Python 正则表达式入门（初级篇）

Python 正则表达式入门（初级篇）

本文主要为没有使用正则表达式经验的新手入门所写。

转载请写明出处

引子

首先说 正则表达式是什么？

> 正则表达式，又称正规表示式、正规表示法、正规表达式、规则表达式、常规表示法（英语：Regular Expression，在代码中常简写为regex、regexp或RE），计算机科学的一个概念。正则表达式使用单个

字符串来描述、匹配一系列匹配某个句法规则的字符串。在很多文本编辑器里，正则表达式通常被用来检索、替换那些匹配某个模式的文本。
许多程序设计语言都支持利用正则表达式进行字符串操作。例如，在Perl中就内建了一个功能强大的正则表达式引擎。正则表达式这个概念最初是由Unix中的工具软件（例如sed和grep）普及开的。正则表达式通常缩写成“regex”，单数有regexp、regex，复数有regexps、regexes、regexen。
引用自维基百科https://zh.wikipedia.org/wiki/%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F

定义是定义，太正经了就没法用了。我们来举个栗子：假如你在写一个爬虫，你得到了一个网页的HTML源码。其中有一段

```
<html><body><h1>hello world<h1></body></html>
```

你想要把这个hello world提取出来，但你这时如果只会python 的字符串处理，那么第一反应可能是

```
s = <html><body><h1>hello world<h1></body></html>
start_index = s.find('<h1>')
```

然后从这个位置向下查找到下一个 `<h1>` 出现这样做未尝不可，但是很麻烦不是吗。需要考虑多个标签，一不留神就多匹配到东西了，而如果想要非常准确的匹配到，又得多加循环判断，效率太低。

这时候，正则表达式就是首选的帮手。

干货开始

< 2018年3月 >

日	一	二	三	四	五	六
25	26	27	28	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签

python(23)

入门级别

接着说我们刚才那个例子。我们如果拿正则处理这个表达式要怎么做呢？

```
import re

key = r"<html><body><h1>hello world<h1></body></html>"#这段是你要匹配的文本
p1 = r"(?<=<h1>).+?(?=<h1>)"#这是我们写的正则表达式规则，你现在可以不理解啥意思
pattern1 = re.compile(p1)#我们在编译这段正则表达式
matcher1 = re.search(pattern1,key)#在源文本中搜索符合正则表达式的部分
print matcher1.group(0)#打印出来
```

你可以尝试运行上面的代码，看看是不是和我们想象的一样（博主是在python2.7环境下）发现代码挺少挺简单？往下看。而且正则表达式实际上要比看起来的那种奇形怪状要简单得多。

首先，从最基础的正则表达式说起。

假设我们的想法是把一个字符串中的所有"python"给匹配到。我们试一试怎么做

```
import re

key = r"javapythonhtmlvhdl"#这是源文本
p1 = r"python"#这是我们写的正则表达式
pattern1 = re.compile(p1)#同样是编译
matcher1 = re.search(pattern1,key)#同样是查询
print matcher1.group(0)
```


随笔分类

随笔档案

最新评论

看完这段代码，你是不是觉得：卧槽？这就是正则表达式？直接写上去就行？

确实，正则表达式并不像它表面上那么奇葩，如果不是我们故意改变一些符号的含义时，你看到的就是想要匹配的。

所以，先把大脑清空，先认为正则表达式就是和想要匹配的字符串长得一样。在之后的练习中我们会逐步进化

初级

0.无论是python还是正则表达式都是区分大小写的，所以当你在上面那个例子上把"python"换成了"Python"，那就匹配不到你心爱的python了。

1.重新回到第一个例子中那个 `<h1>hello world<h1>` 匹配。假如我像这么写，会怎么样？

```
import re

key = r"<h1>hello world<h1>"#源文本
p1 = r"<h1>.+<h1>"#我们写的正则表达式，下面会将为什么
pattern1 = re.compile(p1)
print pattern1.findall(key)#发没发现，我怎么写成findall了？咋变了呢？
```

有了入门级的经验，我们知道那两个 `<h1>` 就是普普通通的字符，但是中间的是什么鬼？

`.` *字符在正则表达式代表着可以代表任何一个字符（包括它本身）*

findall返回的是所有符合要求的元素列表，包括仅有一个元素时，它还是给你返回的列表。

1. Re:Python 正则表达式入门（初级篇）
写的很好啊
--执小白
2. Re:银行家算法学习笔记
可以可以，简明易懂
--icelee
3. Re:卷积神经网络提取特征并用于SVM
list(map())就可以了，主要是python2和3的差别
--隐泊浮生
4. Re:卷积神经网络提取特征并用于SVM
@月明塘 遇到了同样的问题，请问你解决了吗？...
--隐泊浮生
5. Re:PyQt4入门学习笔记（一）
支持。对于了解PYQT的基本使用不错。
--豪门百里

阅读排行榜

1. Python 正则表达式入门（初级篇）(63665)
2. IDEA上安装和使用checkstyle,findbugs,visualVM,PMD插件(9862)
3. 用树莓派从零开始做一个家庭监控(9005)
4. 卷积神经网络提取特征并用于SVM(8819)
5. pycharm连接mysql数据库(8658)

机智如你可能会突然问：那我如果就只是想匹配"."呢？结果啥都给我返回了咋整？在正则表达式中有一个字符 `\` ，其实如果你编程经验较多的话，你就会发现这是好多地方的“转义符”。在正则表达式里，这个符号通常用来把特殊的符号转成普通的，把普通的转成特殊的23333（并不是特殊的“2333”，写完才发现会不会有脑洞大的想歪了）。

举个栗子，你真的想匹配"chuxiuhong@hit.edu.cn"这个邮箱（我的邮箱），你可以把正则表达式写成下面这个样子：

```
import re

key = r"afiouwehrfuichuxiuhong@hit.edu.cnaskdjhfiosueh"
p1 = r"chuxiuhong@hit\.edu\.cn"
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

发现了吧，我们在 `.` 的前面加上了转义符 `\` ，但是并不是代表匹配“\.”的意思，而是只匹配“.”的意思！

不知道你细不细心，有没有发现我们第一次用 `.` 时，后面还跟了一个 `+` ？那这个加号是干什么的呢？其实不难想，我们说了“ `.` 字符在正则表达式代表着可以代表任何一个字符（包括它本身）”，但是"hello world"可不是一个字符啊。

`+` *的作用是将前面一个字符或一个子表达式重复一遍或者多遍。*

比方说表达式“ab+”那么它能匹配到“abbbbb”，但是不能匹配到"a"，它要求你必须得有个b，多了不限，少了不行。你如果问我有没有那种“有没有都行，有多少都行的表达方式”，回答是有的。

`*` *跟在其他符号后面表达可以匹配到它0次或多次*

比方说我们在王叶内遇到了链接，可能既有http://开头的，又有https://开头的，我们怎么处理？

```
import re

key = r"http://www.nsfbuhwe.com and https://www.auhfisna.com"#胡编乱造的网址，别在意
p1 = r"https*://"#看那个星号！
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

输出

```
['http://', 'https://']
```

2.比方说我们有这么一个字符串"cat hat mat qat"，你会发现前面三个是实际的单词，最后那个是我胡编乱造的（上百度查完是昆士兰英语学院的缩写= =）。如果你本来就知道"at"前面是c、h、m其中之一时这才构成单词，你想把这样的匹配出来。根据已经学到的知识是不是会想到写出来三个正则表达式进行匹配？实际上不需要。因为有一种多字符匹方式

`[]` *代表匹配里面的字符中的任意一个*

还是举个栗子，我们发现啊，有的程序员比较过分，，在 `<html></html>` 这对标签上，大小写混用，老害得我们抓不到想要的东西，我们该怎么应对？是写16*16种正则表达式挨个匹配？no

```
import re

key = r"lalala<hTml>hello</Html>heiheihei"
p1 = r"<[Hh][Tt][Mm][Ll]>.+?</[Hh][Tt][Mm][Ll]>"
pattern1 = re.compile(p1)
```

```
print pattern1.findall(key)
```

输出

```
['<hTml>hello</Html>']
```

我们既然有了范围性的匹配，自然有范围性的排除。

`[^]` *代表除了内部包含的字符以外都能匹配*

还是cat,hat,mat,qat这个例子，我们想匹配除了qat以外的，那么就应该这么写：

```
import re

key = r"mat cat hat pat"
p1 = r"[^p]at"#这代表除了p以外都匹配
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

输出

为了方便我们写简洁的正则表达式，它本身还提供下面这样的写法

正则表达式	代表的匹配字符
[0-9]	0123456789任意之一
[a-z]	小写字母任意之一
[A-Z]	大写字母任意之一

正则表达式	代表的匹配字符
\d	等同于[0-9]
\D	等同于[^0-9]匹配非数字
\w	等同于[a-z0-9A-Z_]匹配大小写字母、数字和下划线
\W	等同于[^a-z0-9A-Z_]等同于上一条取非

3.介绍到这里，我们可能已经掌握了大致的正则表达式的构造方式，但是我们常常会在实战中遇到一些匹配的不准确的问题。比方说：

```
import re

key = r"chuxiuhong@hit.edu.cn"
p1 = r"@.+\."#我想匹配到@后面一直到“.”之间的，在这里是hit
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

输出结果

```
['@hit.edu.']
```

呦呵！你咋能多了呢？我理想的结果是 `@hit.` ，你咋还给我加量了呢？这是因为正则表达式默认是“贪婪”的，我们之前讲过，“+”代表是字符重复一次或多次。但是我们没有细说这个多次到底是多少次。所以它会尽可能“贪婪”地多给我们匹配字符，在这个例子里也就是匹配到最后一个“.”。

我们怎么解决这种问题呢？只要在“+”后面加一个“？”就好了。

```
import re

key = r"chuxiuhong@hit.edu.cn"
p1 = r"@.+?\."#我想匹配到@后面一直到“.”之间的，在这里是hit
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

输出结果

```
['@hit.']
```

加了一个“?”我们就将贪婪的“+”改成了懒惰的“+”。这对于[abc]+,\w*之类的同样适用。

小测验：上面那个例子可以不使用懒惰匹配，想一种方法得到同样的结果

个人建议：在你使用"+","*"的时候，一定先想好到底是用贪婪型还是懒惰型，尤其是当你用到范围较大的项目上时，因为很有可能它就多匹配字符回来给你！！！

为了能够准确的控制重复次数，正则表达式还提供

{a,b}(代表a<=匹配次数<=b)

还是举个栗子，我们有sas,saas,saaas，我们想要sas和saas，我们怎么处理呢？

```
import re

key = r"saas and sas and saaas"
p1 = r"sa{1,2}s"
pattern1 = re.compile(p1)
print pattern1.findall(key)
```

输出

```
['saas', 'sas']
```

如果你省略掉{1,2}中的2，那么就代表至少匹配一次，那么就等价于？

如果你省略掉{1,2}中的1，那么就代表至多匹配2次。

下面列举一些正则表达式里的元字符及其作用

元字符	说明
.	代表任意字符
|	逻辑或操作符
[ ]	匹配内部的任一字符或子表达式
[^]	对字符集和取非
-	定义一个区间

元字符	说明
\	对下一字符取非（通常是普通变特殊，特殊变普通）
*	匹配前面的字符或者子表达式0次或多次
*?	惰性匹配上一个
+	匹配前一个字符或子表达式一次或多次
+?	惰性匹配上一个
?	匹配前一个字符或子表达式0次或1次重复
{n}	匹配前一个字符或子表达式
{m,n}	匹配前一个字符或子表达式至少m次至多n次
{n,}	匹配前一个字符或者子表达式至少n次
{n,}?	前一个的惰性匹配
^	匹配字符串的开头
\A	匹配字符串开头
$	匹配字符串结束
[\b]	退格字符
\c	匹配一个控制字符
\d	匹配任意数字

元字符	说明
\D	匹配数字以外的字符
\t	匹配制表符
\w	匹配任意数字字母下划线
\W	不匹配数字字母下划线

中级篇介绍子表达式，向前向后查找，回溯引用 链接：http://www.cnblogs.com/chuxiuhong/p/5907484.html

苍生苦难，不知伊于胡底

标签: python , 正则表达式

唯心不易
关注 - 2
粉丝 - 72
+加关注

14

« 上一篇：PyQt4入门学习笔记（一）
» 下一篇：PyQt4入门学习笔记（二）

posted @ 2016-09-19 14:30 唯心不易 阅读(63667) 评论(8) 编辑 收藏

评论列表

#1楼 2017-01-13 08:49 **jason.hu** 回复 引用

通俗易懂，适合入门，赞一个

支持(0) 反对(0)

#2楼 2017-04-04 09:14 **aehyok** 回复 引用

想问一下，字符串前面为什么都有个“r”,是什么意思

支持(0) 反对(0)

#3楼 2017-05-03 11:59 **zeroonec** 回复 引用

@ aehyok
避免转义字符带来的麻烦

支持(0) 反对(0)

#4楼 2017-07-04 17:22 **Sisimi** 回复 引用

Sisimi:
天猫增添品类规则升级 商家扩大经营有据可依 摘要：6月27日天猫将生效一则新规，对天猫在营店铺扩大经营范围做出了规范，加强对店铺新增品类的标准化。 新零售引领着消费升级的到来。越来越多的商家也开始尝试调整自身货品结构，扩大经营范围的诉求日益旺盛。为了更好的服务商家，6月27日天猫将生效一则新规，对在营店铺扩大经营范围做出规范，加强店铺新增品类的标准化。在此次规则变更中，天猫依然向优质品牌、品质服务、品质商品敞开怀抱，也将根据市场需求及行业特点进行择优招募。 举例来说，某商家想申请添加A类目，如果A类目不在对应的天猫定向招商品牌库内，可尝试申请自荐添加，若品牌影响力及资质要求评估通过，便可添加成功。 除此之外，天猫将针对母婴部分类目在专营店授权链路上收紧要求，加强供应链审查，同时也紧贴政策法规变化，确保商家经营资质的实时合规，籍此为消费者把好关、站好岗，充分保障消费者的购物权益。 本次规则调整将于2017年6月27日正式生效，商家有添加品类需求且符合条件的，可以戳下面链接查看申请流程，或直接进入“商家中心-品牌和类目管理”提交申请。 天猫在营店铺新增品牌申请流程：https://service.tmall.com/support/tmall/knowledge-1124487.htm 天猫在营店铺新增类目申请流程：https:

//service.tmall.com/support/tmall/knowledge-1126642.htm?spm=a225r.8199751.0.0.RBQ7mK 天猫添品添类规范细则如下： 天猫在营店铺申请新增类目细则： https://rule.tmall.com/tdetail-5898.htm?spm=a2177.7731966.0.0.9Di4lN&tag=self 天猫在营店铺申请新增品牌细则： https://rule.tmall.com/tdetail-5900.htm?spm=a2177.7731966.0.0.9Di4lN&tag=self
正大零售业务全线接入京东到家：订单量亮眼

上面的内容怎么提取除网址外的内容

支持(0) 反对(0)

#5楼 2017-07-29 19:45 水怪怪 回复 引用

博主讲得很易懂呀~~

支持(1) 反对(0)

#6楼 2017-11-30 10:10 link啊啊啊 回复 引用

个人感觉讲的非常好 通俗易懂 很适合入门新手 创建个号 就为这篇文章推荐一波

支持(0) 反对(0)

#7楼 2018-01-06 13:44 wang_long 回复 引用

非常实用，通俗易懂。转载一波谢谢谢谢

支持(0) 反对(0)

#8楼 2018-02-25 11:21 执小白 回复 引用

写的很好啊

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

发表评论

昵称： 遥远的绿洲

评论内容：

提交评论 退出 订阅评论

[Ctrl+Enter快捷键提交]