<https://www.cnblogs.com/zery/p/3438845.html>

每个表达试加/ /

/^buct$/

## 匹配规则：

**^ 匹配一行字符串的开头**

**$ 匹配一行字符串的结尾**

^once表示只匹配一once开头的字符

buct$表示只匹配一buct结尾的字符

精确匹配:

^buct$只匹配buct

如果一个模式不包括^和$，那么它与任何包含该模式的字符串匹配

once 表示匹配任何包括once的字符串

## 转义：\

特殊字符需要转义：\* . ? + $ ^ [ ] ( ) { } | \ /





## 量词

**\*重复0次或无数次**

**+重复一次或无数次**

**？重复0次或1次**

**{n}重复n次**

**{n,m}重复n到m次**

**{n,}重复n次或更多**

**1.\b不消耗任何字符只匹配一个位置，用于匹配单词边界**

This is test想匹配is

/^\bis\b$/

\b不会匹配is两边的字符，但他会匹配两边是否为单词的边界

2.\d匹配数字

匹配0开头的四位数字

/^\0\d\d\d\d$/

3.\w匹配字母、数字、下划线

匹配asdWewe22\_QWE

/^\w+$/

4.\s匹配空格

匹配a b c

/^\w\s\w\s\w/

5..匹配除换行符以外的任何字符

在\w的基础上增加一个匹配空格

6.[]匹配括号内的元素的字符

匹配a-z

[a-z]

## 反义

字母大写与原来的意思相反

"\W" 匹配任意不是字母，数字，下划线 的字符

"\S" 匹配任意不是空白符的字符

　"\D" 匹配任意非数字的字符

"\B" 匹配不是单词开头或结束的位置

"[^abc]" 匹配除了abc以外的任意字符

## 贪婪与懒惰

.\* 匹配尽可能多的字符

a.\*b aabab

.? 匹配尽可能少的字符

a.\*?b aab 或 ab

使用 1.\*1 将会匹配到1010000000001, 匹配方法: 先匹配至输入串A的最后, 然后向前匹配, 直到可以匹配到1, 称之为贪婪匹配。

使用 1.?1 将会匹配到101, 匹配方法: 匹配下一个1之前的所有字符, 称之为非贪婪匹配。

2.4 懒惰限定符

　　"\*?" 重复任意次，但尽可能少重复

　　 如 "acbacb" 正则 "a.\*?b" 只会取到第一个"acb" 原本可以全部取到但加了限定符后，只会匹配尽可能少的字符 ，而"acbacb"最少字符的结果就是"acb"

　　"+?" 重复1次或更多次，但尽可能少重复

　　 与上面一样，只是至少要重复1次

　　"??" 重复0次或1次，但尽可能少重复

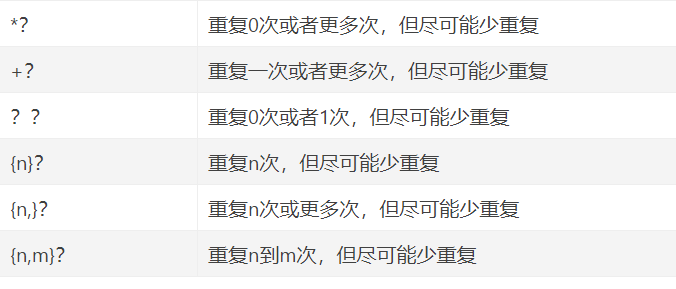
　　 如 "aaacb" 正则 "a.??b" 只会取到最后的三个字符"acb"

　　"{n,m}?" 重复n到m次，但尽可能少重复

如 "aaaaaaaa" 正则 "a{0,m}" 因为最少是0次所以取到结果为空

　　"{n,}?" 重复n次以上，但尽可能少重复

如 "aaaaaaa" 正则 "a{1,}" 最少是1次所以取到结果为 "a"



## 常见正则表达式

Php验证正则preg\_match（/^1[358]\d{9}$/,$phone）

**电话**

/^1[358]\d{9}$/

**邮箱**

（字母 、数字、下划线、连字符组成 的用户名和@与字线数字组成的主机域名）

**^[a-zA-Z0-9\_\-]+@[a-zA-Z0-9\- ]+\. [a-zA-Z0-9\-\.]+$**

**身份证号码（18位数字或17位数字后面加X或加Y字母组成）**

^[0-9]{17}([0-9]|X|Y)$

## 分组

使用小括号分开正则规则每一个小括号就是一个分组，每个组会自动拥有一个组号，从左向右1、2、3

如  "zery zery" 正则 \b(\w+)\b\s\1\b 所以这里的"\1"所捕获到的字符也是 与(\w+)一样的"zery"

为了让组名更有意义，组名是可以自定义名字的

"\b(?<name>\w+)\b\s\k<name>\b"

| 匹配左右任意一个表达式

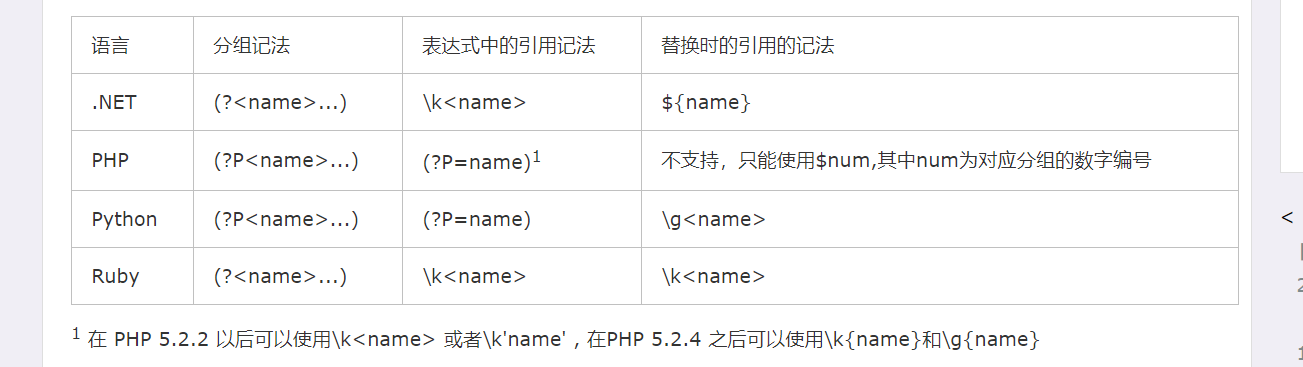
(ab) 将括号中的字符作为一个分组

\num 引用分组num匹配到的字符串

(?P<name>) 分组起别名

(?P=name) 引用别名为name分组匹配到的字符串

命名分组



**"(exp)"**    匹配exp,并捕获文本到自动命名的组里

**"(?<name>exp)"**   匹配exp,并捕获文本到名称为name的组里

**"(?:exp)"**  匹配exp,不捕获匹配的文本，也不给此分组分配组号

以下为零宽断言

**"(?=exp)"**  匹配exp前面的位置

　　如 "How are you doing" 正则"(?<txt>.+(?=ing))" 这里取ing前所有的字符，并定义了一个捕获分组名字为 "txt" 而"txt"这个组里的值为"How are you do";

**"(?<=exp)"**  匹配exp后面的位置

　　如 "How are you doing" 正则"(?<txt>(?<=How).+)" 这里取"How"之后所有的字符，并定义了一个捕获分组名字为 "txt" 而"txt"这个组里的值为" are you doing";

**"(?!exp)"**  匹配后面跟的不是exp的位置

　　如 "123abc" 正则 "\d{3}(?!\d)"匹配3位数字后非数字的结果

**"(?<!exp)"**  匹配前面不是exp的位置

　　如 "abc123 " 正则 "(?<![0-9])123" 匹配"123"前面是非数字的结果也可写成"(?!<\d)123"