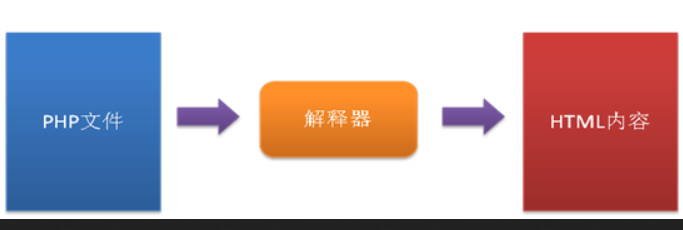
# PHP学习

1. PHP简介

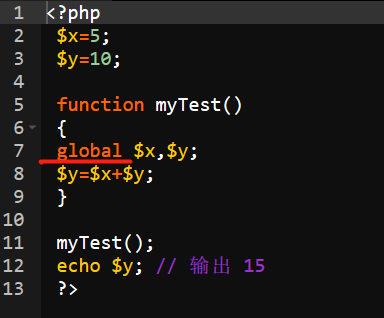
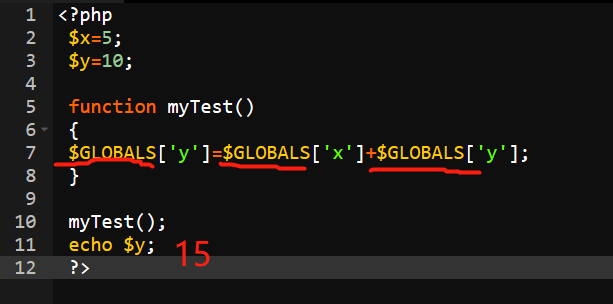
PHP是一种服务器端的，嵌入html的脚本语言。PHP 代码在服务器上执行，结果以纯 HTML 形式返回给浏览器。

PHP 能做什么及其优势？

①PHP可以生成动态内容；②PHP可以创建、打开、读取、写入、关闭服务器上的文件；③PHP可以收集表单数据；④PHP可以发送和接收cookies；⑤PHP可以添加修改删除数据库的数据；⑥PHP可以加密数据



1. PHP基本语法（php代码标志，echo和print函数，变量及内置变量，常量，var\_dump函数，并置运算符”.”，数组，foreach循环，面向对象概念，命名空间）
2. 以<?php开始 ?>结束。这是嵌入html代码的php脚本指令，里面放php代码。
3. echo ”hello” 等于print ”hello” 都是向浏览器输出文本
4. // 单行注释 /\* 多行注释 \*/
5. 变量：通过一个名字（变量名）来标识，系统为程序中的每一个变量分配一个储存单元，变量名的实质就是计算机内存单元的命名，因此，借助变量名就可以访问内存中的数据。变量是以$符号开始，区分大小写（eg：$a 和$A 是不同变量）注意：php是没有声明变量的命令的，在第一次赋值时就已经给创建了。php是一门弱类型语言，不必向php声明该变量的数据类型。在函数内部声明的变量是局部变量，只能在函数内部访问；在函数外声明的是全局变量，无法被函数内部使用，若要使用，得用global关键字修饰；使用global修饰后的全局变量都存储在一个名为$GLOBALS[index]的数组中，index保存变量名，这数组可在函数内部访问。

 等同于下图

当一个函数完成时，它的所有变量都会被删除。但有时你希望一个局部变量别被删除，这时就可以用static修饰，如下举例

**function** myTest()  
{  
 $x=0;  
 **echo** $x;  
 $x++;  
}  
  
myTest(); //函数执行完，x就被删除了  
myTest(); //再次调用，再次初始化x的值，所以x还是0  
myTest();  
  
**?>**

**function** myTest()  
{  
 **static** $x=0;  
 **echo** $x;  
 $x++;  
}  
myTest(); //函数执行完，局部变量x的值仍保存着  
myTest(); //再次调用，x++,所以x变成了1了  
myTest(); //类似上面推导  
**?>**

1. php内置系统变量，大多以下划线开头，如下：

.$GLOBALS 存储当前脚本中的所有全局变量，key为变量名，value为变量值

$GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];

.$\_SERVER 当前web服务器变量数组，包含了如头信息，路径，脚本位置等的数组

.$\_GET 存储以get方法提交表单的数据

.$\_POST 存储以post方法提交表单的数据

.$\_COOKIE 取得或设置用户浏览器cookies中存储的变量数组

.$\_FILES 存储上传文件提交到当前脚本的数据

.$\_ENV 存储当前web环境变量

.$\_REQUEST 存储提交表单中的所有请求数组，包括$\_GET，$\_POST，$\_COOKIE，$\_SESSION中的所有内容

.\_$SESSION 存储当前脚本的会话变量数组

1. var\_dump()函数返回变量的数据类型和值，比如下面的举例



还有对象数据类型，对象必须声明，用class关键字声明，类是可以包含属性和方法的结构。

也可以通过把变量值设为null，即$x=null来清空数据

1. 常量：常量的值是不可改变的，全部大写字母命名，不加$符号，常量默认是全局变量，设置常量使用define（）函数



还有系统常量（魔术常量）（都是以双下划线开头结尾）：

\_LINE\_ 当前代码所在的行号

\_DIR\_ 文件的所在目录

\_FILE\_ 当前代码所在的文件名

\_FUNCTION\_ 当前代码所在的函数名称 == \_METHOD\_

\_NAMESPACE\_ 当前命名空间的名称

\_CLASS\_ 类的名称

PHP\_VERSION 当前PHP的版本号

1. php的字符串并置运算符 . 用来连接字符串 echo $txt1 . “ and ” . $txt2;

php的strlen（）函数，用来返回字符串长度

php的strops（）函数，用来查找字符串中的指定字符，若找到则返回该指定字符在字符串中第一个匹配字符的位置，若为找到则返回false。

1. php数组：数字索引数组和关联数组2种

①数字索引数组

$cars=**array**("Volvo","BMW","Toyota");  
**echo** "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";  
**?>**

count（）返回数字的长度 echo count（$cars）;

②关联数组



③数组排序函数

sort（） 对数组进行升序排列

rsort（） 对数组进行降序排列

asort（） 根据关联数组的值，进行升序排列

ksort（） 根据关联数组的键，进行升序排列

arsort（） 根据关联数组的值，进行降序排列

krsort（） 根据关联数组的键，进行降序排列

shuffle（） 对数组中的元素进行随机排序

array\_reverse（） 讲原始数组的每个元素的顺序倒置

$age=**array**("Peter"=>"35","Ben"=>"48","Joan"=>"18"); //关联数组  
ksort($age);  
print\_r($age); //打印数组。echo不能打印数组！！！  
**?>**

使用随机排序。注意：关联数组被shuffle后键会丢失。

$age=**array**("Peter"=>"35","Ben"=>"48","Joan"=>"18"); //关联数组  
shuffle($age);  
print\_r($age); //打印数组。echo不能打印数组！！！  
**?>**

倒序

$age=**array**("Peter"=>"35","Ben"=>"48","Joan"=>"18"); //关联数组  
print\_r($age); //打印数组。echo不能打印数组！！！  
$age\_reverse = array\_reverse($age);  
**echo** "<br>";  
print\_r($age\_reverse); //打印数组。echo不能打印数组！！！  
**?>**

1. foreach（）循环

<html>  
<body>  
**<?php**$x=**array**("one","two","three");  
**foreach**($x **as** $val){  
 **echo** $val . "<br>";  
}  
**?>**</body>  
</html>

1. 函数 （PS:函数名是大小写不敏感的, 在 PHP 中，提供了超过 1000 个内建的函数。）
2. 命名空间可以解决2类问题：

①用户编写的代码与PHP内部的类、函数、常量等之间的名字冲突

②为很长的标识符名称创建一个别名的名称，提高代码可读性

1. 面向对象内容

①对象是类的实例，类是对象的抽象

②继承：继承是子类自动共享父类数据结构和方法的机制，子类可以扩展父类没有的功能，但是父类的私有属性不能继承

③多态：指的是相同的操作或函数、过程可作用有多种类型的对象上并获得不同的结果。即调用不同对象而方法名相同得重写方法产生不同的结果。

重载：方法名相同，参数类型或者个数不同

重写：方法名和参数类型或者个数都相同，但是方法内部不同

④抽象：指的是将具有一致的数据结果和行为的对象抽象成类

⑤封装：指的是将现实世界中存在的某个客体对象的属性与行为绑定在一起，并放置在一个逻辑单元中

⑥构造函数（function \_construct(){…}）：用来创建对象时初始化对象，总与new创建对象时一起使用

⑦析构函数（function \_destruct(){…}）：与构造函数相反，其作用是当对象生命周期结束则系统自动调用析构函数进行善后处理，例如清理掉建立对象时new开辟的内存空间

1. PHP表单操作

1、<form action=”跳转到哪个处理页面” name=”名称” method=”get/post” enctype=”编码”></form>表单标签，内可包含input元素，menus，textarea，fieldset，legend，label元素，表单用于向服务器传输数据。

2、<input type=”text” name=”wenbenkuang”>

3、<input type=”password” name=”mimakuang”>

4、<input type=”checkbox” name=”fuxuankuang[]” value=”1” check=”checked”>

<input type=”checkbox” name=”fuxuankuang[]” value=”2” check=”checked” >

//服务器获取的值是一个数组，checked是代表被选中，通过name放在同个组里

5、<input type=”radio” name=”danxuankuang” value=”1” check=”checked”>

<input type=”radio” name=”danxuankuang” value=”2” >

//通过name放在同个组里，服务器获取的是一个值

6、<input type=”hidden” name=”yincangyu” value=”123”> //通常用隐藏域来传递id类的值

7、<input type=”select” name=”liebiao” >

<option value=”1”>选项一</option>

<option value=”2” selected=”selected”>选项二</option>

<option value=”3”>选项三</option>

</select>

//选中哪个就获取哪个option的value值，

<select multiple=”multiple”>可以控制是否多选，size属性还可以控制显示的条目数

8、<textarea name=”wenbenyu”>1234</textarea>

文本域和列表一样是双标签，可以通过cols和rows属性控制文本域的可见列和行数，控制宽高

9、<input type=”file” name=”wenjianyu”> //可通过变淡上传文件，但一定要在form标签

中加入属性<form enctype=”multiplart/form-data”>，之后我们就可以通过$\_FILES这个全局变量来取得上传文件的信息

10、<input type=”reset” value=”重置”> //重置按钮，用它来恢复变淡的初始值

<input type=”submit” value=”提交”> //提交按钮，用它来提交表单信息

11、表单验证小案例运用的知识点

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>表单验证</title>  
 <style type="text/css">  
 .error{  
 color:red;  
 }  
 </style>  
</head>  
<body>  
**<?php**//定义各个变量并默认为空值  
$name = $email = $url = $beizhu = $sex = "";  
$nameerr = $emailerr = $urlerr = $sexerr = "";  
 **if**($\_SERVER["REQUEST\_METHOD"]=="POST"){  
 **if**(**empty**($\_POST["$name"])){  
 $nameerr = "名字是必需的";  
 }**else**{  
 //获取name的值  
 $name = test\_data($\_POST["$name"]);  
 //检测名字是否只包含字母和空格  
 **if**(!preg\_match("/^/^[a-zA-Z ]\*$/",$name)){  
 $nameerr = "名字只允许字母和空格";  
 }  
 }  
  
 **if**(**empty**($\_POST["$email"])){  
 $emailerr = "邮箱是必需的";  
 }**else**{  
 $email = test\_data($\_POST["$email"]);  
 //检验邮箱是否合法  
 **if**(!preg\_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email)){  
 $emailerr = "非法邮箱格式";  
 }  
 }  
  
 **if**(**empty**($\_POST["$url"])){  
 $url = "";  
 }**else**{  
 //获取url的值  
 $url = test\_data($\_POST["$url"]);  
 //检验url地址是否合法  
 **if**(!preg\_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~\_|!:,.;]\*[-a-z0-9+&@#\/%=~\_|]/i",$url)){  
 $urlerr = "非法的url地址";  
 }  
 }  
  
 **if**(**empty**($\_POST["$beizhu"])){  
 $beizhu = "";  
 }**else**{  
 $beizhu = test\_data($\_POST["$beizhu"]);  
 }  
  
 **if**(**empty**($\_POST["$sex"])){  
 $sexerr = "性别是必需的";  
 }**else**{  
 $sex = test\_data($\_POST["$sex"]);  
 }  
 }  
 //对提交的数据进行去除空格，换行，制表符，反斜杠，和避免script代码注入  
 **function** test\_data($data){  
 $data = trim($data);  
 $data = stripcslashes($data);  
 $data = htmlspecialchars($data);  
 **return** $data;  
 }  
**?>**<h2>PHP验证表单</h2>  
<p class="error">\*号为必需字段</p>  
<form method="post" action="**<?php echo** htmlspecialchars($\_SERVER["PHP\_SELF"]);**?>**">  
 名字：<input type="text" name="name" value="**<?php echo** $name; **?>**">  
 <span class="error">\***<?php echo** $nameerr;**?>**</span>  
 <br><br>  
 E-mail：<input type="text" name="email" value="**<?php echo** $email;**?>**">  
 <span class="error">\***<?php echo** $emailerr;**?>**</span>  
 <br><br>  
 网址：<input type="text" name="url" value="**<?php echo** $url; **?>**">  
 <span class="error">**<?php echo** $urlerr;**?>**</span>  
 <br><br>  
 备注：<br>  
 <textarea name="beizhu" rows="5" cols="40" value="**<?php echo** $beizhu;**?>**"></textarea>  
 <span class="error">**<?php echo** $beizhu;**?>**</span>  
 <br><br>  
 性别：<input type="radio" name="sex" **<?php if**(**isset**($sex) && $sex=="female") **echo** "checked";**?>** value="female">女  
 <input type="radio" name="sex" **<?php if**(**isset**($sex) && $sex=="male") **echo** "checked";**?>** value="male">男  
 <span class="error">\***<?php echo** $sexerr;**?>**</span>  
 <br><br>  
 <input type="submit" value="提交" name="Submit">  
</form>  
**<?php  
echo** "<h2>您输入的内容是：</h2>";  
**echo** $name;  
**echo** "<br>";  
**echo** $email;  
**echo** "<br>";  
**echo** $url;  
**echo** "<br>";  
**echo** $beizhu;  
**echo** "<br>";  
**echo** $sex;  
**echo** "<br>";  
**?>**</body>  
</html>

$\_SERVER[“PHP\_SELF”]是一种超全局变量，返回当前执行脚本的文件名，于是可以在表单中嵌入php代码实现提交数据发送到页面自身，这样就能在表单页面获得错误信息了。不过这个变量会被黑客利用，因为可以通过<script>标签添加JavaScript代码，把用户重定向到另一台服务器上的某个文件，以此来保存用户提交的数据信息，不过可通过下面的方法就可以防止黑客script注入。

htmlspecialchars()函数是把特殊字符转换为html实体，即把<和>之类的html字符转化为&lt;&gt;这样做的好处可以防止攻击者通过表单注入script代码

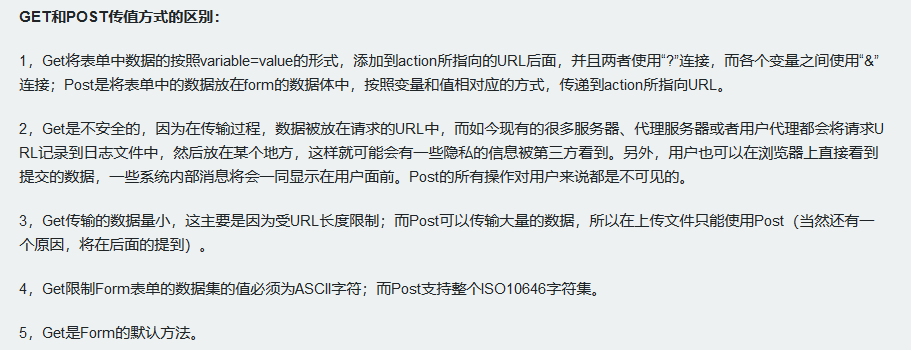
12、正则表达式







1. get和post请求的区别



1. PHP进阶（多维数组，date函数，包含语句，读写文件，文件上传，cookie，session，发送邮件，过滤器，json）
2. 多维数组：超过一维都可以称为多维数组，实际上，数组的每个元素也可以是另一个数组，比如一维数组中的数组元素是另一个数组，这样就变成了二维数组。

**<?php**$cars = **array**(  
 **array**("xiaojiang","110","160"),  
 **array**("kaikai","110","160"),  
 **array**("laozheng","110","160")  
);  
**for**($i=0;$i<count($cars);$i++){  
 **for**($j=0;$j<count($cars[$i]);$j++){  
 **echo** $cars[$i][$j] . "<br>";  
 }  
}  
**?>**

数组内部嵌套多层数组，再格式化打印，选取数组中的某个值输出



1. date（）日期函数

**<?php***/\*\*  
 \* Created by PhpStorm.  
 \* User: 小江  
 \* Date: 2018/11/1  
 \* Time: 12:01  
 \*/***echo** "date()函数的运用举例" . "<br>";  
**echo** date("l") . "<br>"; //小写字母L，表示当前周几  
**echo** date("Y年m月d日 H时i分s秒") . "<br>";  
**echo** date("Y-m-d",strtotime("2018-10-24 18:26:18")) . "<br>"; //指定日期格式化输出  
**echo** date("Y-m-d",strtotime("-1 days")) . "<br>"; //当前日期前一天的日期  
**echo** date("Y-m-d",strtotime("-1 weeks")) . "<br>"; //当前日期前一周的日期  
**echo** date("Y-m-d",strtotime("-1 months ")) . "<br>"; //当前日期前一月的日期  
**echo** date("Y-m-d H",strtotime("next Thursday")) . "<br>"; //下个周四的日期  
**echo** date("Y-m-d",strtotime("2018-10-24 -3 days")) . "<br>"; //指定日期的前3天  
**?>**

附加：一些参数格式设置表格如下



1. include和require语句

include语句和require语句除了处理错误的方式不同外，在其他方面都是相同的。

（即require生成一个致命错误，在错误发生后脚本停止运行，而include生成一个警告，在错误发生后脚本仍会运行。）于是，一般使用include包含（eg：include “filename”;）

导入相对路径（include \_DIR\_.”\foot.php”;）

include\_once和require\_once主要的作用是避免重复包含

1. 文件处理

数据再服务器上的存取一般都是数据库或者文件，目前大多使用数据库来读写数据，文件也有用到（比如记录错误日志，导出数据到文件等）

写文件3个步骤：①fopen（）打开文件，②fwrite（）写入文件，③fclose（）关闭文件

eg：<?php $file = fopen(“welcome.txt”,”r”); ?> 若无法打开文件则返回0，下面是模式参数



写文件代码：

//打开文件,只写  
$yourfile = fopen("formTest.txt","w") **or exit**("Unable to open file!");  
//写文件  
$txt = "xiaoqiong\n";  
fwrite($yourfile,$txt);  
$txt = "xiaojiang\n";  
fwrite($yourfile,$txt);  
fclose($yourfile); //关闭文件

读文件代码：

//打开文件,只读  
$yourfile = fopen("formTest.txt","r") **or exit**("Unable to open file!");  
//读文件  
**echo** fread($yourfile,1024); //不用使用循环便可读取整个文件，第二个参数代表最大字节数，必需的

逐行读文件代码：

//逐行读文件  
$myfile = fopen("formTest.txt","r") **or exit**("Unable to open file");  
//feof（）函数检测是否为文件末尾  
**while** (!feof($myfile)){  
 //fgets（）函数读一行后文件指针对跳到下一行  
 **echo** fgets($myfile) . "<br>";  
}

逐字符读文件代码：

//逐字符读文件  
$myfile = fopen("formTest.txt","r") **or exit**("Unable to open file");  
//feof（）函数检测是否为文件末尾  
**while** (!feof($myfile)){  
 //fgetc（）函数读一字符后文件指针对跳到下一个字符  
 **echo** fgetc($myfile) . "<br>";  
}

1. 文件上传

通过一个html表单+php上传代码实现文件上传的小功能

**<?php***/\*\*  
 \* Created by PhpStorm.  
 \* User: 小江  
 \* Date: 2018/11/3  
 \* Time: 15:55  
 \*/*// 允许上传的图片后缀  
$allowedExts = **array**("gif", "jpeg", "jpg", "png");  
$temp = explode(".", $\_FILES["file"]["name"]);  
**echo** $\_FILES["file"]["size"];  
$extension = end($temp); // 获取文件后缀名  
**if** ((($\_FILES["file"]["type"] == "image/gif")  
 || ($\_FILES["file"]["type"] == "image/jpeg")  
 || ($\_FILES["file"]["type"] == "image/jpg")  
 || ($\_FILES["file"]["type"] == "image/pjpeg")  
 || ($\_FILES["file"]["type"] == "image/x-png")  
 || ($\_FILES["file"]["type"] == "image/png"))  
 && ($\_FILES["file"]["size"] < 204800) // 小于 200 kb  
 && in\_array($extension, $allowedExts)) {  
 **if** ($\_FILES["file"]["error"] > 0) {  
 **echo** "错误：: " . $\_FILES["file"]["error"] . "<br>";  
 } **else** {  
 **echo** "上传文件名: " . $\_FILES["file"]["name"] . "<br>";  
 **echo** "文件类型: " . $\_FILES["file"]["type"] . "<br>";  
 **echo** "文件大小: " . ($\_FILES["file"]["size"] / 1024) . " kB<br>";  
 **echo** "文件临时存储的位置: " . $\_FILES["file"]["tmp\_name"] . "<br>";  
  
 // 判断当期目录下的 upload 目录是否存在该文件  
 // 如果没有 upload 目录，你需要创建它，upload 目录权限为 777  
 **if** (file\_exists("upload/" . $\_FILES["file"]["name"])) {  
 **echo** $\_FILES["file"]["name"] . " 文件已经存在。 ";  
 } **else** {  
 // 如果 upload 目录不存在该文件则将文件上传到 upload 目录下  
 move\_uploaded\_file($\_FILES["file"]["tmp\_name"], "upload/" . $\_FILES["file"]["name"]);  
 **echo** "文件存储在: " . "upload/" . $\_FILES["file"]["name"];  
 }  
 }  
} **else** {  
 **echo** "非法的文件格式";  
}  
**?>**

1. PHP的cookie

cookie常用来识别用户，是由服务端生成，发送给User-Agent（一般是浏览器），浏览器将cookie的key/value保存到某目录的文本文件中，下次请求同一网站时就发送该cookie给服务器。

setcookie（）函数用于设置cookie，注意：setcookie（）必须在<html>标签之前

php读取cookie，使用超全局变量$\_COOKIE数组读取到当前客户端中保存的cookie，（比如print\_r($\_COOKIE)或者echo $\_COOKIE[‘cookie名’]），浏览器对cookie的数量是有限制的，一般为不超过50个，大小不超过4k左右。注意：重要数据不能存在cookie中，要保存还是得靠数据库。不同电脑无法共享cookie，因为cookie存放的位置是在本机中，同一台电脑的不同浏览器也无法共享cookie，甚至同一个浏览器，同一个域名下，不同路径的cookie也无法实现共享。

$\_COOKIE[“user”]取得cookie， print\_r($\_COOKIE)取得cookie所有值

慢半拍 实际开发往往setcookie后，直接通过$\_COOKIE去获取数据，这时是无法获得刚刚设置的cookie的值的，因为cookie实际是保存在浏览器中，只有当前页面返回到浏览器后，setcookie的值才会保存到浏览器中，而$\_COOKIE中的数据是每次页面请求的时候从客户端带给服务器的，所以这就是为什么当时设置当时取不到的原因。

1. PHP的session

Session是用于存储关于用户变量的信息或更改用户会话的设置，对于应用程序中的所有页面的可用。（比如你在计算机操作某个应用程序，做些修改，然后关闭，这就像一次会话，计算机都知道。）

session工作机制：为每个访客创建一个唯一的UID，并基于这个UID来存储变量，UID存储在cookie中或者通过URL进行传导。注意：session\_start（）必须在<html>标签之前

session\_start(); //开启session  
//判断是否有session  
**if**(**isset**($\_SESSION['views'])){  
 $\_SESSION['view'] = $\_SESSION['views']+1;  
}**else**{  
 $\_SESSION['views'] = 1;  
}  
**echo** "浏览量：" . $\_SESSION['views'];

session\_destroy（）删除session数据； unset（）用于释放指定的session变量

1. 发送电子邮件

PHP允许你从脚本直接发送电子邮件 通过mail（）函数

<html>  
<head>  
 <meta charset="utf-8">  
 <title> php中文网(php.cn)</title>  
</head>  
<body>  
**<?php  
function** spamcheck($field){  
 //filter\_var() 过滤 e-mail  
 // 使用 FILTER\_VALIDATE\_EMAIL  
 **if**(filter\_var($field, *FILTER\_VALIDATE\_EMAIL*)) {  
 **return TRUE**;  
 } **else** {  
 **return FALSE**;  
 }  
}  
**if** (**isset**($\_REQUEST['email'])) {  
 // 如果接收到邮箱参数则发送邮件  
 // 判断邮箱是否合法  
 $mailcheck = spamcheck($\_REQUEST['email']);  
 **if** ($mailcheck==**FALSE**) {  
 **echo** "非法输入";  
 } **else** {  
 // 发送邮件  
 $email = $\_REQUEST['email'] ;  
 $subject = $\_REQUEST['subject'] ;  
 $message = $\_REQUEST['message'] ;  
 mail("someone@example.com", "Subject: $subject",  
 $message, "From: $email" );  
 **echo** "Thank you for using our mail form";  
 }  
}  
**else** {  
 // 如果没有邮箱参数则显示表单  
 **echo** "<form method='post' action='mailform.php'>  
 Email: <input name='email' type='text'><br>  
 Subject: <input name='subject' type='text'><br>  
 Message:<br>  
 <textarea name='message' rows='15' cols='40'>  
 </textarea><br>  
 <input type='submit'>  
 </form>";  
}  
**?>**</body>  
</html>

1. PHP过滤器

过滤器用于验证过滤来自非安全来源的数据，比如用户的输入。

一般用于过滤的外部数据（比如表单的输入数据，cookie，web services data，服务器变量，数据库查询结果等）

过滤器函数：filter\_var() 通过一个指定的过滤器来过滤单一的变量

filter\_var\_array() 通过相同的或不同的过滤器来过滤多个变量

$num = 123;  
**if**(!filter\_var($num,*FILTER\_VALIDATE\_INT*)){ //第二个参数就是过滤器  
 **echo** "不是一个合法的整数";  
}**else**{  
 **echo** "是一个合法的整数";  
}

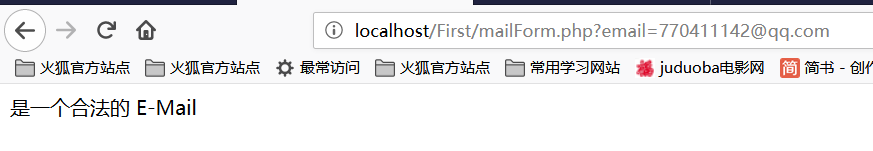
选项和标志用于向指定的过滤器添加额外的过滤选项，不用的过滤器有不同的选项和标志

$var=300;  
//选项  
$int\_options = **array**(  
 "options"=>**array** (  
 "min\_range"=>0,  
 "max\_range"=>256  
 )  
);  
**if**(!filter\_var($var, *FILTER\_VALIDATE\_INT*, $int\_options))  
{  
 **echo**("不是一个合法的整数");  
}  
**else**{  
 **echo**("是个合法的整数");  
}

验证get请求方式发送表单邮箱是否为一个有效的email

**if**(!filter\_has\_var(*INPUT\_GET*, "email")) {  
 **echo**("没有 email 参数");  
} **else** {  
 **if** (!filter\_input(*INPUT\_GET*, "email", *FILTER\_VALIDATE\_EMAIL*)) {  
 **echo** "不是一个合法的 E-Mail";  
 } **else** {  
 **echo** "是一个合法的 E-Mail";  
 }  
}

运行截图：



高级过滤器

$int = 122;  
$min = 1;  
$max = 200;  
  
**if** (filter\_var($int, *FILTER\_VALIDATE\_INT*, **array**("options" => **array**("min\_range"=>$min, "max\_range"=>$max))) === **false**) {  
 **echo**("变量值不在合法范围内");  
} **else** {  
 **echo**("变量值在合法范围内");  
}

1. PHP之json

json\_encode()对变量进行json编码，成功则返回json数据，否则false

$arr = **array**('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5);  
**echo** json\_encode($arr);

json\_decode()对json格式的字符串进行解码，并转换为PHP变量

$json = '{"a":1,"b":2,"c":3,"d":4,"e":5}';  
//解码  
var\_dump(json\_decode($json));  
//assoc为true时返回数组，false返回对象  
var\_dump(json\_decode($json, **true**));

1. require\_once与require其实没多大区别，都是引入外部文件，唯一区别是require\_once会检查该文件是否已经被包含，是的话就不会再包含进来，即加载文件一次。如下使用：

**require\_once** *\_\_DIR\_\_*."/conf.php";

1. PHP之mysql数据库操作
2. mysql简介：mysql是一种在服务器上运行的数据库系统，非常快速可靠，免费开源。是一种关系型数据库（类似于二维数组的概念，即关系型数据库就是一组二维表组成的数据仓库），数据库：一组数据表的集合
3. thinkPHP框架的开发学习

前提：运行php程序前一定要开启wampserver集成开发环境。



1、thinkPHP5.1下载与安装

去官网下载对应thinkPHP框架版本，然后就可以直接应用使用啦，新建一个项目时把这个thinkPHP框架的文件给复制过去就好了。





1. thinkPHP5.1的目录结构



1. 调试模式

thinkPHP5以上是默认打开调试模式的。当开发完成后，项目部署时，修改application里的config.php文件，app\_debug配置参数改为：’app\_debug’ => false

注意：通过控制器类里的protected和private方法是无法通过URL直接访问的，只有public类型的方法才可以

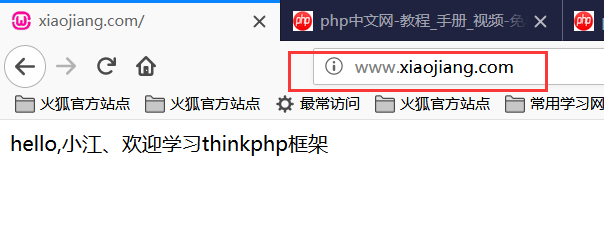
1. 告别localhost+目录的时代，创建虚拟域名



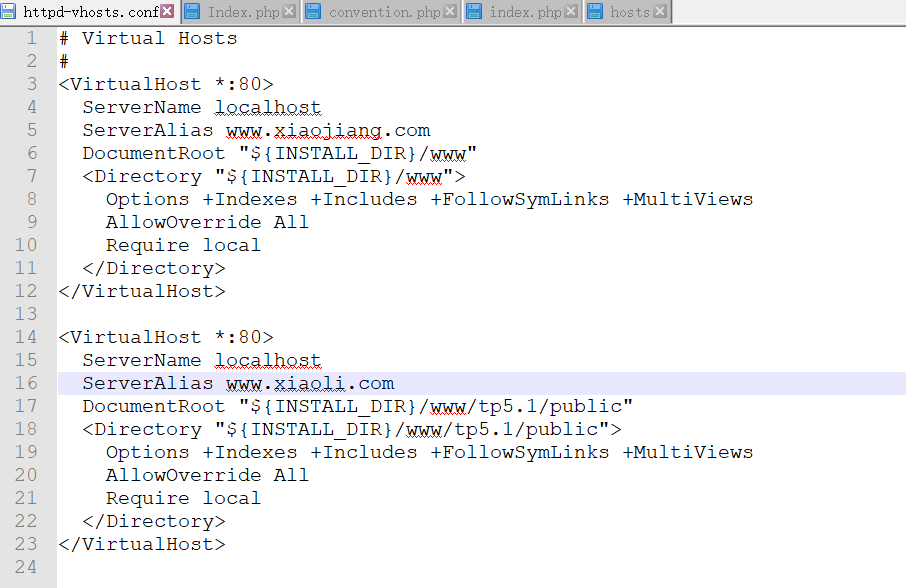


修改虚拟主机配置文件，实现访问虚拟域名就可以访问项目了，虚拟主机其实也就是和localhost形成了映射，实现虚拟主机时一样也就是在访问localhost罢了





配置多个虚拟主机（即不同项目使用不同虚拟主机来访问。不过目前我还没解决设置虚拟主机为项目public目录下的入门文件，而由于项目中有引入外部资源文件，比如bootstrap，css等资源无法加载的问题，所以下图有外部资源的项目我还是设置为网站根目录www）

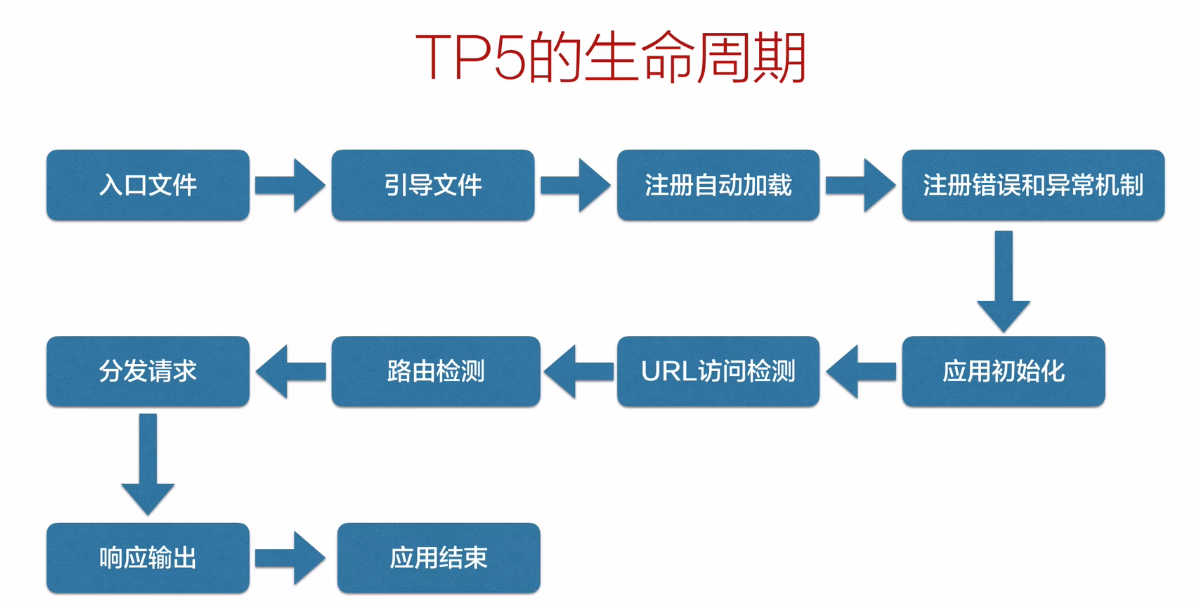


1. 架构总览，生命周期，URL访问

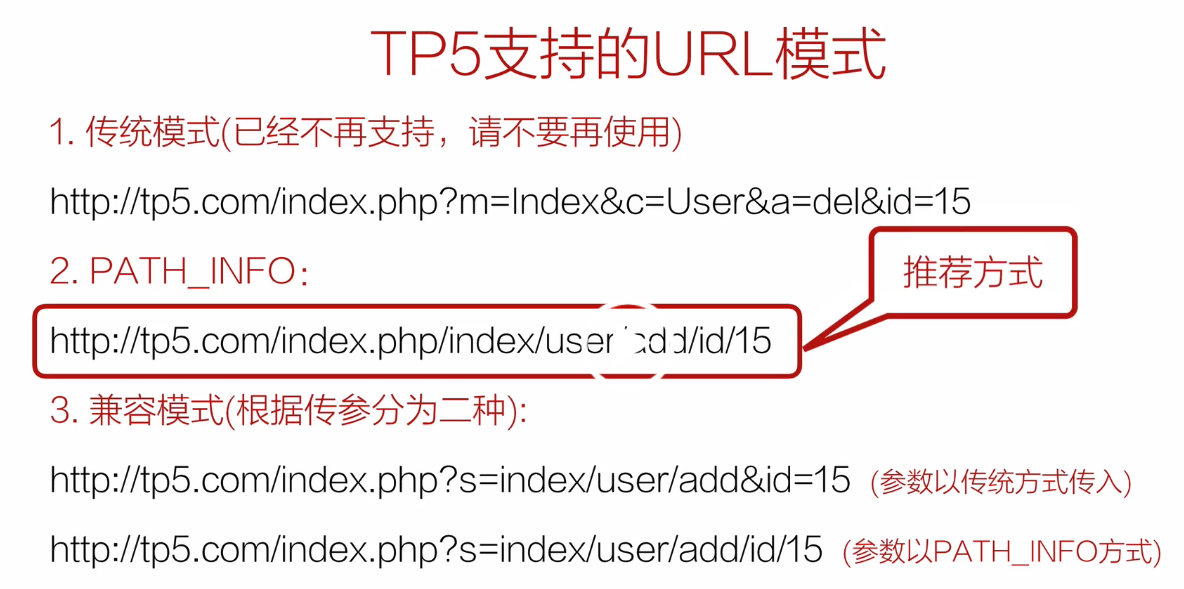


URL访问默认采用PATH\_INFO方式，即<http://域名/入口文件/模块/控制器/操作>



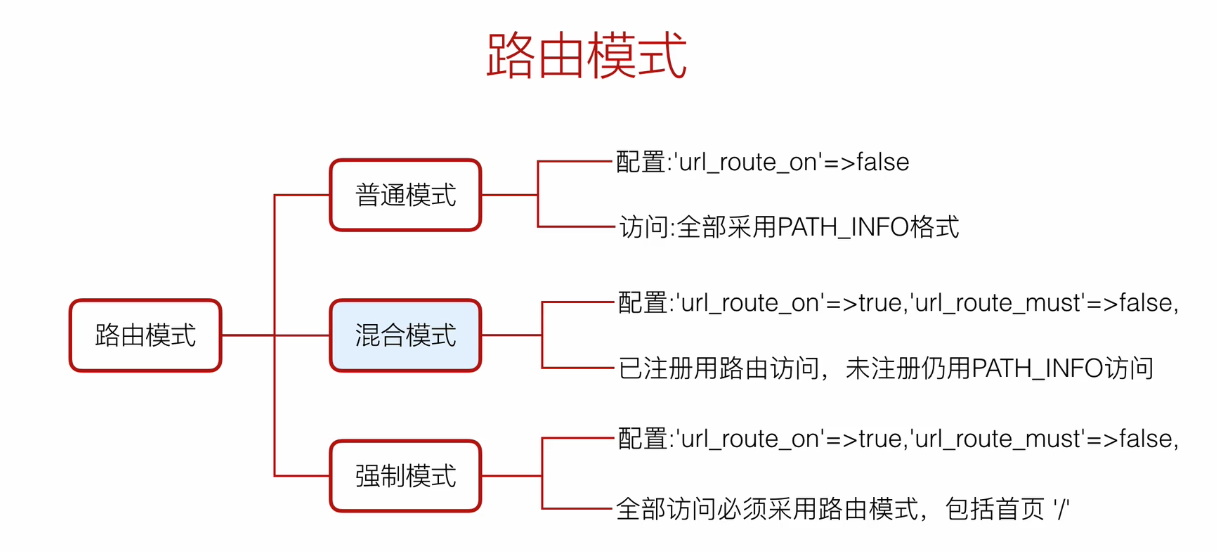


1. 入口文件主要2个任务：①定义常量；②加载引导文件

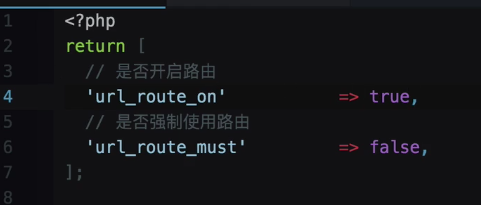


1. 路由（本质就是用来简化URL）

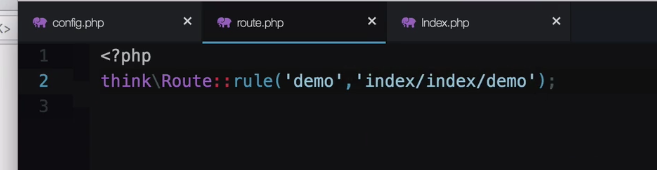
注意：路由配置不支持在模块配置文件中设置

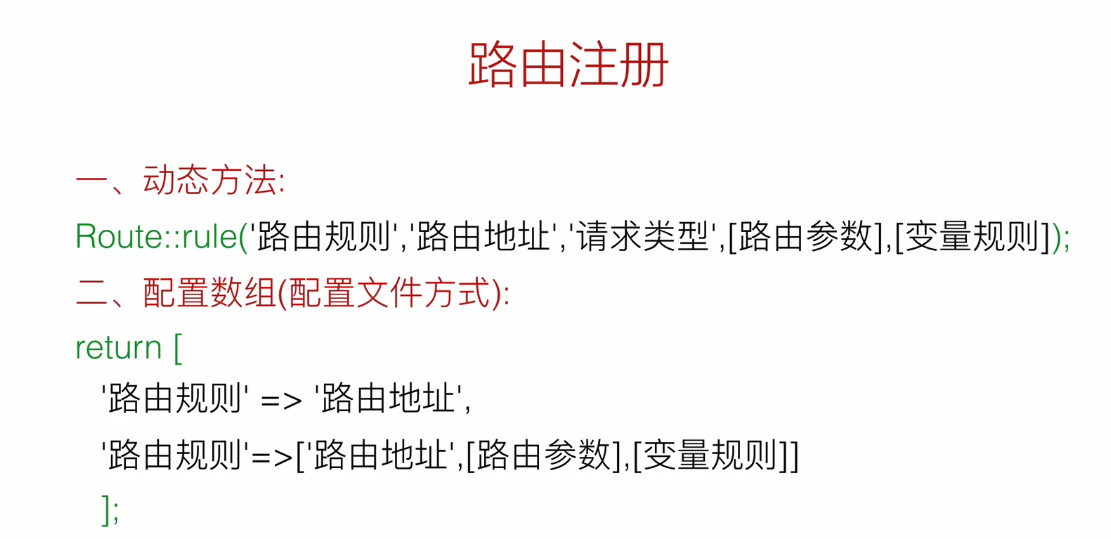


自定义路由规则后，原来的URL地址就会失效，变成非法请求

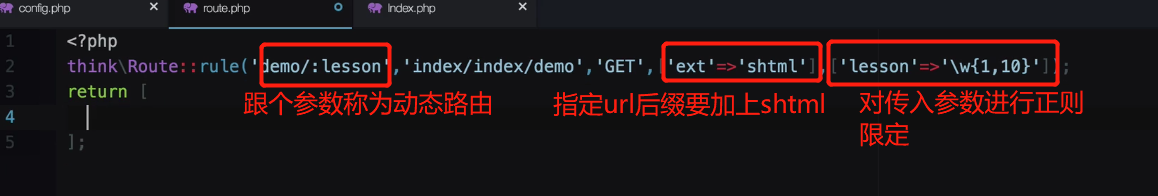


创建使用路由规则

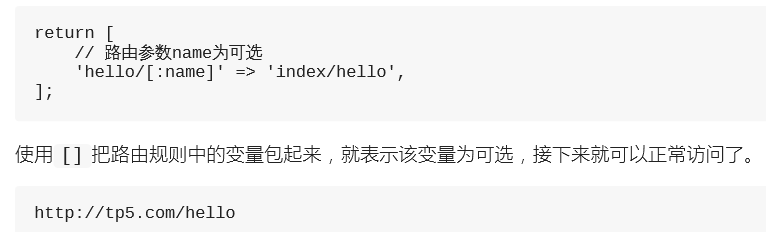




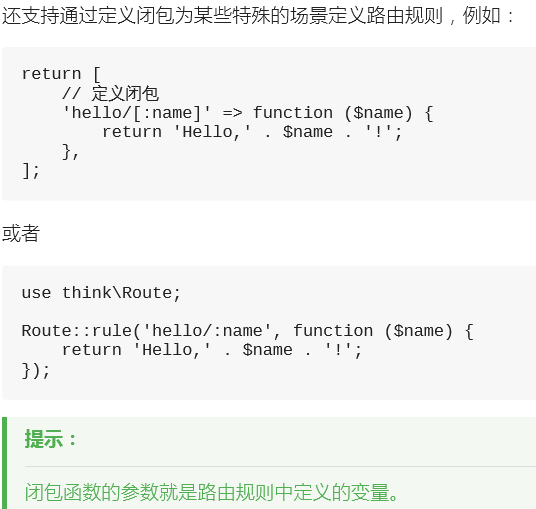
动态注册如下：



配置数组注册如下：



闭包路由



路由分组



生成URL地址



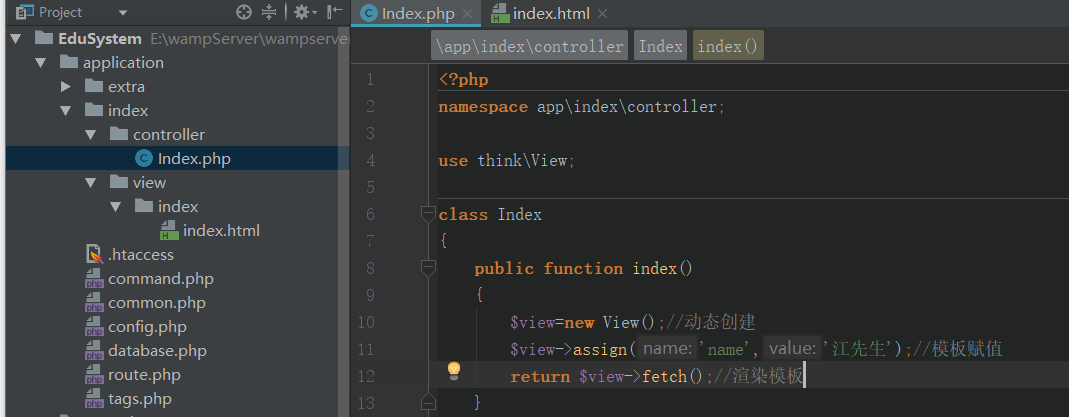
1. 关于thinkphp如何实例化视图的三种方法

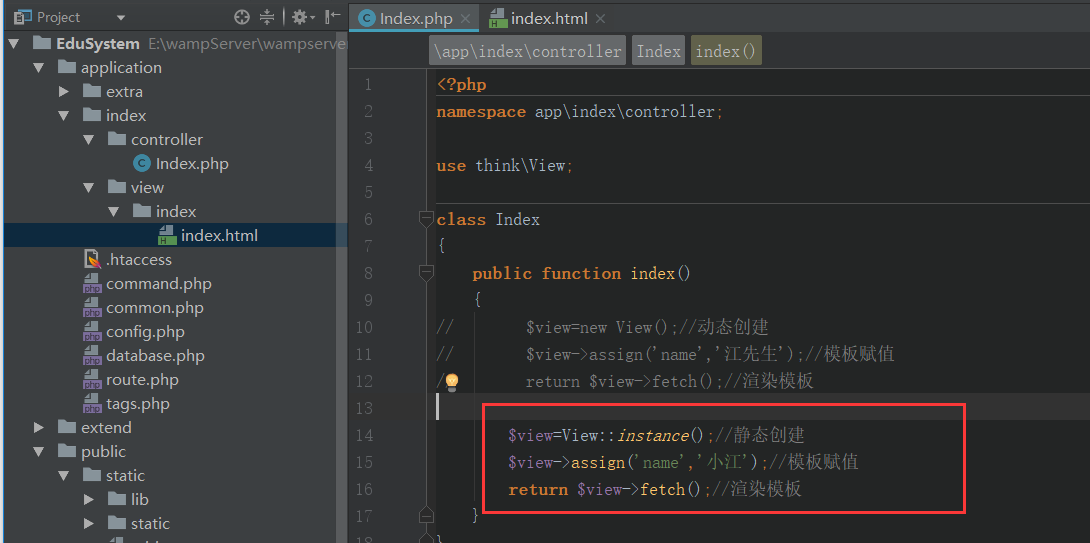
Tips：fetch()渲染模板，assign()模板赋值，display()渲染内容

①直接实例化视图类

分为动态创建（即new View()）和静态创建（即View::instance()）

下面进行代码演示





②继承控制器基类Controller



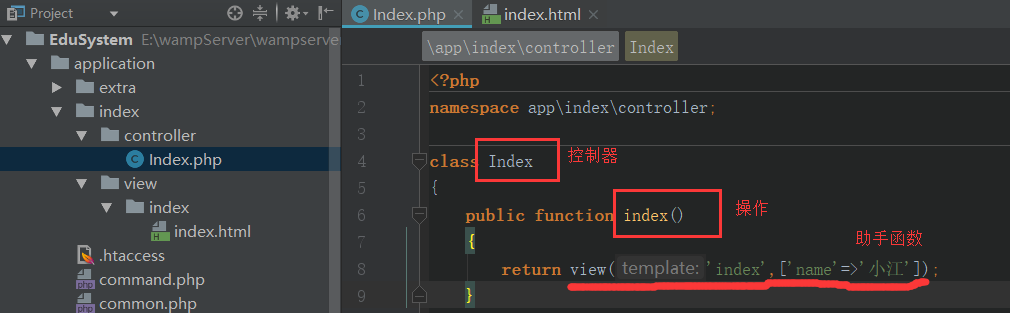
③使用View()助手函数

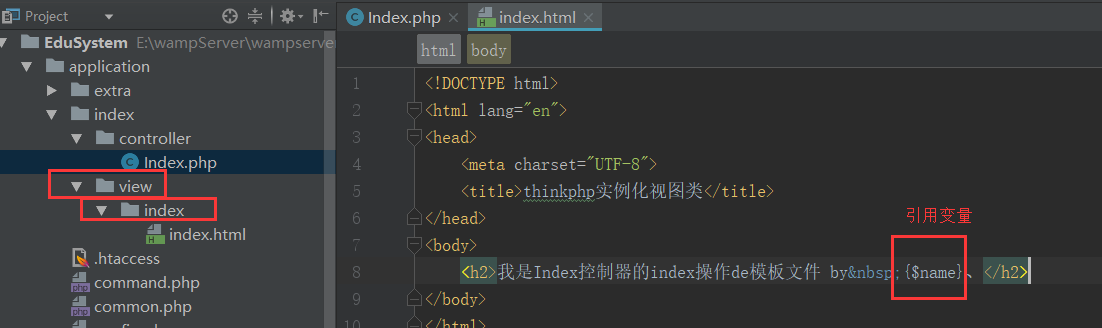
优势是不需要继承基类Controller，也不用实例化视图类

劣势是仅适合于模板渲染，功能单一，不能加载配置

语法：view（[模板文件],[模板变量数组],[模板替换数组]）

eg:view([index],[‘name’=>’小江’],[‘\_\_SITE\_\_’=>’php.cn’]);



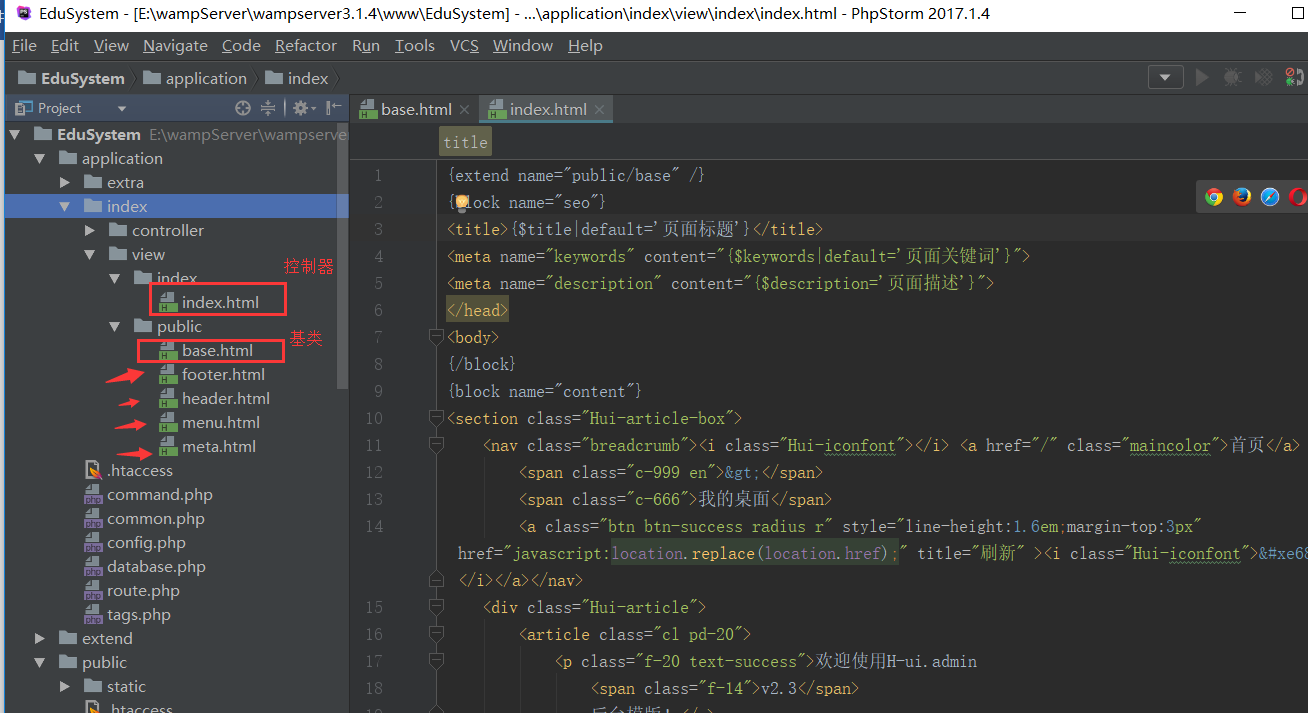


1. 模板分离

把页面拆分成几个模块页面，使用{include name=”...” /}包含进来，

使用新建一个基础模块让index模块去继承它，在基础base模块中只能包含{block name=”...” }...{/block}标签和{include file=”...” /}

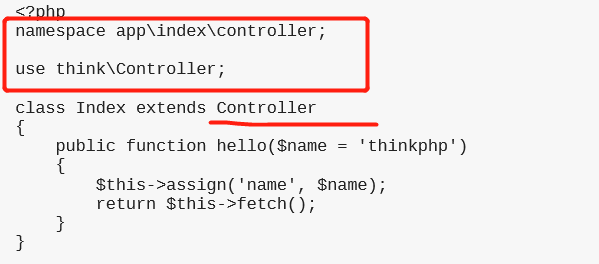
index模块中只能包含{extend name=”public/base” /}标签和{block name=”...”}{/block}



{load href=”引入的css或者js外部文件”}标签

10、use用法和命名空间的解释

在tp5中，往往根据类的命名空间来快速定位文件位置，在目录中，app命名空间指的是文件的起始目录application，think命名空间指的是thinkphp/library/think，后面的命名空间则表示起始目录到子目录。



使用use来导入一个命名空间的类库，这样就可以使用别名来替代完整命名空间路径访问类库。如果不使用use导入的话，继承就得这样写成class Index extends \think\Controller

默认情况下URL的访问是不区分大小写的，这样会使驼峰式命名的控制器访问不是很正常，比如class HelloWorld ，在URL访问时得写hello\_world才能正常访问，直接URL输入HelloWorld是无法正常访问的。若要区分大小写访问，可以在应用配置文件设置 ‘url\_convert’ => false 来关闭URL自动转换 。区分大小写后访问控制器时URL就得HelloWorld了

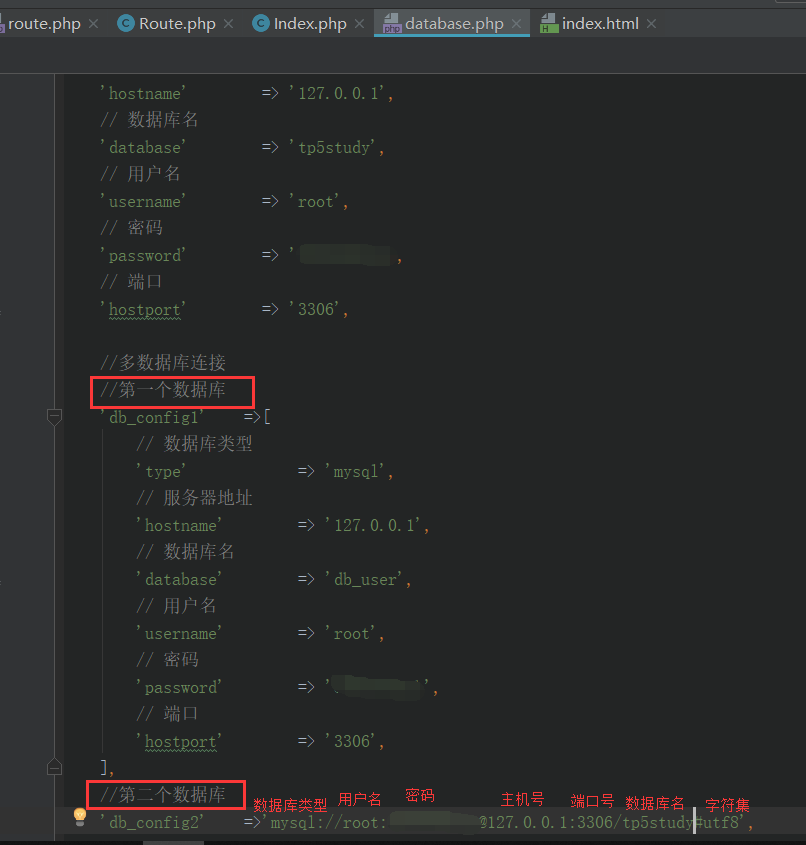


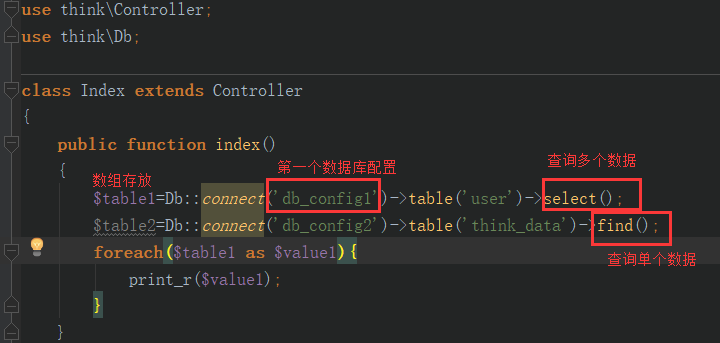
1. 控制器Controller

控制器中的public类型的操作才能在URL中直接访问，而protected类型和private类型的操作是无法直接通过URL来访问的，访问protected和private类型的操作会报异常。

1. request请求对象和response响应对象
2. 数据库配置

tp5.1的数据库配置文件统一在config模块下的database.php文件，config配置目录是项目级的模块，相当于database.php是全局的数据库配置文件，可以在database.php中配置多数据库连接。如下





需要注意：thinkphp的数据库连接是惰性的，指的是并不是在实例化的时候就创建数据库，而是在有实际数据操作的时候才会去连接数据库。

数据库也可开启单独的数据库调试模式：database.php里的‘debug’ => true

数据库的CRUD操作

Create增加：

Read查询：

查询单个数据用find（）方法，查询多个数据用select（）方法

find（）和select（）返回的都是数组，查询结果不存在时均返回null。不过select（）返回的是二维数组，查询结果不存在时均返回空数组。

优化：findOrFail（）和selectOrFail（）在没有查找到数据时会抛出DataNotFoundExecption异常。还有findOrEmpty（）如果查询不到数据返回空数组

如果没设置表前缀的话，Db::name(‘user’)和Db::table(‘user’)函数是一样的效果的

助手函数db(‘表名user’,’数据库配置db\_config1’)

查询某字段的值可以用Db::table('think\_user')->where('id',1)->value('name');若值为空则返回null。查询某一列的值可以用column()

数据分批处理chunk（）方法

Update查询：

Delete删除：