# 知识是进步的阶梯

“\\”是Java正则表达式中比较特殊的存在，这个符号随后跟的字符是有特殊意义的。和其他语言不一样，其他语言是什么样的，不要管了，我头都大了！只要在这个基础上在记一条，Java正则表达式表示反斜线得这么写“\\\\”

贪婪匹配模式（Greedy quantifiers）

X\* 0个或多个（管他有没有！）

X+ 1个或多个（粗暴点理解就是必须要有！）

* 字符串仅能包含数字、-、X、x

Java： [\\d-Xx]\* // \* 一定不能忘了

Common: [\d-Xx]\*

为什么那个“-”前面不需要加“\\”呢？因为 [a-z] 表示从“a”到“z”那么照道理“-”算一个特殊字符，莫非是只有被两个顺序排列的字母夹在中间才会被特殊解析？

# 实践出真知

/\*\*

\* 递归目录，删除文件名中符合正则表达式的部分

\* **@author** jojo

\* **@param** fileDirectory

\* 文件目录

\* **@param** regexArray

\* 正则表达式数组

\*/

**public** **static** **void** fileRename(File fileDirectory, String[] regexArray) {

// 若该File对象为目录，则取出该目录下的所有文件进行递归

**if** (fileDirectory.isDirectory()) {

**for** (File file : fileDirectory.listFiles()) {

*fileRename*(file, regexArray);

}

}

// 确定该File对象为文件，则进行重命名操作

String path = fileDirectory.getAbsolutePath();

**for**(String regex : regexArray){

path = path.replaceAll(regex, "");

}

fileDirectory.renameTo(**new** File(path));

}

"abc".matches("a")

String类的matches方法，照我的理解，上面应该返回true，因为a存在于abc里。但它返回false。。。。matches()方法是看整个字符串是否符合正则表达式。符合正则表达式“a”的只有字符串“a”。

"abc".matches("abc")

这个返回的就是true。我最初的思路对应的其实是matcher类的find()方法，不能把这两个概念联系在一起。

批量改名，不过这段代码还是有问题。顺序不对，目前这个是递归至文件进行改名然后再回过头对目录改名。这会出现什么情况呢，文件名我是按绝对路径获取的，如果目录本身也被正则表达式匹配到了，那么这个绝对路径中的目录部分也会被改掉，也因此，文件就放不进原有目录了。

正确做法是先对目录进行修改，然后再递归。

获取磁力链接，代码是没问题，但文件过大时就毛都读不到了。。。。

/\*\*

\* 根据正则表达式从一个HTML文件内获取自己想要的磁力链（或是电驴连接）

\*

\* **@author** jgy

\* **@param** html

\*/

**public** **static** **void** getMangetLinks(String htmlFilePath) {

// 读取文件至内存

String htmlString = *convertHtmlToString*(**new** File(htmlFilePath));

// 获取想要的内容

String content = *getContentByRegex*(htmlString);

System.***out***.println(content);

}

/\*\*

\* 根据正则表达式提取磁力链

\*

\* **@param** htmlString

\* **@param** regex

\* **@return**

\*/

**private** **static** String getContentByRegex(String htmlString) {

StringBuilder result = **new** StringBuilder();

// 这个串链随意啦

Matcher matcher = Pattern.*compile*(

"<a\\s(href=\"([\\w:\\|%\\./\\-=\u4e00-\u9fa5]\*)\")[\\s\\w\"=]\*>\u8fc5\u96f7\u002d\u7535\u9a74</a>")

.matcher(htmlString);

// 这是另一个正则表达式，用来过滤分辨率

Matcher resolutionRatioMatcher = Pattern.*compile*("1024X576").matcher("");

String temp = **null**;

**while** (matcher.find()) {

temp = matcher.group(2);

resolutionRatioMatcher.reset(temp);

**if** (resolutionRatioMatcher.find()) {

result.append(temp);

result.append("\n");

}

}

**return** result.toString();

}

/\*\*

\* 提取HTML文件的内容，并以String类对象的形式返回。

\*

\* **@author** jgy

\* **@param** htmlFile

\* **@return**

\*/

**private** **static** String convertHtmlToString(File htmlFile) {

BufferedReader reader = **null**;

String htmlString = **null**;

**try** {

reader = **new** BufferedReader(**new** FileReader(htmlFile));

StringBuilder temp = **new** StringBuilder();

**while** ((htmlString = reader.readLine()) != **null**) {

temp.append(htmlString);

}

htmlString = temp.toString();

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**if** (reader != **null**) {

**try** {

reader.close(); // 要不是为了把这玩意关掉，我压根不想加 try 块

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**return** htmlString;

}

写的有乱，不过整整重要的是这个正则

<a\\s(href=\"([\\w:\\|%\\./\\-=\u4e00-\u9fa5]\*)\")[\\s\\w\"=]\*>\u8fc5\u96f7\u002d\u7535\u9a74</a>

爬取图片路径，并直接保存至本地。