PHP学习笔记

一.学前准备

- 需要安装的软件

> web服务器　apache
>
> PHP应用服务器　　PHP
>
> 数据库管理系统 数据库服务器　Mysql

- 上述方法太难了，所以直接用**wampserver**集成环境即可
- `127.0.0.1`即为自己主机的ip，访问该网址等于访问自己
- 根目录下有有很多.php文件，当php和html文件同时存在时，默认运行php文件
- 选择一款适合自己的编辑器，我选了Sublime

二.PHP基本语法

2.1初识PHP脚本

- PHP语言标记:

> 开始:<?php
>
> 结束：?>
>
> ps：类似c语言的中括号

- 页面最终通过html，css，js来展示出绚丽的界面，可以嵌入任意多个php文件去html文件
- `echo '内容';`表示输出内容，如：echo 'hello world'
- 结束标志?>隐含一个分号，所以PHP代码最后一行可以不加分号
- > 关于注释
  >
  > `/*　　注释内容　　*/` #多行注释
  >
  > `//注释内容` #单行注释

2.2变量

- 使用php时使用变量无需声明
- `$变量名=赋值内容`php赋值变量语句；

```
1 //赋值数字
2 $price=10;
3 //赋值字符串
4 $name='kino';
```

变量名只可以包含字母，数字，下划线，并且以字母或者下划线开头

- 变量的销毁：`unset($变量)`

```php
//引例
<?php
$name='kino';
$price=10;
echo $name,$price
    ?>
    // 输出结果为：
    kino10
//引例2
    <?php
    $name='kino';
    unset($name);
echo $name;
//会报错

//引例3
<?php
    $name='kino';
    $$name='YYL';  //意思是$kino='YYL'
echo $kino
    //输出结果为YYL
```

- 连续赋值

```php
$a=10;
$b=$a;
echo $b;
//结果为10
```

2.3变量类型

-

> bool布尔类型　　true=1,false=0
>
> int整数型
>
> float浮点型
>
> str字符串型
>
> array 数组
>
> object 对象
>
> resource 资源
>
> Null 空

- `var_dump(%变量名)`可以输出变量的类型，如var_dump($a)

- \为转义，如\ '输出的就是一个'
- 单引号内的变量不会被解析，不会将变量替换
- ""，双引号中的内容，内容中假如有变量那就会被解析，用{}将变量包裹

```
$a=1;
echo "hello,world$abc"; //里面的$a不会被替换
echo "hello,world{$a}bc"//正确写法
```

- <<<定界符

```
$a=<<<abc
sadsadasdasdas
abc;
echo $a //输出的将是sadsadasdasdas
//<<<中的变量也会被解析，也要加{}
```

- `（int）内容` 将内容强制转换为int型，和C语言类似
- `echo '< br >'`，表示输出空行。

2.4常量

- `define('常量名称'，常量值)`，其中常量名称可以在单引号里面也可以在双引号里面，常量面前没有$符号，常量定义后不能更改

  如：`define('myname','kino)`

  `echo myname;` #表示输出myname

- 预定义常量：系统定义好了的常量，可直接拿来使用

```
_FILE_        代码所在文件夹位置
_LINE_        当前行数
_FUNCTION_ 当前函数名
_CLASS_       当前类名
_METHOD_    当前对象的方法名
PHP_OS         UNIX或WINNT
PHP_VERSION   当前PHP版本
DIRECTORY_SEPARATOR   /或\，根据操作系统决定目录的分隔符

```

三.循环

- `break` 语句，跳出当前循环，后面跟数字几就是跳出第几重循环，如：break2就是跳出第二层循环。
- `exit()` 结束当前整个PHP脚本的执行，作用与`die`类似

```php
echo 'HELLO,WORLD';
echo 'HELLO,WORLD';
exit('对不起程序结束');
echo 'HELLO,WORLD';
echo 'HELLO,WORLD';
echo 'HELLO,WORLD';
echo 'HELLO,WORLD';
echo 'HELLO,WORLD';
/*结果为:HELLO,WORLD
        对不起，程序结束
         HELLO,WORLD
```

- while和for的用法和c语言中的一模一样就是多了个$

四.函数

- `function 函数名(参数)`创建自定义函数

```php
function test()
{
    echo 'hello,world';
}
```

- `global $变量`声明变量，使得自定义函数内可以调用该全局变量
- 常量可以不用理会范围，在任何范围都可以使用
- `static $变量`定义静态变量，静态变量只会执行一次，仅在第一次调用执行

```php
function test()
{
    static $a=10; //静态变量a，只在第一次调用时执行
    echo ++$a; //当第二次执行时，$a依然存在
}
test();
test();

/* 输出结果
11
12
```

- `func_get_args()`，读取所调用函数传输进去的参数
- `func_get_arg(数组号)`，将输入的参数看为数组，读取指定的元素
- `func_num_args()`返回传入自定义函数的参数的个数
- **其他内容也和C语言的一模一样，没啥要记得**

五.数组

5.1数组类型

- 索引数组：数组全是整数型
- 关联数组：数组是字符串型

5.2创建数组

- `print_r(数组名)`输出数组
- 第一种创建方法:

```php
$student[0]=10;
$student[1]='杨奕亮';
$student[2]=true;
$student[3]=60.5;
print_r($student);
/* 输出结果：
Array([0]=>10[1]=>杨奕亮[2]=>1[3]=>60.5)
```

- 方法二:

```php
$student[]=10;
$student[]='杨奕亮';
$student[]=true;
$student[]=60.5;
var_dump($student);
/*输出结果
array(size=4)
0=> int 10
1=> string '杨奕亮'
2=> boolean true
3=> float 60.5
ps：不写索引值（中括号里的内容）默认写整数
```

- 方法三:

```php
$student=array(10,'杨奕亮',true,60.5);
var_dump($student);
/*
array(size=4)
0=> int 10
1=> string '杨奕亮'
2=> boolean true
3=> float 60.5
```

```php
$student=array('num'=>10,'name'=>'杨奕亮','sex'=>true,'grade'=>60.5)
    var_dump($student);
/*
array(size=4)
num=> int 10
name=string '杨奕亮'
sex=true
grade= float 60.5
```

- 数组里面同样可以存放数组，即为二维数组

5.3遍历数组

- `count(数组)`:返回数组里面数据的个数
- `foreach(数组变量 as 变量1){}`实现遍历数组

```php
$test=array('name'=>'YYL','num'=10);
foreach($test as $a)
{
    echo $a,'<br>';
}

/* 输出结果:
            YYL
            10
```

```php
$test=array('name'=>'YYL','num'=10);
foreach($test as $key=>$a)
{
    echo $key.'=>'.$a.'<br>'
}

/* name=>YYL
   num=>10
```

- echo不能输出数组，假如数组中含有数组，那么echo就不能输出该数组
- `foreach`的递归，foreach只会遍历第一层

```php
$arr=array(
    array(1,2,3,4,5),
    array(1,2,3,4,5,6),
    array(1,2,3,4),
)
foreach ($arr as $val1)
{
    foreach($val1 as $val2)
    {
        echo $val2.'<br>';
    }
}
```

```
13
14 //输出结果就是遍历数组
```

5.4预定义超全局变量

- 传递数据给服务器的方法有两种：

> GET方式：
>
> 比如http://localhost/test/index.php?参数1=数值&参数2=数值
>
> POST方式：
>
> 结合html页面提交数据，然后用$_POST去获取数据
>
> GET和POST的区别：
>
> GET是通过URL方式请求，可以直接看到，明文传输。
>
> POST是通过请求header请求，可以开发者工具或者抓包可以看到，同样也是明文的。

5.5处理数组的相关函数

- ```
  array_count_values
      $arr=array_count_values($array);//函数返回一个数组
      print_r($array) //原来的数组不会受到任何影响
  ```

- ```
  array_key_exists--检查给定的键名或者索引是否存在数组中
      $search_array=array('first'=>1,'second'=>2);
    var_dump(array_key_exists('YYL'));//返回值为false说明不存在

  ```

- ```
  array_search 在数组中搜索给定的值，如果成功则返回相应的键名
     $a=array('first'=>1,'second'=>2);
  var_dump(array_search('first',$a));
  /* first
  若不存在返回false
  ```

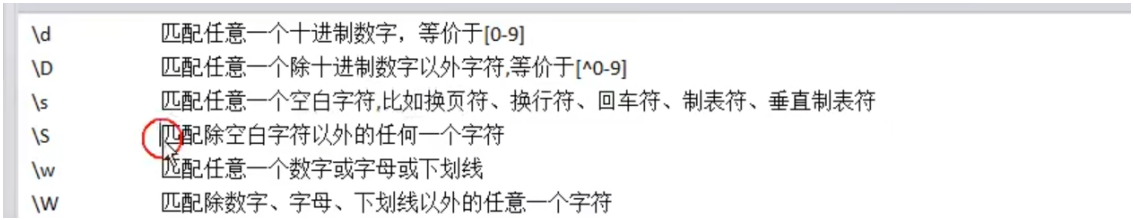
- ```
  in_array(需查询的数，数组) 在数组中搜索数据，如果找到返回ture，反之false

      $os=array(1,2,3,4,5);
  var_dump(in_array(1,$os));//返回的是true
  ```

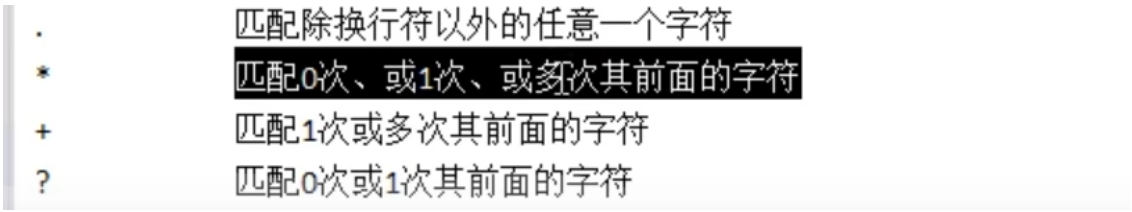
- ```
  list();

      $arr=array(60,80,100);
      list($YYL,$xiaoming,$xiaohong)=$arr;
  echo $YYL;
  //输出结果为：60
  ```
  ```

- ```
  asort 对数组进行排序并保持索引关系
      $students=array(
                          'YYL'=100;
                           'XX'=60;
                           'TT'=70;
                            'MM'=90;
                  );
  var_dump(asort($student));//返回true，说明排序成功
  print_r($student);//依次输出XX,TT,NN,YYL,由低到高
  ```

六.字符串处理

指令内容

- `trim(需要修改的字符串，指定去除的字符串)`去除字符串首尾处的空白字符（或者其他字符）；

  ```
  $str='abcdefga';
      $b=trim($str,'a');
  echo $b;
  //bcdefg
  ```

- `ltrim`去除左边的特殊字符，和trim用法一致
- `rtrim`去除右边的特殊字符，和trim用法一致
- `strtoupper`将字符串变成大写
- `strtolower`将字符串变为小写
- `substr_count`计算子串出现的次数
- `strpos`查找字符串首次出现的位置
- `strstr`查找字符串的首次出现，返回的是从开始到结尾的字符串
- `str_replace`字符串替换
- `substr`字符串截取函数
- `explode(用于分割的字符，字符串)`分割字符串
- `str_split`将字符串分割为字符数组

七.正则表达式

- `preg_match_all`函数
  ```

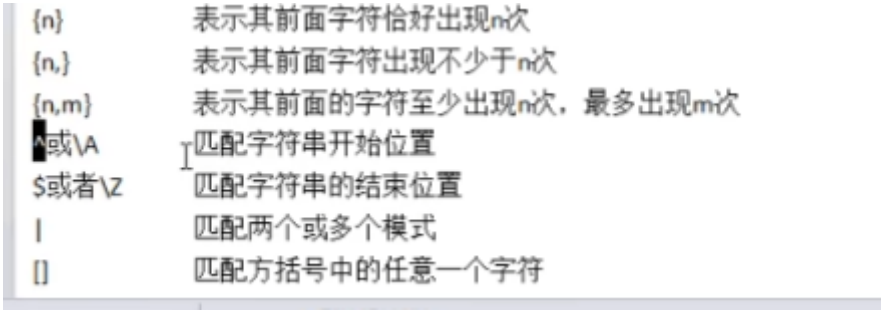
```php
$pattern='/test/'; //正则表达式
$str='asdasdastestsasdxzcxztest';
var_dump(preg_match_all($patterm,$str,$arr));
var_dump($arr);
/*输出结果：int 2 （表示有2个匹配结果）
            将匹配的字符存入数组arr中
```

- 定界符除了/还有#,!,{},|
- 元字符`\d`表示[0-9];

```php
$pattern='/t\dest/'; //正则表达式
$str='abct1st'
   var_dump(preg_match_all($patterm,$str,$arr));
var_dump($arr);
/*
int 1
array
0=>t1st

```

- 元字符`\D`表示`[^0-9]`除了0-9以外的字符
- 

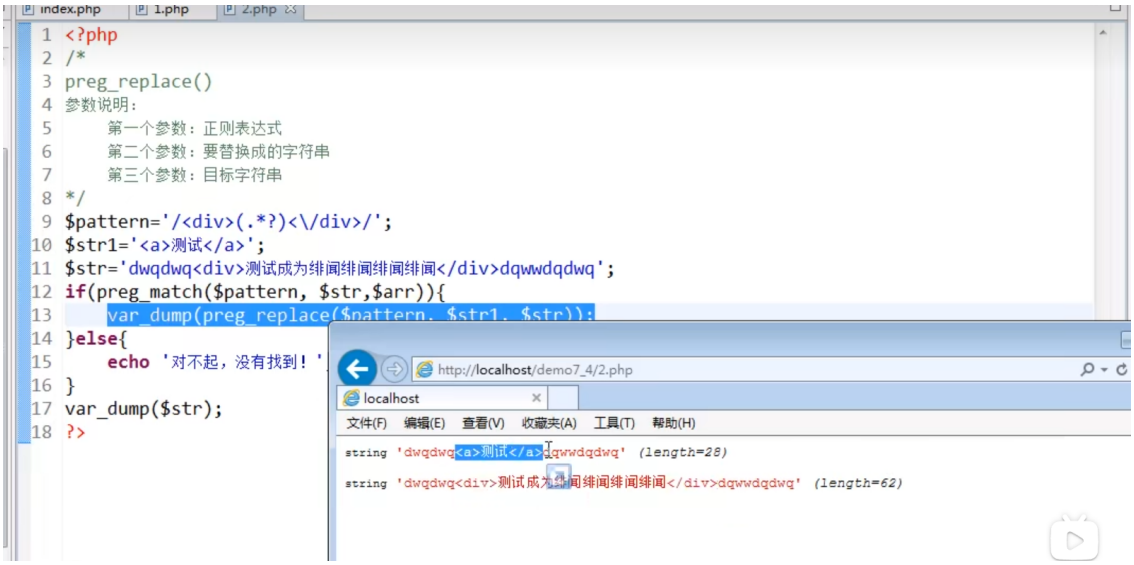
- 

- 

- ```php
  $pattern='/^test/';
  $str='abctestabc';
  var_dump(preg_match_all($pattern,$str,$arr);
            /*匹配不到结果
            ^是开始的位置，要求字符串以t为开始
  ```
- `/test$/`表示以t结尾
- `/t[^e]st/`方括号中表示除了e都可以取
-
  ```

常见模式修正符

i	在和模式进行匹配时不区分大小写
m	多行匹配，如果目标字符串 中没有"\n"字符，或者模式中没有出现^或$，设置这个修饰符不产生任何影响
s	如果设定了此修正符，那么.将匹配所有的字符包括换行符
U	禁止贪婪匹配

- 

八.文件与目录操作

8.1目录的操作

- `is_file()` 判断给定文件名是不是一个正常的文件

```
is_file函数返回值是布尔类型
    var_dump(is_file('文件名或者路径'));
    //返回值是布尔类型的true或false
```

- `is_dir()` 判断是否为目录，返回值为布尔
- `file_exists()` 检查文件或者目录是否存在，返回值为布尔
- `filesize()` 取得文件的大小，单位为字节数，返回值为int
- `is_readable()` 判断给定文件是否可读
- `is_writable()` 判断文件是否可写
- `filectime()` 获取文件创建时间
- `filemtiome` 获取文件的修改时间
- `fileatiem` 获取文件上次访问时间
- `stat()` 给出文件信息
- `basename` 返回路径中的文件名部分basename(路劲);
- `dirname` 返回路径中的目录部分
- `pathinfo` 返回文件路径的信息
- `opendir` 返回路径句柄

8.2文件的操作

- `fopen`打开文件或者URL，返回句柄，将指针指向文件头（光标）
- `fread`读取文件
- `fgets`从文件中读取一行
- `reof`测试文件指针有没有到末尾
- `fwrite（句柄，写入的数）`写入文件
- `fseek`定位文件指针

九.类与对象

9.1复合类型 基本概念

- 伪变量$this
- extends 继承另一个类的方法和属性
- 范围解析操作符(::)访问静态成员，::class获得类完全限定名称

```php
<?php
class tom{
    public $a=123;
    public function a(){   //可以写成function a()，省掉public，默认有

    }
}
$a=new tom;
var_dump($a);
```

```
object(tom)#1 (1) {
  ["a"]=>
  int(123)
}
```

```php
<?php
class tom{
    public $a=123;
    public function a(){   //可以写成function a()，省掉public，默认有
        return 666;
    }
}
$a=new tom;
var_dump($a->a());
```

改成return $a会无法获取，这个和之前有点不一样

改成return $this->$a,获取到的结果是123

9.1.1继承

```php
<?php
class jack {
    public $j=1986;
}
class tom extends jack
{
    public $a=123;
    public function a(){    //可以写成function a()，省掉public，默认有
        return 666;
    }
}
$a=new tom;
var_dump($a);
```

- 若果在jack类里面加入`public $a=1988;`输出结果依然如上，因为是tom继承jack，所以tom里面原来有$a将jack里面的$a覆盖了
- 如果在jack类里加入`function a(){return 666}`也会覆盖，因为tom类里有function a(),将a改为ab，`var_dump($a->abc())`的结果就是666

9.1.2范围解析操作符(::)

范围解析操作符::是**只能**用来访问静态量

static静态常量是不会被输出的如：

```php
<?php
class jack {
    public static $j=1986;
}
```

```php
class tom extend jack
{
    public $a=123;
    public function a(){    //可以写成function a()，省掉public，默认有
        return 666;
    }
}
$a=new tom;
var_dump($a);
echo jack::$j;
```

```
object(tom)#1 (1) {
  ["a"]=>
  int(123)
}
```

```php
echo jack::$j;
```

run (ctrl+x) 输入 分享当前代码 出现故障，请

文本方式显示 html方式显示

```
1986
```

9.2final 关键字

- final:父类方法声明final，不能覆盖；类声明final，不能继承
- 属性不能被定义为final(就类似于public $a=123这种)
- 通过parent::来访问被覆盖的方法或者静态属性

```php
<?php
    clas jack{
    public $j=1986;
    public $a=10086;
    function belt(){
        return 666;
    }
}
    class tom extends jack{
        public $ab=123;
        function bela(){
            return 100;
        }
    }

$a=new tom;
    var_dump($a);
```

```
object(tom)#1 (3) {
  ["ab"]=>
  int(123)
  ["j"]=>
  int(1986)
  ["a"]=>
  int(10086)
}
```

变成`final class jack`之后就会报错，因为final方法声明的不能被继承

变成`public final $j=1986`,也不行，属性不能加final

final加在belt前面，tom就不能覆盖jack里面的方法

9.2.1 parent

```php
<?php
    clas jack{
    public $j=1986;
    public $a=10086;
  final function belt(){
        return 666;
```

```
    }
}
    class tom extends jack{
        public $ab=123;
        function belt(){
            return 100;
        }
        function test(){
            return self::belt();
        }
    }

$a=new tom;
    var_dump($a->test());

//返回值为100
```

```
<?php
    clas jack{
   public static $j=777;
    public $j=1986;
    public $a=10086;
  final function belt(){
        return 666;
    }
}
    class tom extends jack{
        public $ab=123;
        function belt(){
            return 100;
        }
        function test(){
            return parent::belt();
        }
         function test2(){
            return parent::$k;
        }
    }

$a=new tom;
    var_dump($a->test());
    var_dump($a->test2());

//返回值为666
//返回值为777
```

9.3属性properties

- 类的变量：成员叫做属性（常量不是属性）
- 调用：用->（对象运算符）：$this->property 非静态熟悉
- nowdoc结构：大段文本而无需对其中的特殊字符进行转义

笔记暂时记录到这里