAll(),$this->db->getRow(),$this->db->getOne(),$this->db->query()`
以返回相应的数据
详细可查看 libs 下的 mypdo.class.php

7.4 划分模块

当稍微大一些的项目，模块和文件将会很多，那么控制器，模型，视图文件不可能都放置于 controls,models,views 的根目录下。Power2php 很容易解决此问题。

如果要增加一个模块 new_module 那么，你需要这么做：

在 application/controls 新建一个文件夹 new_module 用于放置该模块的控制器

在 application/views 新建一个文件夹 new_module 用于放置该模块的视图

在 Power2php 跟目录创建该模块的入口文件 new_module.php，内容如下：

```
<?php
/**
 * 模块new_module的入口文件
 */
session_start();
require("config.php");
require(PROJECT_DIR."/core/common.php");

?>
```

那么访问地址将如下

http://localhost/Power2php/new_module.php?control=控制器名&view=函数名