* 正则表达式
  + 作用：做字符串的匹配或者替换，是字符串函数的补充
  + 总体要求：如果是简单的字符串操作，能用字符串函数的，就用字符串函数，不行的再用正则
* 组成：
  + 定界符
    - 作用：限定正则边界的
    - 可用符号：好多，不能用的有：字母，数字，反斜线\
    - 常用的符号：/
    - 注意：
      * 1，定界符是成对出现的
      * 2，定界符要一致
  + 原子
    - 作用：就是正则的核心，也就是表达式
    - 分类：
      * 1.普通字符，a-z A-Z 0-9
      * 2,特殊的转义符号
        + 注意：和php中的转义字符不一样，而是正则当中的特定符号的转义，比如说：” ‘ ? + . \*
      * 3.非打印字符，常规意义上的转义字符
        + \n \r \f \t \cx
      * 4.通用原子
        + \d:匹配数字
        + \D：匹配非数字
        + \s：匹配空白字符，\n \r \f \t ..
        + \S:匹配非空白字符
        + \w：匹配字母，数字，下划线
        + \W：匹配非字母，数字，下划线
      * 5.自定义原子
        + 特点：

[]：匹配[]内的任一字符

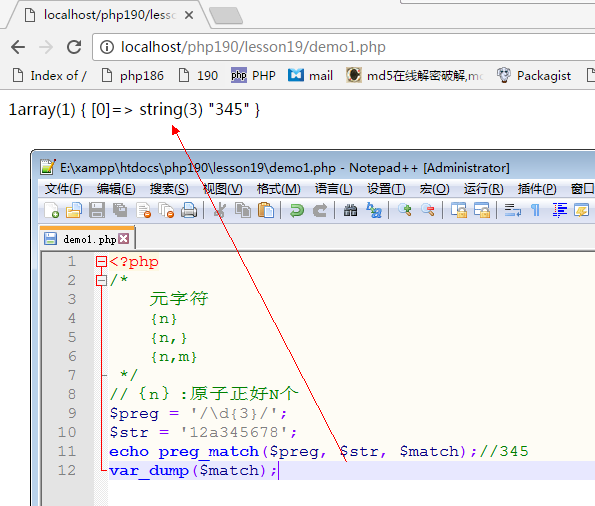
[^]:匹配除[]内的任一字符

* + 元字符
    - 作用：修饰原子，增强原子功能
    - ？：匹配0，1
    - +：匹配1，多次
    - \*：匹配0，1，多次
    - .:匹配除\n之外所有字符
    - |:分支中挑一个
    - {n}：原子正好N个
    - {n,}：最少N个
    - {n,m}：最少N个，最多M个
  + 模式修饰符

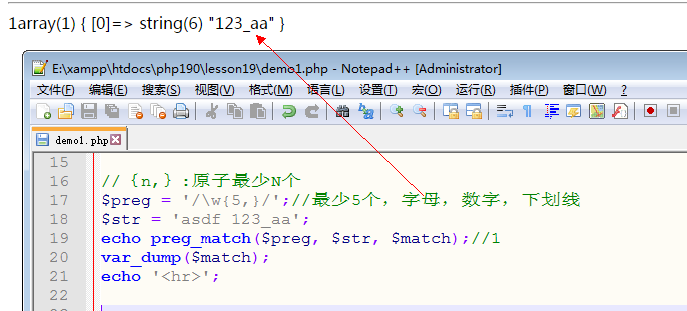
# 正则表达式

## 元字符

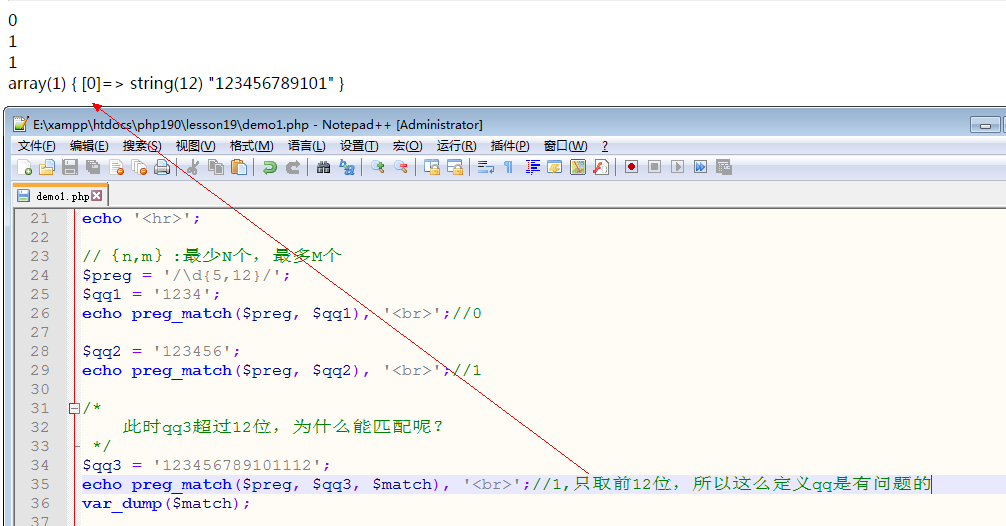
### ｛n｝:匹配前面的原子正好N个



### {n,}：匹配前面的原子最少N个

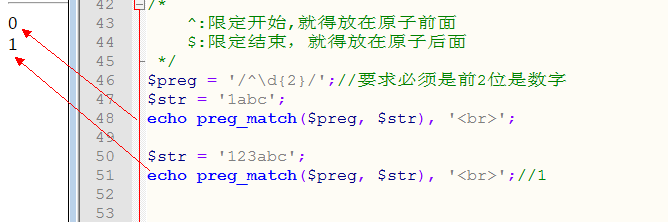


### {n,m}：匹配前面的原子最少N个，最多M个



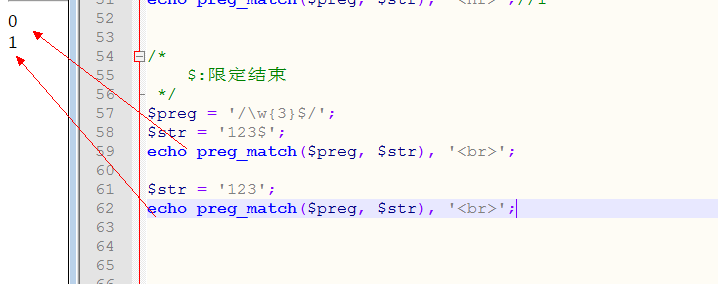
### ^：限定开头

* **注意：如果想要限定某个原子开头，必须把^放在原子的前面**

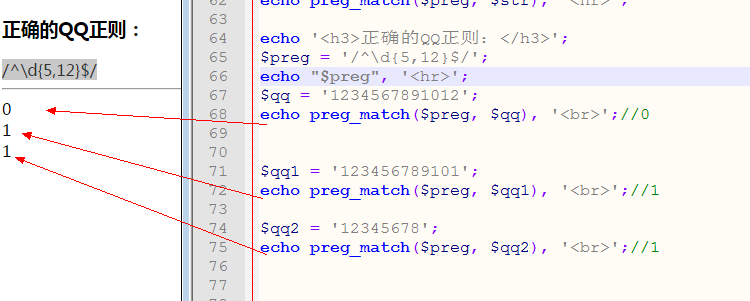


### $:限定结尾

**注意：如果想要限定原子，必须把$放在原子后面**



#### Qq号的正则：

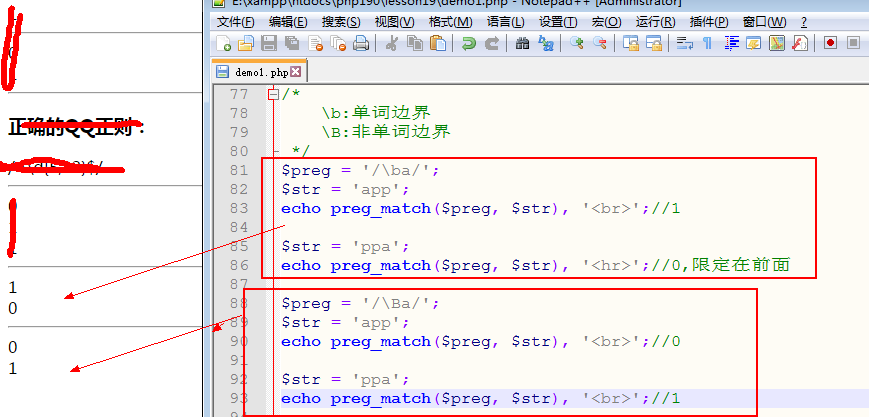


### \b：匹配单词边界

* 可以放在原子的前面或者后面

### \B：匹配非单词边界

* 也可以放在原子的前面或着后面

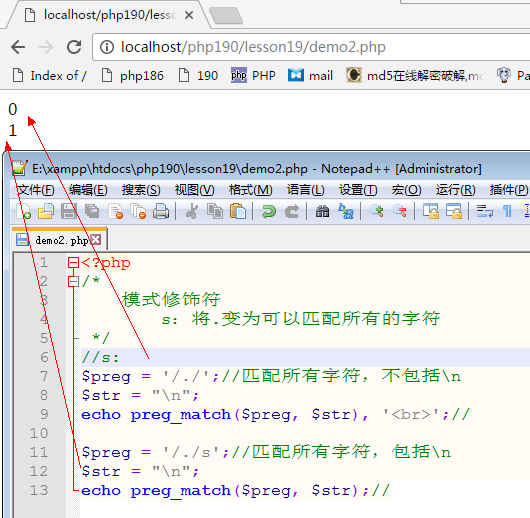


## 模式修饰符

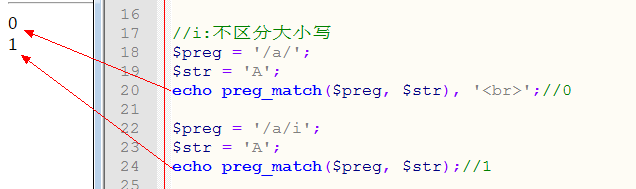
* 位置：
  + 在结束定界符外，
* 作用：
  + 修饰整条正则表达式

### S: 将.变为可以匹配所有的单个字符

* 包括\n在内



### I:不匹配大小写



### M: 将单行字符串视为多行

## 函数：

### Preg\_match(正则，字符串[,$match])

* 返回0，1
* 匹配到了返回1，匹配不到返回0
* 只要匹配到1个就结束了

### Preg\_match\_all(正则，字符串[,$match])

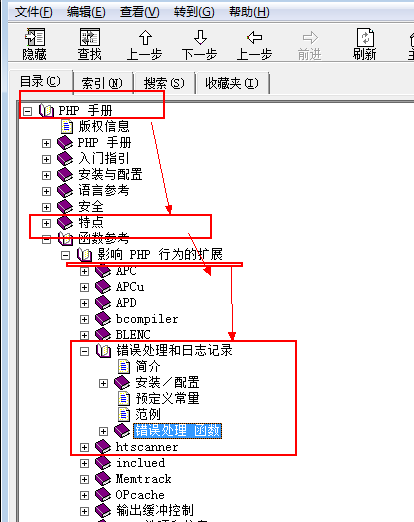
* 返回匹配到的数目
* 会尽可能 的匹配
* 可以使用第三个参数来捕获匹配到的结果

## 正则的要求：

* 1.能写出常见的正则表达式
* 2，可以拆解所有的正则表达式

# PHP错误处理

## 手册位置：



## 错误的级别

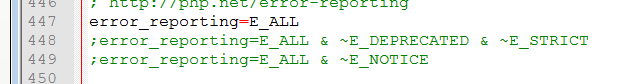
* 错误有级别，最常见的：notice,warning,error
* 每个级别都是常量，错误级别常量对应的值是唯一的，值可以代表级别
* 注意;错误级别对应的值是：
  + **2^(n-1)？为什么这么做？**
  + **保证任意一个值的结果都是唯一的**
  + **比如说：10，肯定是2+8**
  + **比如说：7，肯定是1+2+4**



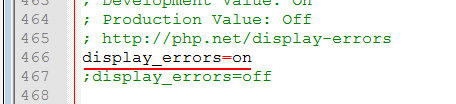


## 错误的配置

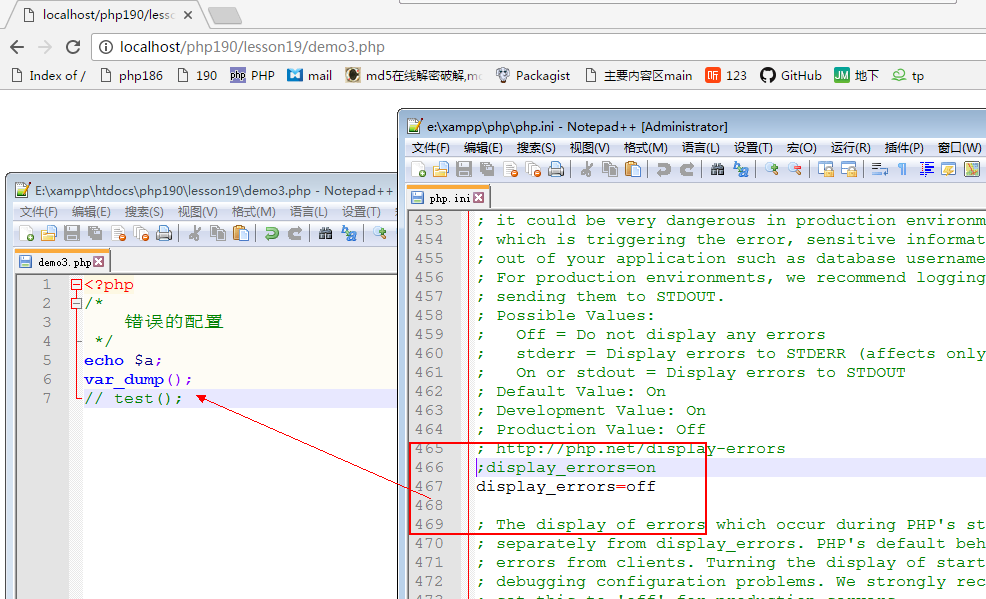
* Php.ini中的配置
  + **error\_reporting=E\_ALL & ~E\_DEPRECATED & ~E\_STRICT**
    - **错误级别配置**



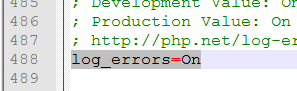
* **display\_errors=on**



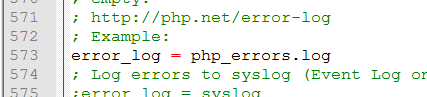
* 默认开启，可以关闭。关闭之后不再有错误提示



* **log\_errors=On**
  + **是否开启错误日志功能，是**



* **error\_log = php\_errors.log**
  + **错误日志的名称**

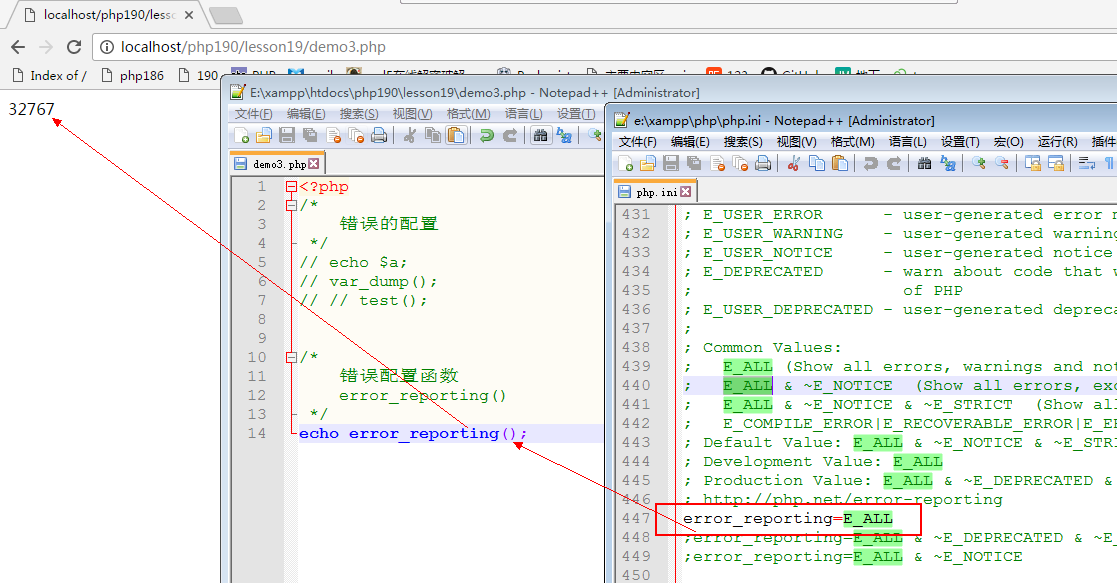


## 错误配置的函数

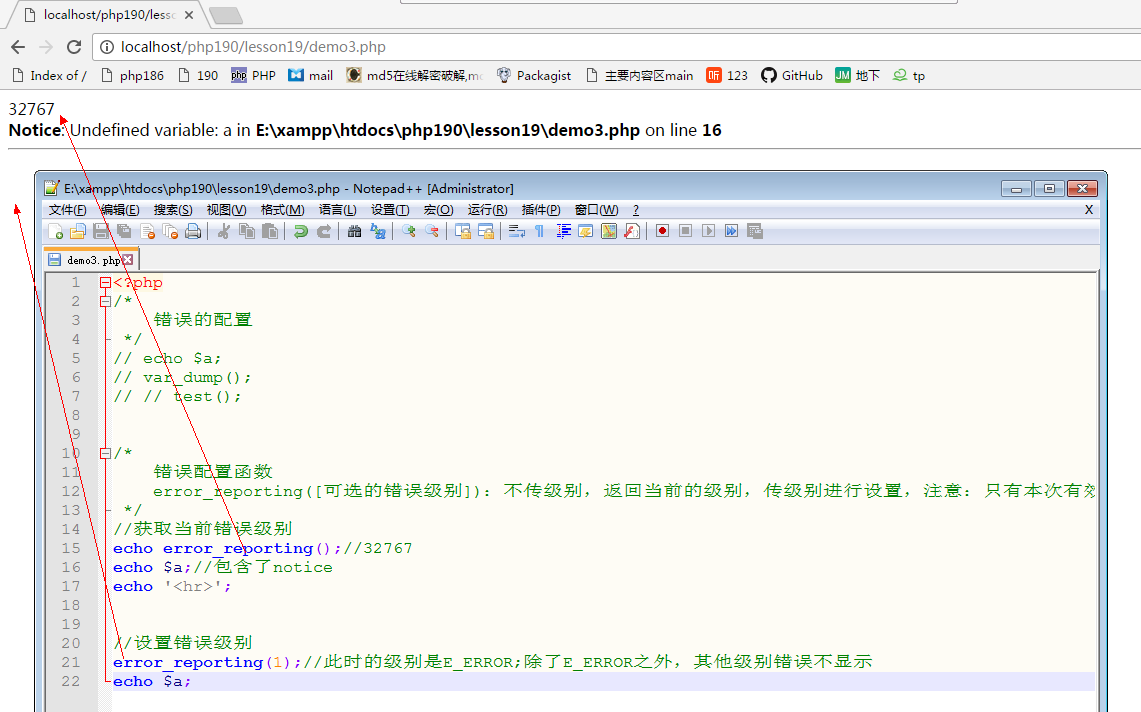
### Error\_reporting()



* **注意：php7中，E\_ALL包含了E\_STRICT这个错误级别，所以，E\_ALL对应的值是32767**



* 示例：

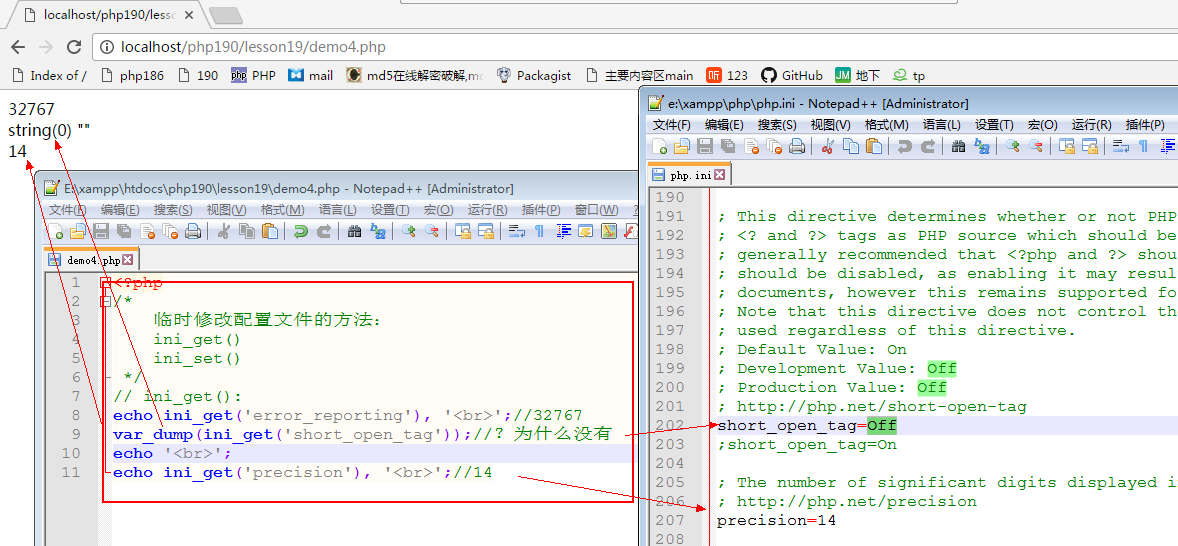


## 扩展-phpini配置的临时修改

### Ini\_get(‘配置项名’)



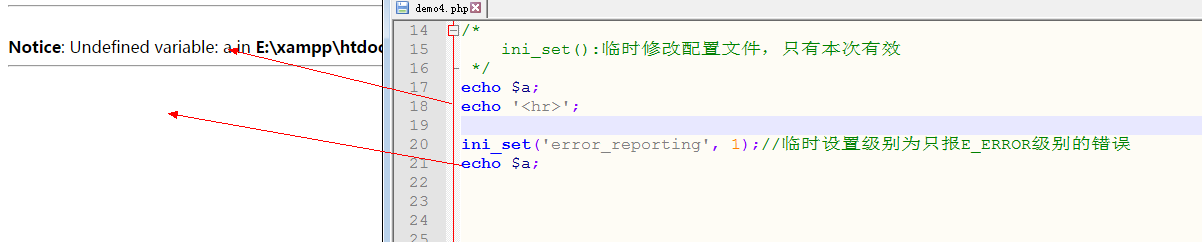
* 示例：



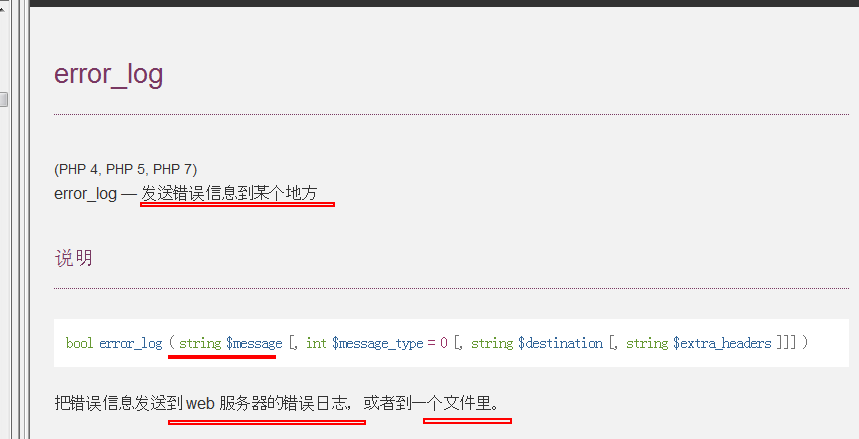
### Ini\_set(‘配置名’, ‘配置值’)



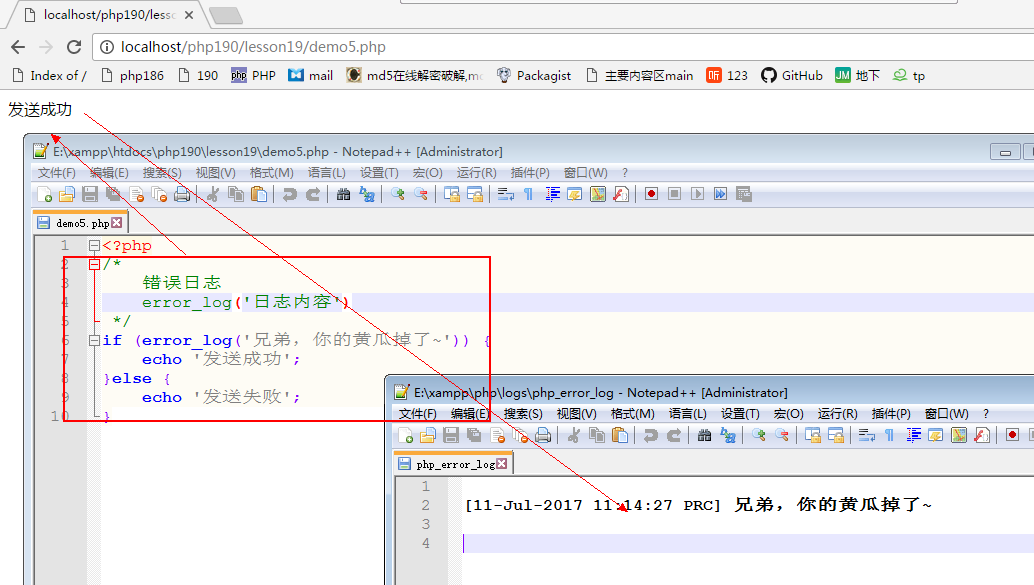
示例：



### Error\_log()

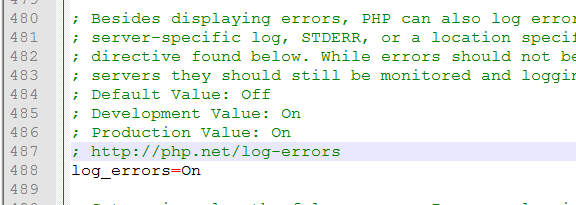


* 示例：

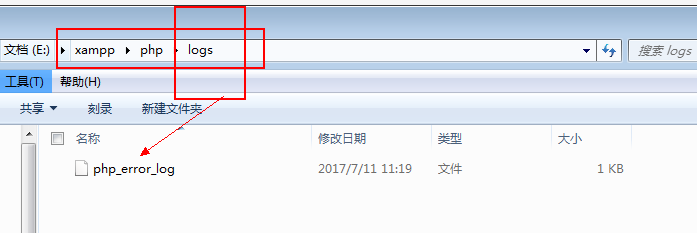


### 注意：如果想要日志发送成功，必须满足：

* **1.php.ini中的配置log\_errors必须开启，默认开启**



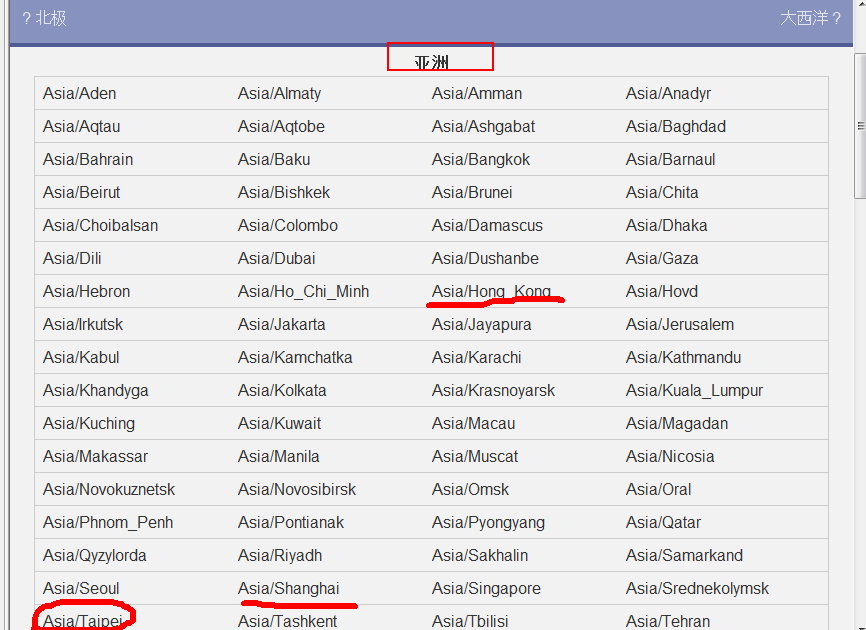
* 2.错误日志文件必须和对应目录下的文件名一致。注意：没有后缀名
* 3.注意：默认好像你的环境下面没有这个文件和目录



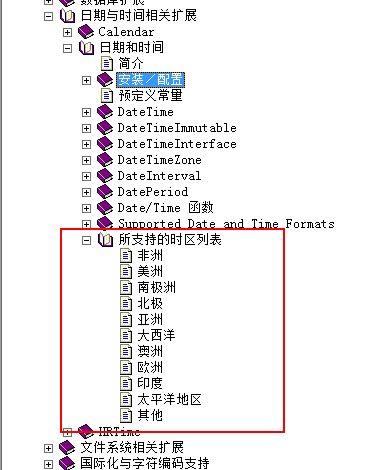
# 日期&时间

## 时区

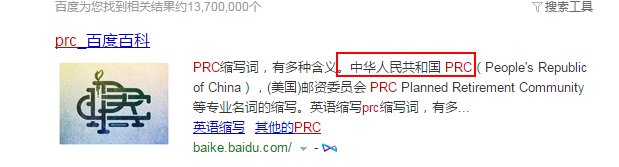
* 拿到php环境的那一刻，第一件事情，就是修改时区。
* 如何修改？可以修改配置文件中的：
  + date.timezone=
  + 方式有两种：
    - 1.洲/地区

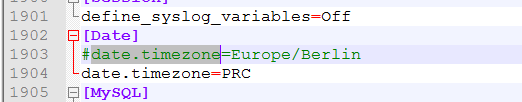


* 支持的时区列表：



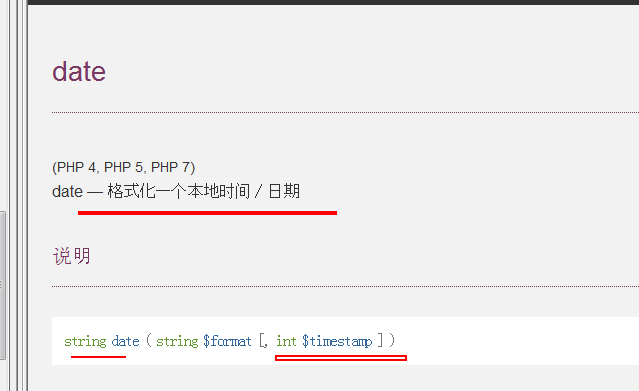
* + - 2.国的缩写—推荐





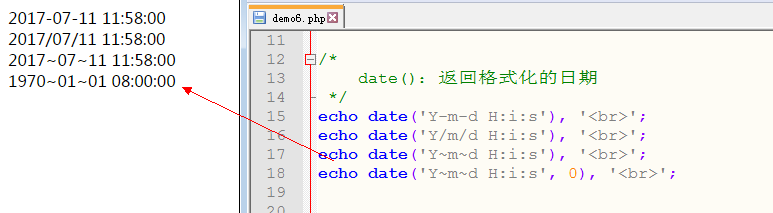
## 常用函数

### Date(格式，[可选时间戳])

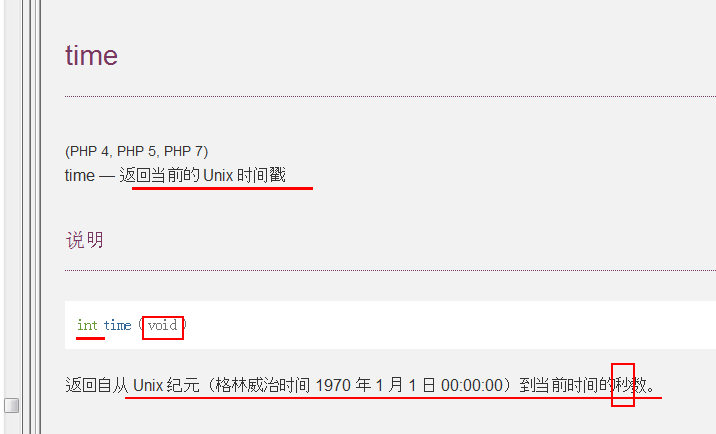


#### 常用的format格式：

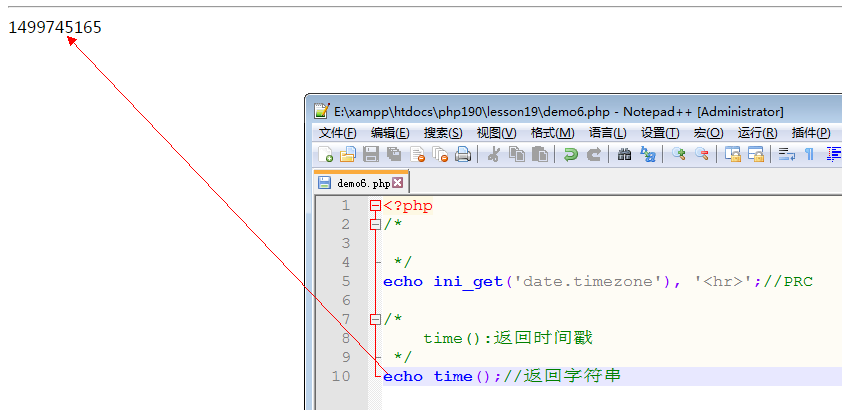
* **Y:4位年**
* **‘m:2位月**
* **‘d:2位日**
* **H:2位时，24小时**
* **‘i:2位分**
* **‘s:2位秒**



### Time()



**时间戳：就是从时间元年(1970.1.1.00：00：00)到现在的秒数，就是时间戳**



* **对于时间的存储，更多的时候用时间戳，然后取出的时候进行格式化**

### 下来自己找的函数？给一个时间，求出昨天，前天，上周，去年某一天的时间。

## 小例子：日历



### 思路：

1. 对应的月有多少天
2. 对应的月有多少周
3. 这个月第一天是星期几