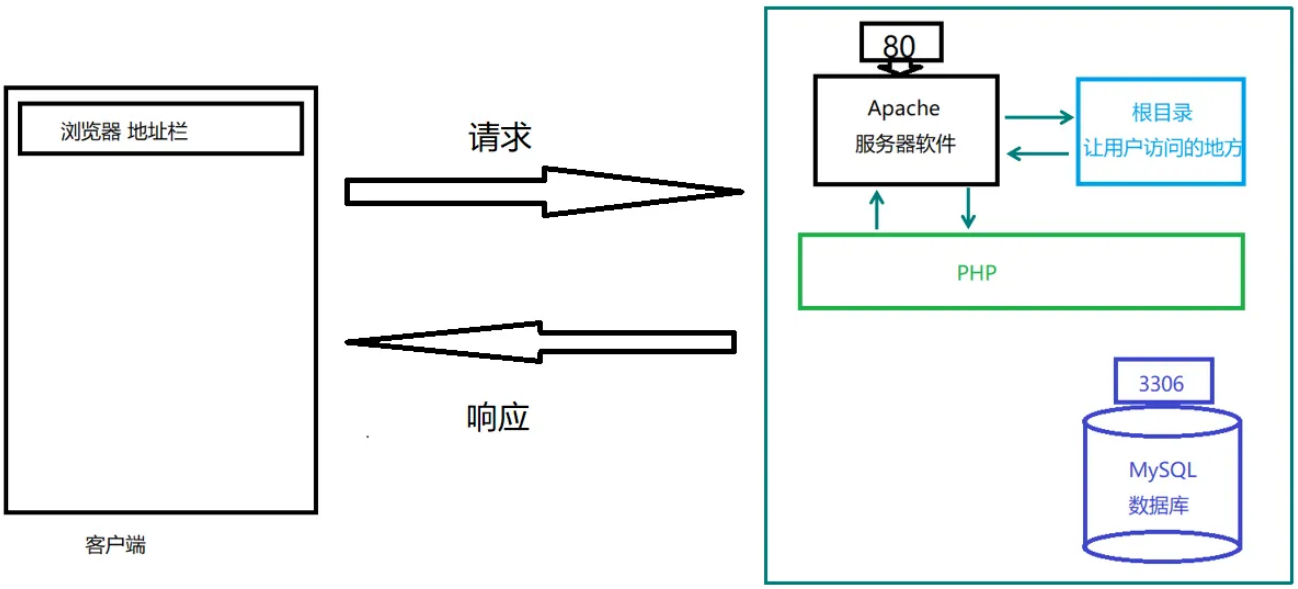
web安全-后端PHP

一 客户端与服务器交互流程



Apache中间件

Apache是Web服务器软件

它可以运行在几乎所有的计算机平台上

最流行的Web服务器端软件之一

Apache的官方网站：http://www.apache.org

二 PHP介绍

PHP（全称"PHP：超文本预处理器"）是一种通用开源脚本语言。

PHP 可免费下载使用，PHP 文件可包含文本、HTML、JavaScript代码和 PHP 代码，PHP 代码在服务器上执行，结果以纯 HTML 形式返回给浏览器。

PHP特点

PHP 可在不同的平台上运行（Windows、Linux、Mac OS X 等）

PHP 与目前几乎所有的正在被使用的服务器相兼容（Apache、IIS 等）

PHP 提供了广泛的数据库支持

PHP 易于学习，并可高效地运行在服务器端

PHPstudy介绍

PHPStudy 是一个集成的 PHP 开发环境工具包，主要用于 Windows 平台，它简化了 PHP 开发环境的搭建过程，特别适合初学者和快速开发需求。

主要特点

- **简单易用**：图形化界面操作，降低开发环境配置难度
- **轻量快速**：相比同类产品占用资源更少
- **多版本切换**：可自由切换不同 PHP 版本

- **安全防护**：内置防火墙和安全检测功能
- **扩展丰富**：支持常见 PHP 扩展一键安装

三 PHP基本语法

PHP 文件默认扩展名为 .php，通常包含 HTML 标签和一些 PHP 脚本代码。PHP 中的每个代码行都必须以分号结束。英文分号是一种分隔符，用于把指令集区分开来。PHP 脚本可以放在文档中的任何位置。

PHP标记

```
<?php
  // PHP 代码
?>
```

PHP注释

//单行注释

/*多行注释*/

PHP变量

变量是用于存储信息的"容器"。

PHP变量名规则

- 变量以 $ 符号开始，后面跟着变量的名称
- 变量名必须以字母或者下划线字符开始
- 变量名只能包含字母、数字以及下划线（A–z、0–9 和 _ ）
- 变量名不能包含空格
- 变量名是区分大小写的（$y和$Y是两个不同的变量）

```php
<?php
$name="xiaolin";
?>
```

PHP页面输出

echo	输出单一类型(数值，字符串，布尔)，多个用逗号隔开
print_r()	输出复合类型（数组，对象），一般用于输出数组
var_dump()	打印数据详细信息(所有类型)

例子

```php
<?php
$name="xiaolin";
echo $name;
?>
```

数据类型–字符串类型

字符串定义方式：单引号，双引号

- 单引号字符串中出现的变量不会被变量的值替代
- 双引号字符串中最重要的一点是其中的变量会被变量值替代
- 如果遇到美元符号($)，解析器会尽可能多地取得后面的字符以组成一个合法的变量名，如果想明确的指定名字的结束，用花括号把变量名括起来

转义字符

\n	换行
\r	回车（ WINDOW \r\n) (linux \n) （Mac OS \r)
\t	水平制表符 (按键盘 tab 产生的效果)
\\	反斜线
\$	美元符(表示变量的开始)
\"	双引号

PHP运算符

算术运算符

+ – * / %

赋值运算符

=

字符串运算符

.（点　字符串拼接符）

递增 ++ 递减 --

++x（前置递增运算符）：先将 x 的值增加1。然后返回 x 递增后的新值
x++（后置递增运算符）：先返回 x 的当前值。然后将 x 的值增加1

运算符	名称	描述
++x	预递增	x 加 1，然后返回 x
x++	后递增	返回 x，然后 x 加 1
--x	预递减	x 减 1，然后返回 x
x--	后递减	返回 x，然后 x 减 1

逻辑运算符

&& ‖ !

运算符	名称	描述	实例
x ‖ y	或	如果 x 和 y 至少有一个为 true，则返回 true	x=6 y=3 (x6 or y5) 返回 true
x && y	与	如果 x 和 y 都为 true，则返回 true	x=6 y=3 (x < 10 && y > 1) 返回 true
! x	非	如果 x 不为 true，则返回 true	x=6 y=3 !(x==y) 返回 true

比较运算符

> < >= <= == === != !== <>

三元运算符

? :

另一个条件运算符是"?:"（或三元）运算符

```
<?php
// 普通写法
$aa = 11>10 ? 'yes' : 'no';
echo $aa;
?>
```

四 PHP流程控制

PHP流程控制语句用于决定代码的执行顺序

PHP 条件语句

if 语句

if 语句用于仅当指定条件成立时执行代码。

语法

```
if (条件){
    条件成立时要执行的代码;
}
```

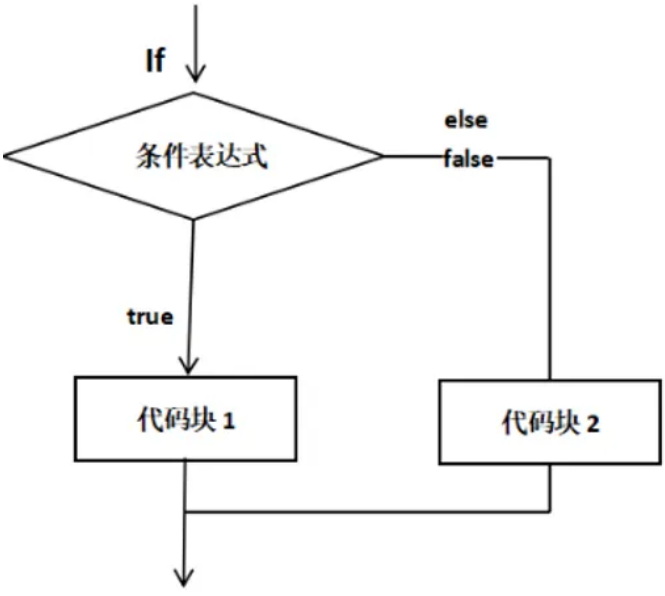
如果当数字小于30，下面的实例将输出 "num<30"

```php
<?php

$num=20;
if ($num<30){
    echo "num<30";
}
echo "num>=30";

?>
```

if...else 语句

在条件成立时执行一块代码，条件不成立时执行另一块代码，请使用 if....else 语句。

```
if (条件){
  条件成立时执行的代码;
}else{
  条件不成立时执行的代码;
}
```



如果当num小于30时，输出num<30；否则输出num>=30。

```php
<?php

$num=20;
if ($num<30){
    echo "num<30";
}else{
    echo "num>=30";
}

?>
```

if...elseif....else 语句

在若干条件之一成立时执行一个代码块，请使用 if....elseif...else 语句。

```
if(条件1){
    条件1成立时执行的代码;
}else if(条件2){
    条件2成立时执行的代码;
}else{
    以上条件不成立时执行的代码;
}
```

switch 语句

switch 语句用于根据多个不同条件执行不同动作，**有选择地执行若干代码块之一。**

工作原理：首先对一个简单的表达式 n（通常是变量）进行一次计算。将表达式的值与结构中每个 case 的值进行比较。如果存在匹配，则执行与 case 关联的代码。代码执行后，使用 break 来阻止代码跳入下一个 case 中继续执行。default 语句用于不存在匹配（即没有 case 为真）时执行。

```php
<?php

$favcolor="red";
switch ($favcolor){
case "red":
    echo "你喜欢的颜色是红色!";
    break;
case "blue":
    echo "你喜欢的颜色是蓝色!";
    break;
case "green":
    echo "你喜欢的颜色是绿色!";
    break;
default:
    echo "你喜欢的颜色不是 红, 蓝, 或绿色!";

}

?>
```

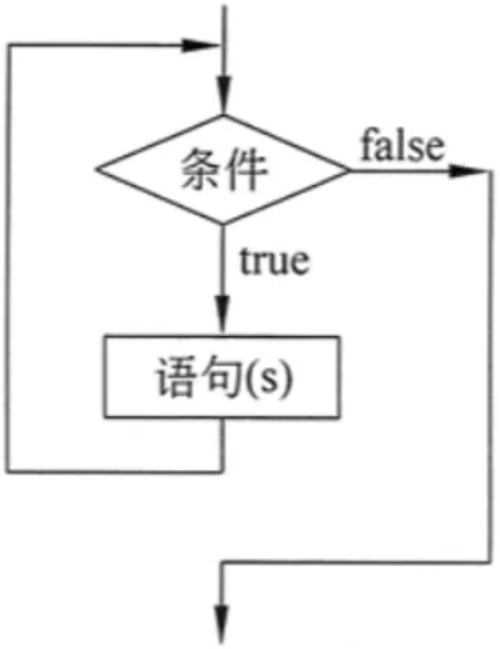
PHP 循环

需要让相同的代码块一次又一次地重复运行。

while 循环

while 循环将重复执行代码块，直到指定的条件不成立。

```
while (条件){
    要执行的代码;
}
```



下面的实例首先设置变量 *i* 的值为 1 ($i=1;)

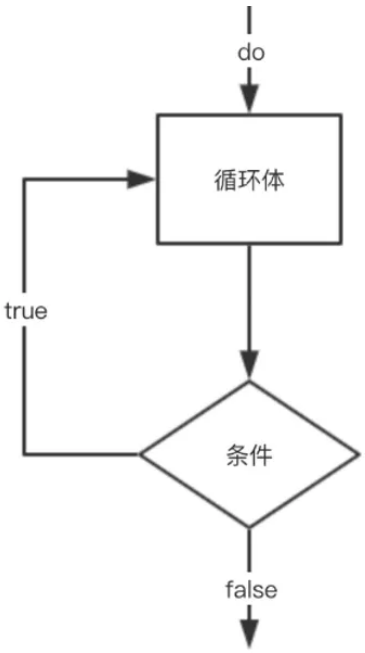
只要 *i* 小于或者等于 5，while 循环将继续运行。循环每运行一次，*i* 就会递增 1

```php
<?php

$i=1;
while($i<=5){
    echo "The number is " . $i . "<br>";
    $i++;
}

?>
```

do...while 语句

do...while 语句会至少执行一次代码，然后检查条件，只要条件成立，就会重复进行循环。

```
do{
    要执行的代码;
}while(条件);
```



下面的实例首先设置变量 *i* 的值为 1 ($i=1;)

然后，开始 do...while 循环。循环将变量 *i* 的值递增 1，然后输出。先检查条件（*i* 小于或者等于 5），只要 *i* 小于或者等于 5，循环将继续运行。

```php
<?php

$i=1;
do{
    echo "The number is " . $i . "<br>";
    $i++;
}while ($i<=5);

?>

```

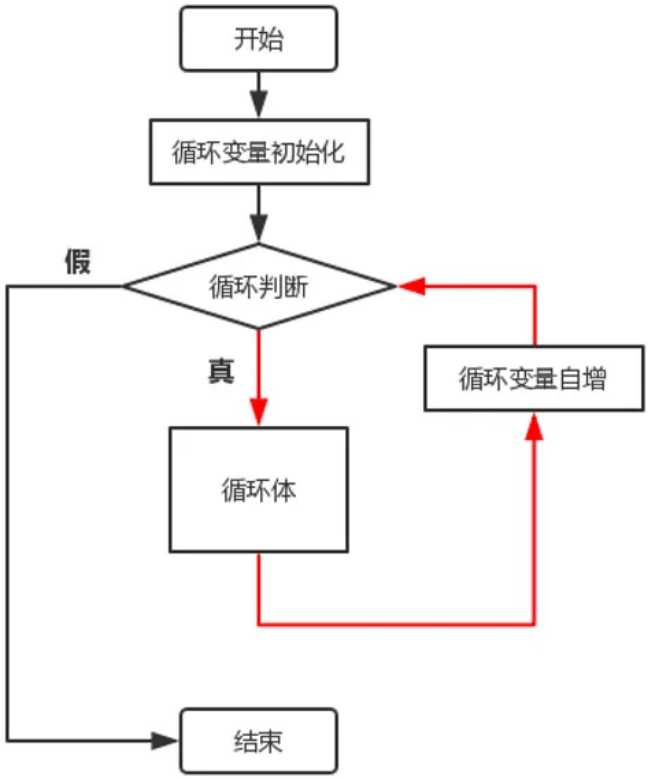
for 循环

for 循环用于您预先知道脚本需要运行的次数的情况。

```
for (初始值; 条件; 增量){
    要执行的代码;
}
```

- **初始值**：主要是初始化一个变量值，用于设置一个计数器（但可以是任何在循环的开始被执行一次的代码）
- **条件**：循环执行的限制条件。如果为 TRUE，则循环继续。如果为 FALSE，则循环结束

- **增量**：主要用于递增计数器（但可以是任何在循环的结束被执行的代码）



下面的实例定义一个初始值为 i=1 的循环。只要变量 i 小于或者等于 5，循环将继续运行。循环每运行一次，变量 i 就会递增1。

```php
<?php

for($i=1; $i<=5; $i++){
    echo "数字为".$i;
}

?>

```

五 PHP函数

PHP 的真正威力源自于它的函数。在PHP中，提供了超过 1000 个内建的函数。

PHP自定义函数

函数是通过调用函数来执行的。

```php
<?php
function functionName()
{
    // 要执行的代码
}
?>
```

PHP 函数准则

- 函数的名称应该提示出它的功能
- 函数名称以字母或下划线开头（不能以数字开头）

一个简单的函数，在其被调用时能输出我的名称

```php
<?php
function writeName()
{
    echo "Kai Jim Refsnes";
}

echo "My name is ";
writeName();//调用函数
?>
```

PHP函数 添加参数

为了给函数添加更多的功能，我们可以添加参数，参数类似变量。

参数就在函数名称后面的一个括号内指定。

实例

下面的函数有两个参数：

```php
function name($s)   //形参
{
    echo $s,"<br>";
}

echo "我的名字是";
name("张三");  //实际参数

```

PHP 函数 – 返回值

如需让函数返回一个值，请使用 return 语句。

```php
<?php
function add($x,$y)
{
    $total=$x+$y;
    return $total;
}

echo "1 + 16 = " . add(1,16);//return调用要使用echo，不然不会返回值
?>
```

PHP内置函数

PHP 有很多标准的函数和结构。PHP系统提供了大量功能强大的函数，帮助我们解决各种问题;

isset();	判断变量是否被设置
empty();	判断变量是否为空
md5();	32位加密字符串
include();	php包含文件
include_once();	php包含文件
require();	php包含文件
require_once();	php包含文件
serialize()	序列话函数
unserialize()	反序列话函数

六 PHP 数组

PHP数组是一种数据结构，用于存储键值对集合，其中每个键可以是整数索引或字符串索引，而值可以是任何类型的数据。数组可以在单个变量中存储多个值，可以根据键访问其中的值。

创建数组

在 PHP 中，array() 函数用于创建数组。或者[　]创建数组，需要php版本5.4+

```
arry();//$声明变量，在括号里面赋予值
```

PHP 索引数组

这里有两种创建索引数组的方法：

自动分配 ID 键（ID 键总是从 0 开始）

```
$a=array("蓝色","红色","黑色");//自动分配ID键，从0开始
echo  $a[0],"<br>",$a[1],"<br>",$a[2];//输出
```

人工分配 ID 键

```
$cars[0]="蓝色";//手动分配下标，也是从0开始
$cars[1]="红色";
$cars[2]="黑色";
    echo  $a[0],"<br>",$a[1],"<br>",$a[2];//输出
```

PHP 关联数组

下标是字符串，关联数组的键值对之间存在明确的对应关系。

```
$ren = array(
    'name' => 'xiaolin',
    'age' => 18
);
```

添加或修改元素

```
$ren['phone'] = '13800138000'; // 添加新元素
$ren['age'] = 26; // 修改已有元素
```

获取数组的长度 – count() 函数

count() 函数用于返回数组的长度（元素的数量）

```
<?php
$a=array("蓝色","红色","黑色");//自动分配下标，从0开始
echo  count($a);  //输出数组长度  ；；；输出结果为3
?>
```

遍历数值数组 foreach

语法：遍历$array数组中的每个元素

```
foreach ($array as $value){
    要执行代码;
}
```

实例：

```
<?php
$a=array("蓝色","红色","黑色");  //定义数组

foreach ($a as $value){    //使用foreach遍历数组，将a数组的值赋值给$value
    echo $value,"<br>";  //再输出$value值
?>
```

七 PHP表单操作

用户提交数据通常是使用表单进行提交，也可以使用网址中的参数传递数据，这些数据通过HTTP请求的方式发送，使web服务器获取。

超全局变量

PHP提供了预定义的超全局变量，用来获取HTTP请求信息。超全局变量都是数组。

$_GET	收集通过URL参数（GET方法）提交的数据
$_POST	收集通过HTTP POST方法提交的表单数据
$_FILES	处理通过HTTP POST上传的文件

字符串处理函数

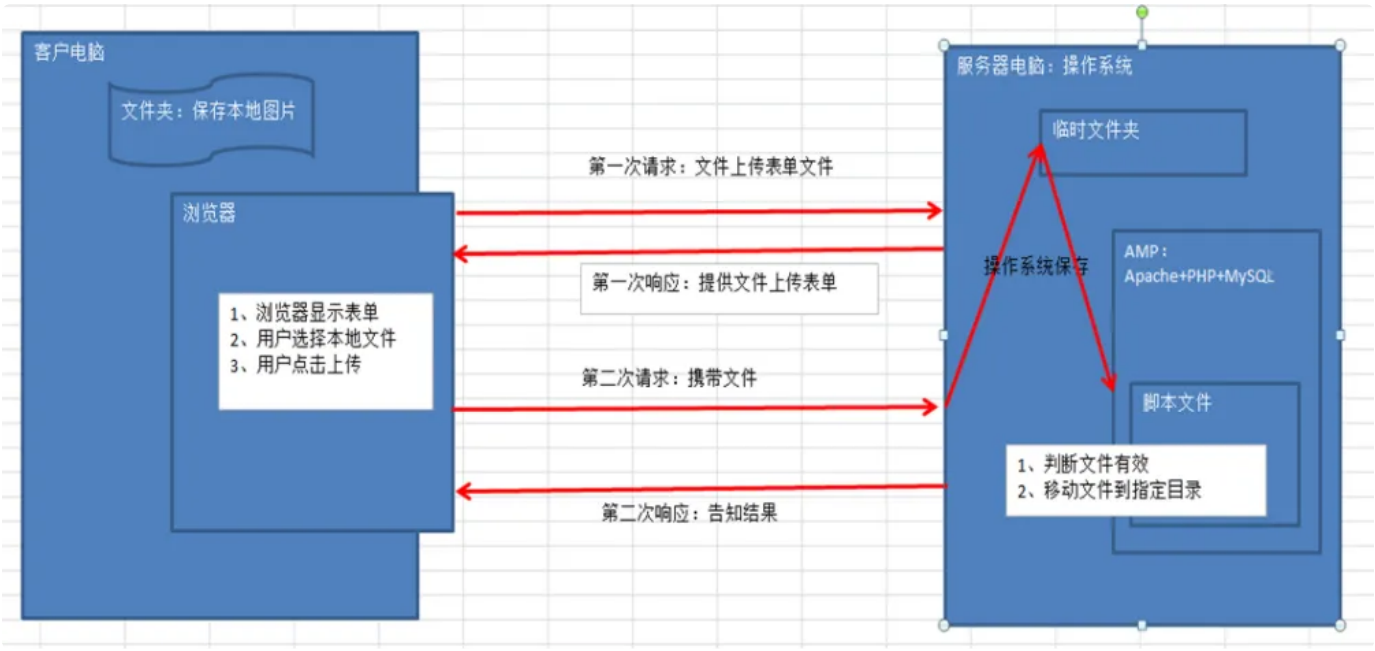
ltrim()	删除字符串左边的空白字符，或指定字符
rtrim()	删除字符串右边的空白字符，或指定字符
trim()	删除字符串两边的空白字符，或指定字符
strlen()	获取字符串长度
substr()	字符串截取
str_replace()	字符串替换
strtolower()	将字符串转换为小写字母
strtoupper()	将字符串转换为大写字母
strip_tags()	删除字符串中HTML XML PHP JS 标签
htmlspecialchars()	函数把一些预定义的字符转换为 HTML 实体字符

表单操作POST请求代码

```php
<?php

if($_POST){
    // print_r($_POST);
    // 数据处理
    $username=trim($_POST["username"]);
    $password=md5($_POST["password"]);
    // echo $password;
    // 保存到txt中  账号和密码之间用空格隔开  换行符
    file_put_contents("user.txt",$username." ".$password."\n");

}

?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>post</title>
</head>
<body>
    <h1>post页面</h1>
    <form action="" method="post">
        <input type="text" name="username" id="">
        <input type="password" name="password" id="">
        <input type="submit" value="登录">
    </form>

</body>
</html>
```

八 PHP文件上传

文件上传流程



php文件上传的必要条件

1）method属性：表单提交方式必须为POST

2）enctype属性：form表单属性，主要是规范表单数据的编码方式enctype="multipart/form-data"

$_FILES

$_FILES超级全局变量作用是存储各种与上传文件有关的信息;

$_FILES是一个二维数组，数组中共有5项：

$_FILES["upload"]["name"]　上传文件的名称，客户端机器文件的原名称。

$_FILES["upload"]["type"]　上传文件的类型，文件的 MIME 类型

$_FILES["upload"]["tmp_name"]　文件上传后在服务器端储存的临时存储路径

$_FILES["upload"]["error"]　文件上传相关的错误代码

$_FILES["upload"]["size"]　上传文件的大小, 以字节为单位

注:upload只是一个占位符，代表文件上传表单元素的名字; 因此这个值将根据你所给定的名称有所不同;

PHP文件上传代码

```php
<?php

header('Content-Type: text/html; charset=utf-8');


if($_POST){
    // 更改原文件名字
    $name=$_FILES['file']['name'];
    $tmp_name=$_FILES['file']['tmp_name'];



    $ext=substr(strrchr($name,"."),1); // 取后缀名
    // 生成随机的文件名
    $file_name=time().rand().".".$ext;

    // 永久保存下来
    $dir="upload";
    if(!is_dir($dir)){
        mkdir($dir,0777,true); // 递归创建
    }
    $path=$dir."/".$file_name; // upload/1234567890.1234567890.txt

    if(!move_uploaded_file($tmp_name,$path)){ // 把临时文件  永久保存下来
        echo "上传失败";
        exit;
    }
    echo "上传成功";
}


?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>文件上传</title>
</head>
<body>
    <h1>文件上传</h1>
    <form action="" method="post" enctype="multipart/form-data">
        <input type="text" name="username" id="">
        <input type="file" name="file" id="">
        <input type="submit" value="上传">
    </form>
```

```
</body>
</html>
```

九 PHP面向对象概念

面向对象（Object Oriented Programming）简称 OOP，是一种编程思想，面向对象是一种以对象（Object）为中心的编程思想。面向对象编程更注重对问题的抽象和封装，通过将问题分解为一系列相互协作的对象来实现程序的功能。

面向过程：根据业务逻辑从上到下写代码

面向对象：将数据与函数绑定到一起，进行封装。减少重复代码的重写过程。

php类和对象

类是抽象的概念，仅仅是模板。用来描述具有相同属性和方法的对象的集合。比如："人"是一个类。对象是类的实例，是某一个具体的事物 。如“张三”则是具体存在 是一个对象

创建类

以下实例中创建了一个名为 Fruit 的类，包含两个属性($name 和 $color),以及两个用于设置 $name 属性的方法 set_name()。在类中，变量称为属性，函数称为方法！$this 关键字引用当前对象，并且只在方法内部可用。

```php
<?php
class Fruit {
  // 属性
  public $name;
  public $color;

  // 方法
  function set_name($name) {
    $this->name = $name;
  }

}

?>
```

类修饰符

public（公有）

- 特性：在任何范围可访问（类内部、子类、类外部）

protected（保护）

- 特性：仅限类内部和子类访问

private（私有）

- 特性：仅定义该成员的类内部可访问

创建对象

没有对象，类什么都不是！ 我们可以从一个类中创建多个对象。 每个对象都有类中定义的所有属性和方法。

PHP 中使用关键字 new 来创建对象。

对象的语法： 对象->属性或者方法()

```
1  //创建了两个对象分别是 $apple 和 $banana
2  $apple = new Fruit();
3  $banana = new Fruit();
4
5  //通过创建的对象调用 Fruit 类中的 set_name 方法来修改属性值
6  $apple->set_name('Apple');
7  $banana->set_name('Banana');
```

魔术方法

PHP 中把以两个下划线__开头的方法称为魔术方法(Magic methods)，这些方法在 PHP 中充当了举足轻重的作用。

__construct()构造函数

php 中构造方法是对象创建完成后第一个被对象自动调用的方法。在每个类中都有一个构造方法，如果没有显示地声明它，那么类中都会默认存在一个没有参数且内容为空的构造方法。

构造方法的作用

通常构造方法被用来执行一些有用的初始化任务，如对成员属性在创建对象时赋予初始值。

构造方法的在类中的声明格式

```
function __constrct([参数列表]){
方法体 //通常用来对成员属性进行初始化赋值
}
```

例子

```
<?php

class Person{
  public $name;
  public $age;
  public $sex;
/**
* 显示声明一个构造方法且带参数
*/
  public function __construct($name="", $sex="男", $age=22){
    $this->name = $name;
    $this->sex = $sex;
    $this->age = $age;
  }

  public function say(){
    echo "我叫: " . $this->name . ", 性别: " . $this->sex . ", 年龄: " . $this->age;
  }

}

//创建对象$Person1 且不带任参数
$Person1 = new Person();
echo $Person1->say(); //输出:我叫: , 性别: 男, 年龄: 27
//创建对象$Person2 且带参数“小明”
$Person2 = new Person("小明");
echo $Person2->say(); //输出: 我叫: 张三, 性别: 男, 年龄: 27
```

__destruct()析构函数

析构方法允许在销毁一个类之前执行的一些操作或完成一些功能，比如说关闭文件、释放结果集等。析构方法是 PHP5 才引进的新内容。析造方法的声明格式与构造方法 __construct() 比较类似，也是以两个下划线开始的方法 __destruct() ，这种析构方法名称也是固定的。

注意：析构函数不能带有任何参数。

```
function __destruct(){
//方法体
}
```

析构方法的作用

一般来说，析构方法在 PHP 中并不是很常用，它属类中可选择的一部分，通常用来完成一些在对象销毁前的清理任务。

```
<?php
class Person{
  public $name;
  public $age;
  public $sex;

  public function __construct($name="", $sex="男", $age=22){
    $this->name = $name;
    $this->sex = $sex;
    $this->age = $age;
  }

  public function say(){
    echo "我叫：".$this->name."，性别：".$this->sex."，年龄：".$this->age;
  }
  public function __destruct(){
    echo "我觉得我还可以再抢救一下，我的名字叫".$this->name;
  }

}

$Person = new Person("小明");
unset($Person); //销毁上面创建的对象$Person
```

__sleep()

php序列化：将PHP数据结构转换为可存储/传输的字符串格式；序列化函数serialize()

在对象被序列化之前，此方法会被调用。可以在这里做一些清理工作，比如关闭数据库连接，或者决定哪些对象属性应该被序列化。

__wakeup()

php反序列化：将序列化字符串还原为原始PHP数据结构；反序列化函数unserialize()

在对象被反序列化之后，此方法会被调用。这里可以用来重新建立那些在__sleep()中被断开的链接，比如重新打开数据库连接。