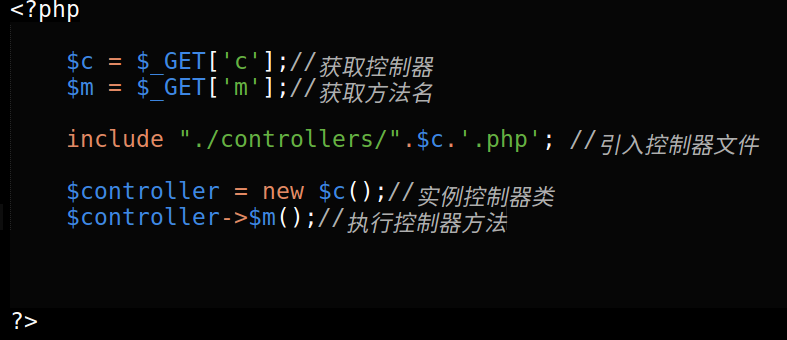
第一章、CI学习笔记

1. Ci简介
2. Mvc简介
3. 入口文件
   * 1. 在根目录下的index.php是ci搭建的项目的唯一入口，只有通过入口文件才能使用ci框架的功能。
     2. 原理是，根据访问的url以及参数，就能控制访问不同的控制器。

index.php



Admin.php



1. MVC

数据库、资源

直接访问

controll

直接访问

用户

存储数据

获取数据

表单提交

用户操作视图

加载视图

向用户展示

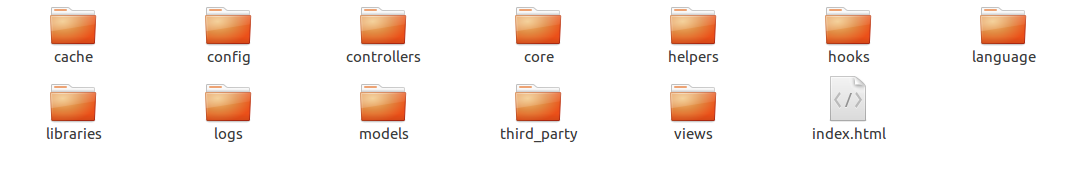
view

model

1. CI的目录
   1. 根目录



* 1. Application目录



Config:配置文件，在开发过程中的所有配置基本配置文件都在这个文件下

Controllers:控制器，所有的控制器都在这个文件夹下

Models:模型文件夹，所有的模型都在这个文件夹下

Views:视图文件夹，所有视图都在这个文件夹下

Helpers:辅助类文件夹，一些辅助类，如：表单验证、、、

Libraries:自定义类库文件加，自己定义的类库可在这个文件夹下

1. CI的访问原理
   1. 访问url使用的是pathinfo，就是访问url类似路径一样

如：localhost/study\_ci/welcome

* 1. 访问规则

主机名／根目录名／控制器名／方法名

* 1. Ci默认控制器和方法

控制器：welcome

方法：index

1. 控制器
   1. 不需要加后缀
   2. 文件名全部小写，类名要与文件名一致但是要大写
   3. 所有控制器直接或间接继承CI\_Controller类
   4. 控制器中方法的要求
      1. public
      2. 不能以\_开头
      3. 与控制器同名的方法被视为构造函数，劲量避免定义与类名相同的方法
2. 视图
   1. 在控制器中加载视图

$this->load->view(‘视图文件’);

一个控制器可以加载多个视图。

* 1. 在视图中使用原生php
  2. 向视图传递数据:数据需要封装成关联数组的形式

$name = ‘zhangsan’;

$age = 23;

$data[‘name’] = $name;

$data[‘age’] = $age;

方法一：

$this -> load->view(‘视图文件’,$data);

方法二：

$this->load->vars($data);

$this->load->view(‘视图文件’);

* 1. 在视图中遍历数组的推荐的用法

<?php foreach($arr as $key=>$val): ?>

<?= $key.”:”.$val;?>//直接使用短标签

<?php endforeach;?>

1. 加载器$this ->load是CI\_Loader的实例,system/core/Loader.php，用于加载内容的，load有很多方法
   1. view()方法：加载视图
   2. model()方法：加载模型
   3. vars()方法：为视图传递变量
   4. helper()方法：加载辅助类
   5. database()方法：加载数据库操作类
2. 输入类input,$this ->input是CI\_Input的实例,system/core/Input.php，用于处理接收数据
   1. post()方法：用于接收post提交的数据,等同于$\_POST[];
   2. server()方法：等同于$\_SERVER[];
   3. get()方法：用于接收get提交的数据,等同于$\_GET[];
3. uri类，$this ->load是CI\_URI的实例,system/core/URI.php，用于处理url的pathinfo
   1. segment()方法：用于获得url的第几个参数，从控制器开始计算个数。

例如：localhost/study\_ci/welcome/index/5/zhangsan

　$this->uri->segment(1);//welcome

$this->uri->segment(2);//index

$this->uri->segment(3);//5

$this->uri->segment(4);//zhangsan

* 1. 控制器的方法也能默认获取url参数

例如：localhost/study\_ci/welcome/index/5/zhangsan

function index($arg1=’’,$arg2=’’){

echo $arg1;//5

echo $arg2;//zhangsan

}

1. 数据库操作(一)
   1. 配置数据库
      1. 配置文件：application -> config -> database.php
      2. 常用参数：

hostname 主机名

username 用户名

password 密码

database 数据库名

dbprefix 数据库前缀

* 1. 数据库操作
     1. 加载数据库对象到超级对象的db属性里

$this->load->database();

* + 1. 执行sql语句

$sql = ‘select \* form table\_name where id=5’;

$this->db->query($sql);

* + 1. 返回结果

$sql = ‘select \* form table\_name where id=5’;

$res = $this->db->query($sql);

$res->result();返回一个数组，数组里是一个一个对象

$res->result\_array();返回一个数组，数组里是一个一个数组

1. 数据库操作(二)
   1. 自动加载数据库

application -> config -> autoload.php

$autoload[‘libraries’] = array(‘database’);

* 1. 安全问题

Ci的sql语句可以使用?代替要插入、要更新、要匹配的值例如：

$sql = ‘insert into table\_name (name,age) values(?,?)’;

$data = array(‘lili’,23);

$this->db->query($sql,$data);

这样不存在安全问题。

1. AR类:操作数据库
   1. 配置：application -> config ->database.php $active\_record = TRUE;
   2. 常见方法：　$this ->db->get();//获取数据库数据，返回的是资源结果

$this->db->result();//返回数据结果，是一个数组，数组里面是一个一个对象

$this->db->result\_array();//返回数据结果，是一个数组，数组里面也是数组

$this->db->insert(‘表名’,’关联数组’);//向数据库插入数据，关联数组的键值要与数据库字段对应,并且返回值是一个布尔值

$this ->db->updata(‘表名’,’关联数组’);//更新数据库数据，关联数组的键值要与数据库字段对应,并且返回值是一个布尔值

$this->db->rows();//返回数据结果，只返回一行

* 1. 连贯操作

使用CI的AR类的连贯操作，将下面的sql语句拆分

$sql = ‘select id,name from table\_name where id = 3 order by id desc limit 2,3’;

$res = $this->db -> select(id,name)

->from(‘table\_name’)

->where(‘id =’,3)//注意id和=之间有一个空格

->limit(3,2)

->order\_by(‘id desc’)//注意id和desc之间有一个空格

->get()->result();

* 1. where 方法
     1. 直接将字段和值作为参数

where(‘id =’,3); 或者where(‘id’,3);

* + 1. 讲一个关联数组作为参数,推荐

$arr = array(“id”=>3);

where($arr);

1. 扩展CI\_Controller

CI允许我们自己扩展控制器，首先在application ->core创建一个MY\_Controller.php的文件，

然后在文件里写入：

class MY\_Contriller extends CI\_Controller{

function \_\_construct(){

parent::\_\_construct();

}

//.........定义自己扩展的方法

}

然后我们在控制器里就可以直接继承我们自己的MY\_Controller.

1. 模型
   1. 在application -> models下建立自己的model,
   2. 模型必须继承CI\_Modle，类名首字母大写
   3. 模型里可以直接使用超级对象的属性和方法
   4. 在模型里定义自己的方法
   5. 在控制器里加载模型
      1. $this->load->modle(‘模型名’，‘模型别名’);
      2. 加载后模型就成为了超级对象的属性
      3. 调用模型方法

$this->模型名->方法名;

1. Url相关函数
   1. 加载url：$this ->load->helper(‘url’);//自动加载在autoload.php里设置
   2. site\_url(‘控制器/方法’);//用于访问控制器和方法
   3. base\_url();//用于访问项目的根目录，就是与index.php同一目录下的
   4. 区别：就是相差一个index.php，site\_url()带index.php所以可以直接访问控制器和方法，base\_url(),不可以，只能访问更目录但是也不能用于上传文件。
2. CI的分页
   1. 加载分页类：$this ->load->library(‘pagination’);
   2. 配置：
3. CI文件上传
   1. form表单的书写 <form action=”upload” method=”post” enctype=”multipart/form-data”>
   2. 配置：

$config[‘upload\_path’] = ‘./uploads/’;//不能url函数，使用相对路径，而且文件加需要手动建立

$config[‘max\_size’] = ‘1000’;//设置文件的最大值

$config[‘file\_name’] = uniqid();//设置文件名

* 1. 加载文件上传类

$this ->load ->library(‘uplad’,$config);

$this -> upload ->do\_upload(‘pic’);//执行上传

* 1. 获取文件的数据

$data = $this -> upload ->data();

1. CI的session
   1. 加载：$this -> load -> library(‘seeion’);
   2. 配置：校验字符串

Application -> config -> config.php

$config[‘enoryption\_key’] = ‘加密字符串’;

* 1. 配置是否加密

$config[‘sess\_encrypt\_cookie’] = TRUE;

* 1. 设置seeion
     1. $this ->seeion ->set\_userdata(‘key’,’value’);//value 可以是数组和对象
     2. $this ->seeion ->set\_flashdata(‘key’,’value’);//只能获取一次，然后就自动被删除
  2. 获取seeion
     1. $this ->seeion ->userdata(‘key’);
     2. $this ->seeion ->flashdata(‘key’);//只能获取一次，然后就自动被删除

1. CI隐藏入口文件

在项目的根目录下建立 .htaccess文件内容是

RewriteEngine on

RewriteCond $1 !^(index\.php|js|css|img|assets|captcha|UEditor)

RewriteRule ^(.\*)$ index.php/$1 [L]

其中第二行的括号里是在根目录的里的文件和文件夹（除ci自带的application和system文件夹）

1. CI设置路由：application -> config -> routes.php
   1. 设置默认控制器：$route[‘default\_controller’] = ‘defaultController’;
   2. 设置４０４错误页面：$route[‘404\_oberride’] = ‘404’;
   3. 自定义路由

url: localhost/controller/function

$route[‘controller/function/(:any)’] = ‘controller/function/$1’;

1. CI的验证码
   1. 加载验证码类：$this ->load ->helper(‘captha’);
   2. 在根目录创建一个存放验证码的文件夹
   3. 配置验证码

$config = array(

‘word’ => ‘XXX’,//验证码显示内容

‘img\_url’ => ‘./cpatcha’,//验证码存放路径

‘exporation’ => 6000//设置验证码过期时间

);

* 1. 创建验证码

$cap = create\_captaha($config);//创建验证码

$cap[‘image’]//img标签

$cap[‘word’];//验证码内容

1. CI的表单验证
   1. 加载表单验证类

$this ->load ->library(‘form\_validation’);

* 1. 设置规则

$this -> form\_validation ->set\_rules();

* 1. 进行验证

$bool = $this -> form\_validation ->run();

* 1. 验证后的操作

If(!$bool){

$this -> load -> view(‘form’);

}