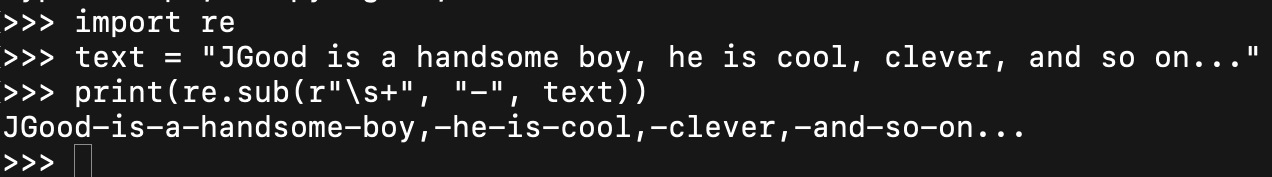
正则表达式和re.sub()的用法

re.sub()用于替换字符串中的匹配项。下面一个例子可以将字符串中的空格“ ”替换成“-”。



要想理解re.sub()的用法，首先需要理解正则表达式的语法。

1. 正则表达式基础

正则表达式并不是python的一部分，正则表达式是用于处理字符串的强大工具，拥有自己独特的语法，效率上可能不如str自带的方法，但功能十分强大。

正则表达式的大致匹配过程如下：依次拿出表达式和文本中的字符比较，如果每一个字符都能匹配，则匹配成功，一旦有匹配不成功的字符则匹配失败。如果表达式中有量词或边界，这个过程会稍微有些不同。

下图列出了python支持的正则表达式元字符和语法：

|  |  |  |  |
| --- | --- | --- | --- |
| 语法 | 说明 | 正则匹配表达式 | 匹配的字符串 |
| 一般字符 | 匹配自身 | abc | abc |
| 单个字符 | | | |
| . | 匹配任意除换行符“\n”外的字符 | a.c | abc  acc等 |
| \ | 转义字符，使后一个字符改变原来的意思。如果字符串中有字符\*需要匹配，可以使用\\*或者字符集[\*]来匹配 | a\.c | a.c |
| […] | 字符集；对应的位置可以是字符集中任意字符。字符集中的字符可以逐个列出，如[abc]，也可以给出范围[a-c]。第一个字符如果是^则表示取反，如[^abc]表示不是abc的其他字符 | a[bcd]e | abc、ace、ade |
| 预定义字符集—都可以用字符集[…]来实现 | | | |
| \d | 匹配数字0-9，相当于[0-9] | a\dc | a1c、等 |
| \D | 匹配非数字，相当于[^\d] | a\Dc | abc、acc等 |
| \s | 匹配空白字符,相当于[<空格> \t\r\n\f\v] | a\sc | a c、a c等 |
| \S | 匹配非空白字符，相当于[^\s] | a\Sc | abc、adc |
| \w | 匹配单词字符，相当与[A-Za-z0-9] | a\wc | abc等 |
| \W | 匹配非单词字符，相当与[^\w] | a\Wc | a c等 |
| 数量词—用在字符或…之后 | | | |
| \* | 匹配前一个字符0次或者无限次 | abc\* | ab、abccc等 |
| + | 匹配前一个字符一次或者无限次 | abc+ | abc、abcccc等 |
| ? | 匹配前一个字符0次或者1次 | abc? | ab、abc |
| {m} | 匹配前一个字符m次 | ab{2}c | abbc |
| {m,n} | 匹配前一个字符m至n次。m和n可以省略；若省略m，则匹配0到n次；若省略n，则匹配m至无限次 | ab{1,2}c | abc、abc |
| 边界匹配—不消耗待匹配字符串中的字符 | | | |
| ^ | 匹配字符串开头；在多行模式中匹配每一行的开头。 | ^abc | abc |
| $ | 匹配字符串末尾。  在多行模式中匹配每一行的末尾。 | abc$ | abc |
| \A | 仅匹配字符串开头 | \Aabc | abc |
| \Z | 仅匹配字符串末尾 | abc\Z | abc |
| \b | 匹配\w和\W之间  【这个其实没太懂】 | a\b!bc | a!bc |
| \B | 等价于[^\b]，也没太懂 | a\Bbc | abc |
| 逻辑、分组 | | | |
| | | |代表左右表达式任意匹配一个，  它总是先尝试匹配左边的表达式，一旦成功匹配则跳过匹配右边的表达式。  如果|没有被包括在()中，则它的范围是整个正则表达式。 | abc|def | abc  def |
| (…) | 被括起来的表达式将作为分组，从表达式左边开始每遇到一个分组的左括号,编号加1；另外，分组表达式作为一个整体，可以后接数量词。表达式中的|仅在该组中有效。 | (abc){2} | abcabc |
| (?P<name>…) | 分组，除了原有的编号外再指定一个额外的别名 | (?P<id>abc){2} | abcabc |
| \<number> | 引用编号为<number>的分组匹配到的字符串 | (\d)abc\1 | 1abc1、5abc5 |
| 特殊构造—不作为分组 | | | |
| (?:…) | (…)的不分组版本，用于使用“|”或后接数量词 | (?:abc){2} | abcabc |
| 后面还有一些特殊方式，就不一一介绍了。 |  |  |  |
| (?=…) | 之后的字符串内容需要匹配表达式才能匹配成功，不消耗字符串内容 | a(?=\d) | 匹配后面是数字的a |
| (?!...) | 之后的字符串需要不匹配表达式才能匹配成功，不消耗字符串内容 | a(?!\d) | 匹配后面不是数字的a |
| (?<=…) | 之前的字符串内容需要匹配表达式才能成功匹配。不消耗字符串内容 | (?<=\d)a | 匹配前面是数字的a |
| (?<!...) | 之前的字符串内容需要不匹配表达式才能成功匹配。不消耗字符串内容 | (?<!\d)a | 匹配前面不是数字的a |
|  |  |  |  |
|  |  |  |  |

参考网址：<https://www.cnblogs.com/huxi/archive/2010/07/04/1771073.html>

实例，去掉英语文章中的非英语单词和数字的符号，可以用以下正则表达式：

