正则表达式

第二章

目标

- 掌握正则表达式的规则

案例

- 一个小爬虫

正则表达式

第一节

milo-邹琪鲜

案例

- 一个小爬虫

正则表达式

- 正则表达式（或 RE）是一种小型的、高度专业化的编程语言，（在Python中）它内嵌在Python中，并通过 re 模块实现。
  - 可以为想要匹配的相应字符串集指定规则
  - 该字符串集可能包含英文语句、e-mail地址、TeX命令或任何你想搞定的东西
  - 可以问诸如“这个字符串匹配该模式吗？”
  - “在这个字符串中是否有部分匹配该模式呢？”
  - 你也可以使用 RE 以各种方式来修改或分割字符串。

正则表达式

- 正则表达式模式被编译成一系列的字节码，然后由用 C 编写的匹配引擎执行
- 正则表达式语言相对小型和受限（功能有限）
  - 并非所有字符串处理都能用正则表达式完成

正则表达式

- 字符匹配
  - 普通字符
    - 大多数字母和字符一般都会和自身匹配
    - 如正则表达式 test 会和字符串“test”完全匹配
  - 元字符

    . ^ $ * + ? {} [] \ | ()

正则表达式-元字符

- [ ]
  - 常用来指定一个字符集：[abc]； [a-z]
  - 元字符在字符集中不起作用：[akm$]
  - 补集匹配不在区间范围内的字符：[^5]
- ^
  - 匹配行首。除非设置 MULTILINE 标志，它只是匹配字符串的开始。在 MULTILINE 模式里，它也可以直接匹配字符串中的每个换行。
- $
  - 匹配行尾，行尾被定义为要么是字符串尾，要么是一个换行字符后面的任何位置。

正则表达式-元字符

- \
  - 反斜杠后面可以加不同的字符以表示不同特殊意义
  - 也可以用于取消所有的元字符： \[或 \\

\d 匹配任何十进制数；它相当于类 [0-9]。
\D 匹配任何非数字字符；它相当于类 [^0-9]。
\s 匹配任何空白字符；它相当于类 [\t\n\r\f\v]。
\S 匹配任何非空白字符；它相当于类 [^ \t\n\r\f\v]。
\w 匹配任何字母数字字符；它相当于类 [a-zA-Z0-9_]。
\W 匹配任何非字母数字字符；它相当于类 [^a-zA-Z0-9_]。

正则表达式-元字符

- 重复
  - 正则表达式第一功能是能够匹配不定长的字符集，另一个功能就是你可以指定正则表达式的一部分的重复次数。
- *
  - 指定前一个字符可以被匹配零次或更多次，而不是只有一次。匹配引擎会试着重复尽可能多的次数（不超过整数界定范围，20亿）
  - a[bcd]*b-- "abcbd"

正则表达式

- +
  - 表示匹配一或更多次。
  - 注意 * 和 + 之间的不同；＊ 匹配零或更多次，所以可以根本就不出现，而 + 则要求至少出现一次
- ？
  - 匹配一次或零次；你可以认为它用于标识某事物是可选的

正则表达式

- {m,n}
  - 其中 m 和 n 是十进制整数。该限定符的意思是至少有 m 个重复，至多到 n 个重复。 a/{1,3}b
  - 忽略 m 会认为下边界是 0，而忽略 n 的结果将是上边界为无穷大（实际上是20亿）
  - {0,} 等同于 *，{1,} 等同于 +，而{0,1}则与 ? 相同。如果可以的话，最好使用 *，+，或?

正则表达式

- 使用正则表达式
  - re 模块提供了一个正则表达式引擎的接口，可以让你将 REstring 编译成对象并用它们来进行匹配。
  - 编译正则表达式

```
#!python
>>> import re
>>> p = re.compile('ab*')
>>> print p
<re.RegexObject instance at 80b4150>
```

正则表达式

- re.compile() 也接受可选的标志参数，常用来实现不同的特殊功能和语法变更

```
#!python
>>> p = re.compile('ab*', re.IGNORECASE)
```

- 反斜杠的麻烦
  - 字符串前加 "r" 反斜杠就不会被任何特殊方式处理

字符	阶段
\section	要匹配的字符串
\\section	为 re.compile 取消反斜杠的特殊意义
"\\\\section"	为"\\section"的字符串实值(string literals)取消反斜杠的特殊意义

正则表达式

- 执行匹配
  - `RegexObject` 实例有一些方法和属性,完整的列表可查阅 Python Library Reference

方法/属性	作用
match()	决定 RE 是否在字符串刚开始的位置匹配
search()	扫描字符串，找到这个 RE 匹配的位置
findall()	找到 RE 匹配的所有子串，并把它们作为一个列表返回
finditer()	找到 RE 匹配的所有子串，并把它们作为一个迭代器返回

如果没有匹配到的话，match() 和 search() 将返回 None。
如果成功的话，就会返回一个 `MatchObject` 实例，

正则表达式

– MatchObject 实例方法

方法/属性	作用
group()	返回被 RE 匹配的字符串
start()	返回匹配开始的位置
end()	返回匹配结束的位置
span()	返回一个元组包含匹配 (开始,结束) 的位置

- 实际程序中，最常见的作法是将 `MatchObject` 保存在一个变量里，然後检查它是否为 None

```
#!python
p = re.compile( ... )
m = p.match( 'string goes here' )
if m:
print 'Match found: ', m.group()
else:
print 'No match
```

正则表达式

- 模块级函数
  - re 模块也提供了顶级函数调用如 match()、search()、sub() 、subn()、split()、findall()等

正则表达式

- 编译标志-flags

标志	含义
DOTALL, S	使 . 匹配包括换行在内的所有字符
IGNORECASE, I	使匹配对大小写不敏感
LOCALE, L	做本地化识别（locale-aware）匹配.法语等 "é" 或 "ç"
MULTILINE, M	多行匹配，影响 ^ 和 $
VERBOSE, X	能够使用 REs 的 verbose 状态，使之被组织得更清晰易懂

```
#!python
charref = re.compile(r"""
(
[0-9]+[^0-9]      # Decimal form
| 0[0-7]+[^0-7]   # Octal form
| x[0-9a-fA-F]+[^0-9a-fA-F] # Hexadecimal form
)
""", re.VERBOSE)
```

正则表达式

- 分组
  - "(" 和 ")"

一个小爬虫

- 下载贴吧或空间中所有图片

本节小结

- 正握正则表达式的定义方式及使用方法
- 会过滤诸如HTML标签