正则表达式

> *作者：西岭*
> *Email：xiling3105@163.com*
> *指导：燕十八*
> *出自：布尔教育--高级PHP工程师培训*

第1章、认识正则

其实就是字符串规则表达式

```
$str = 'hi,this is his history';

$patt = '/hi/';

preg_match_all($patt,$str,$matches);

print_r($matches);
```

程序员都会用到，但是平常用的不多，所以容易忘；

入手：找谁？怎么找？找几个？

- 具体字符 (字面值)
- 字符边界
- 字符集合[ace],[01235689]
- 字符补集[^ qxz]:不在qxz 范围内
- 字符范围[a-z0-9]
- 字符簇(系统定义好的常用集合)

字符边界

- ^ 匹配字符串的开始
- $ 匹配字符串的结尾
- \b 匹配单词的开始和结尾(边界)
- \B匹配单词的非边界

第2章、常用字符簇：

簇	代表
.(点)	任意字符,不含换行
\w	[a-z A-Z 0-9_]
\W	\w 的补集
\s	空白符,包括 \n\r\t\v 等
\S	非空白符
\d	[0-9]
\D	非数字

第3章、单词匹配

```php
// 把字符串的hi单词找出来
// 规律, 单词开始处-->hi--->单词结束处 \b
$patt = '/\bhi\b/';
$str = 'hi, this is some history book';
preg_match_all($patt, $str, $res);
print_r($res);

// 把包含在单词内部的hi找出来
$patt = '/\Bhi\B/';
$str = 'hi, this is some history book';
preg_match_all($patt, $str, $res);
print_r($res);
```

第4章、集合与补集示例

```php
/*
给定一组手机号,必须由[01235689]组成的,才选出来
从哪儿找? 从字符串的开始找,找到字符串的结束 ^ $
找谁[01235689]
找几个? 11个
*/
$patt = '/^[01235689]{11}$/';

$patt = '/^[^47]{11}$/';

$arr = array('13800138000','13426060134','170235','18289881234568782');
foreach($arr as $v) {
    preg_match_all($patt, $v, $res);
    print_r($res);
}
```

第5章、字符范围

```php
// 试着找纯字母组成的单词?
$str = 'o2o, b2b, hello,world,  that';
//$patt = '/\b[a-zA-Z]{1,}\b/';  //{1,}最少1个字母
$patt = '/\b[a-zA-Z]+\b/';
preg_match_all($patt, $str, $res);
print_r($res);
```

第6章、字符簇

就是系统规定好的表示方法

```php
// 把单词拆开
$str = 'tommorw is    another day , o2o , you dont bird me i dont bird
you';
$patt = '/\W{1,}/';  // \W \w[a-zA-Z0-9]的补集
//preg_split 通过正则表达式，分割字符串
print_r(preg_split($patt, $str));
// 把多个空格或制表换成1个空格
$str = 'a    b       heloo          world'; // 'a b hello world';
$patt = '/\s{1,}/'; //\s空白符,包括 \n\r\t\v 等
//preg_replace – 执行一个正则表达式的搜索和替换
echo preg_replace($patt, ' ', $str);
```

第7章、找几个

* 匹配前面的子表达式零次或多次。
+ 匹配前面的子表达式一次或多次。
\? 匹配前面的子表达式零次或一次。
{n} n 是一个非负整数。匹配确定的 n 次。
{n,m} m 和 n 均为非负整数，其中n <= m
最少匹配 n 次且最多匹配 m 次。。
{n,} n 是一个非负整数。至少匹配n 次。

```
// 5个字母组成的单词
$patt = '/[a-zA-Z]{5}/';
// 3-5个字母组成的单词
$patt = '/[a-zA-Z]{3,5}/';
// 5个以上字母组成的单词
$patt = '/[a-zA-Z]{5,}/';

/*
某编辑部,键盘坏了,0键弹不出来,经常打出多个0
于是god常用打成good,goood,gooood,
请把这些单词替换为god
*/

$s = 'goooood,goood,gooooooooooood';
$p = '/go+d/';
print_r(preg_replace($p,'god',$s));
```

第8章、或者的用法

```
// 查询纯数字或纯字母的词
$str = 'hello o2o 2b9 250';
$patt = '/\b[a-zA-Z]+\b|\b[0-9]+\b/';
preg_match_all($patt, $str, $res);
print_r($res);

// 查询苹果系列的产品
$str = 'ipad,iphone,imac,ipod,iamsorry';
$patt= '/\bi(pad|phone|mac|pod)\b/';
preg_match_all($patt, $str, $res);
print_r($res);
```

第9章、贪婪与非贪婪

```
$str = 'ksda good  goooood good kl s ja dfs dk '
// 把g(任意多的内容)d 这样的字符串,换成god
$patt = '/g.+d/'; // 默认贪婪模式（会尽量多的匹配）
preg_match_all($patt, $str, $res);
print_r($res); // god is not good
$patt = '/g.+?d/'; //在数量(+ * {n,})限定符后,加?,非贪婪模式
preg_match_all($patt, $str, $res);
print_r($res); // god , good
```

第10章、采集手机号

```
$str = '王先生,卖洗衣机 联系13800138000, 备用电话18902587413, QQ:258963,emai
l:wang@qq.com, 诚心急卖,身份证号:101101197912123039';
// 采集电话号码
$patt = '/\b1[358]\d{9}\b/';
preg_match_all($patt, $str, $res);
print_r($res);
```

第11章、后向引用

找首尾字母相同的单词

```
$str = 'txt hello, high, bom , mum';
// 简化,先找首尾字母都是t的
$patt = '/\bt\w+t\b/';
preg_match_all($patt, $str, $res);
print_r($res);

此方法重复26次，也能找到

// 第n个小括号内内的子表达式,命中的内容,后面就用\n 来引用
// 后向引用
$patt = '/\b([a-z])\w+\1\b/';  //两种情况nginx($1)、php (\1)
preg_match_all($patt, $str, $res);
print_r($res);
```

把手机号中间4位替换为*

```
// 假设给的全是手机号
$str = '13800138000 , 13426060134 ';
$patt = '/(\d{3})\d{4}(\d{4})/';
preg_match_all($patt, $str, $res);
print_r($res);
echo preg_replace($patt, '\1****\2', $str);
```

第12章、模式

模式修饰符,可以一定程度上影响正则的解析行为
比如i, 就代表正则不区分大小写, /[a-z A-Z]+/ --->/[a-z]+/i
比如s, 单行模式, 就代表把整个文件看成一个"单行"

```
$str = 'hello WORLD, ChINa';
//$patt = '/\b[a-z]+\b/'; //hello
$patt = '/\b[a-z]+\b/i'; # 忽略大小写
preg_match_all($patt, $str, $matches);
print_r($matches);


$str = "abc haha
abc dgh";
$patt = '/.+/s'; # single 单行模式,将所有内容看成一整行
preg_match_all($patt, $str, $matches);
print_r($matches);
```

```
// u 模式,把传入的参数看成是unicode字符集的编码,可以判断中文
// http://blog.sina.com.cn/s/blog_640937d101017pca.html
// PHP下正则匹配中文, u模式, \x{4e00}-\x{9fa5}

$str = 'bob李';
$patt = '/^[\x{4e00}-\x{9fa5}]+$/u';
echo preg_match($patt, $str) ? '纯中文' : '杂货';
```

第13章、预查(选学)

```php
// 把ing结尾的单词词根部分(即不含ing部分)找出来

$str = 'hello ,when i am working , don not coming';

// 零宽度(没有消耗字符)
//正预测 (是什么样, 负预测--不是什么)
//前瞻 (往前看)
//断言(判断会是什么)
//$patt = '/\b(\w+)ing\b/';
//$patt = '/\b\w+(?=ing)\b/'; // 语义矛盾,没有谁后面是ing,同时又是\b
$patt = '/\b\w+(?=ing\b)/';
preg_match_all($patt, $str, $matches);
print_r($matches);

// 把不是ing结尾的单词找出来
// 零宽度 负预测 前瞻 断言
$patt = '/\b\w+(?!ing)\w{3}\b/';
preg_match_all($patt, $str, $matches);
print_r($matches);


// 把un开头的单词词根找出来
// 零宽度 正预测 回顾(返回来看) 断言
$str = 'luck ,unlucky, state , unhappy';
$patt = '/(?<=\bun)\w+\b/';
preg_match_all($patt, $str, $matches);
print_r($matches);

// 把非un开头的单词找出来
// 零宽度 负预测 回顾 断言
$patt = '/\b\w{2}(?<!un)\w*\b/';
preg_match_all($patt, $str, $matches);
print_r($matches);
```

第14章、常用正则及练习题

常用正则：http://www.zixue.it/thread-10221-1-1.html

练习题:

1. email(验证,采集 , $str , $html ,' /???/')
2. 验证用户输入的时间是否为 yyyy-mm-dd hh:ii:ss
3. 清除一个页面上的所有script代码,和onclick,onready等事件代码
4. 把网页的链接换成#,空连接
5. 正则分析文件后缀
6. 采集163新闻标题和内容,并过滤html标签(单行模式,贪婪模式)

第15章、PHP利用CURL实现网络请求

libcurl可以使用URL的语法模拟浏览器来传输数据，因为它是模拟浏览器，因此它同样支持多种协议，libcurl目前支持http、https、ftp、gopher、telnet、dict、file和ldap协议。libcurl同时也支持HTTPS认证、HTTP POST、HTTP PUT、 FTP 上传(这个也能通过PHP的FTP扩展完成)、HTTP 基于表单的上传、代理、cookies和用户名+密码的认证。下载文件断点续传，上传文件断点续传，http代理服务器管道，甚至它还支持IPv6，scoket5代理服务器，通过http代理服务器上传文件到FTP服务器等等。

操作系统要安装libcurl，系统提供接口供PHP调用，PHP内部实现相应的方法或者函数来供程序猿使用

添加讲解内容：外挂方式添加PHP模块(若此课程在linux后)

使用的步骤:
1.初始化，创建一个cURL资源
2.设置URL和相应的选项
3.抓取URL返回值并处理
4.关闭cURL并释放系统资源

用CURL写一个GET请求

```
  //1、初始化
$ch = curl_init();

//2、设置选项，包括URL
curl_setopt($ch, CURLOPT_URL, "http://localhost/18/ze/c.php?id=1");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_HEADER, 0);

//3、执行并获取返回内容
$output = curl_exec($ch);

//4、释放curl句柄
curl_close($ch);

//打印获得的数据
print_r($output);
```

以上方式获取到的数据是json格式的，使用json_decode函数转译成数组。

```
$output_array = json_decode($output,true);
```

如果使用json_decode($output)解析的话，将会得到object类型的数据。

其中第二步最为关键，可以设置一些高级选项,就可以完成相应的功能

//用CURL写一个POST请求

```
  $url = "http://localhost/18/ze/post.php";
$post_data = array ("username" => "bob","key" => "12345");

$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
// post数据
curl_setopt($ch, CURLOPT_POST, 1);
// post的变量
curl_setopt($ch, CURLOPT_POSTFIELDS, $post_data);

$output = curl_exec($ch);
curl_close($ch);

//打印获得的数据
print_r($output);
```

效率方面，curl的原理是模拟浏览器的操作，它的效率要比file_get_contents()高出四倍以上,稳定性也远比file_get_contents()函数要好很多