UNIX和LinuxShell正则表达式语法介绍 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/268/2021_2022_UNIX_E5_92_8CLin_c67_268979.htm 一个正则表达式就是由普通字符（例如字符 a 到 z）以及特殊字符（称为元字符）组成的文字模式。该模式描述在查找文字主体时待匹配的一个或多个字符串。正则表达式作为一个模板，将某个字符模式与所搜索的字符串进行匹配。 \ 将下一个字符标记为一个特殊字符、或一个原义字符、或一个 后向引用、或一个八进制转义符。例如，n 匹配字符 "n"。\n 匹配一个换行符。序列 \\ 匹配 "\" 而 "\(" 则匹配 "("。 ^ 匹配输入字符串的开始位置。 $ 匹配输入字符串的结束位置。 * 匹配前面的子表达式零次或多次。例如，zo* 能匹配 "z" 以及 "zoo"。 * 等价于{0,}。 匹配前面的子表达式一次或多次。例如，zo 能匹配 "zo" 以及 "zoo"，但不能匹配 "z"。 等价于 {1,}。 ? 匹配前面的子表达式零次或一次。例如，"do(es)?" 可以匹配 "do" 或 "does" 中的"do" 。? 等价于 {0,1}。 {n} n 是一个非负整数。匹配确定的 n 次。例如，o{2} 不能匹配 "Bob" 中的 o，但是能匹配 "food" 中的两个 o。 {n,} n 是一个非负整数。至少匹配n 次。例如，o{2,} 不能匹配 "Bob" 中的 o，但能匹配 "foooood" 中的所有 o。o{1,} 等价于 o 。o{0,} 则等价于 o*。 {n,m} m 和 n 均为非负整数，其中n ? 当该字符紧跟在任何一个其他限制符 (*, , ?, {n}, {n,}, {n,m}) 后面时，匹配模式是非贪婪的。非贪婪模式尽可能少的匹配所搜索的字符串，而默认的贪婪模式则尽可能多的匹配所搜索的字符串。例如，对于字符串 "oooo"，o ? 将匹配单个 "o"，而 o 将匹配所有 o。 . 匹配除 "\n" 之外的任何单个字符。要匹配包括 \n 在

内的任何字符，请使用象 [.\n] 的模式。 (pattern) 匹配pattern 并获取这一匹配。所获取的匹配可以从产生的 Matches 集合得到，在VBScript 中使用 SubMatches 集合，在Visual Basic Scripting Edition 中则使用 $0…$9 属性。要匹配圆括号字符，请使用 \(或 \)。 (?:pattern) 匹配 pattern 但不获取匹配结果，也就是说这是一个非获取匹配，不进行存储供以后使用。这在使用 "或" 字符 (|) 来组合一个模式的各个部分是很有用。例如， industr(?:y|ies) 就是一个比 industry|industries 更简略的表达式。 (?=pattern) 正向预查，在任何匹配 pattern 的字符串开始处匹配查找字符串。这是一个非获取匹配，也就是说，该匹配不需要获取供以后使用。例如， Windows (?=95|98|NT|2000) 能匹配 "Windows 2000" 中的 "Windows" ，但不能匹配 "Windows 3.1" 中的 "Windows"。预查不消耗字符，也就是说，在一个匹配发生后，在最后一次匹配之后立即开始下一次匹配的搜索，而不是从包含预查的字符之后开始。 (?!pattern) 负向预查，在任何不匹配Negative lookahead matches the search string at any point where a string not matching pattern 的字符串开始处匹配查找字符串。这是一个非获取匹配，也就是说，该匹配不需要获取供以后使用。例如Windows (?!95|98|NT|2000) 能匹配 "Windows 3.1" 中的 "Windows"，但不能匹配 "Windows 2000" 中的 "Windows"。预查不消耗字符，也就是说，在一个匹配发生后，在最后一次匹配之后立即开始下一次匹配的搜索，而不是从包含预查的字符之后开始。 x|y 匹配 x 或 y。例如，z|food 能匹配 "z" 或 "food"。(z|f)ood 则匹配 "zood" 或 "food"。 [xyz] 字符集合。匹配所包含的任意一个字符。例如， [abc] 可以匹配 "plain" 中的 a。 [^xyz] 负值字符

集合。匹配未包含的任意字符。例如， [^abc] 可以匹配 "plain" 中的p。 [a-z] 字符范围。匹配指定范围内的任意字符。例如，[a-z] 可以匹配 a 到 z 范围内的任意小写字母字符。[^a-z] 负值字符范围。匹配任何不在指定范围内的任意字符。例如，[^a-z] 可以匹配任何不在 a 到 z 范围内的任意字符。\b 匹配一个单词边界，也就是指单词和空格间的位置。例如， er\b 可以匹配"never" 中的 er，但不能匹配 "verb" 中的 er。\B 匹配非单词边界。er\B 能匹配 "verb" 中的 er，但不能匹配 "never" 中的 er。 \cx 匹配由x指明的控制字符。例如， \cM 匹配一个 Control-M 或回车符。 x 的值必须为 A-Z 或 a-z 之一。否则，将 c 视为一个原义的 c 字符。 \d 匹配一个数字字符。等价于 [0-9]。 \D 匹配一个非数字字符。等价于 [^0-9]。 \f 匹配一个换页符。等价于 \x0c 和 \cL。 \n 匹配一个换行符。等价于 \x0a 和 \cJ。 \r 匹配一个回车符。等价于 \x0d 和 \cM。 \s 匹配任何空白字符，包括空格、制表符、换页符等等。等价于 [\f\n\r\t\v]。 \S 匹配任何非空白字符。等价于 [^ \f\n\r\t\v]。 \t 匹配一个制表符。等价于 \x09 和 \cI。 \v 匹配一个垂直制表符。等价于 \x0b 和 \cK。 \w 匹配包括下划线的任何单词字符。等价于[A-Za-z0-9_]。 \W 匹配任何非单词字符。等价于 [^A-Za-z0-9_]。 \xn 匹配 n，其中 n 为十六进制转义值。十六进制转义值必须为确定的两个数字长。例如， \x41 匹配 "A"。\x041 则等价于 \x04 &amp. "1"。正则表达式中可以使用 ASCII 编码。. \num 匹配 num，其中 num 是一个正整数。对所获取的匹配的引用。例如，(.)\1 匹配两个连续的相同字符。 \n 标识一个八进制转义值或一个后向引用。如果 \n 之前至少 n 个获取的子表达式，则 n 为后向引用。否则，如果 n 为

八进制数字 (0-7)，则 n 为一个八进制转义值。 \nm 标识一个八进制转义值或一个后向引用。如果 \nm 之前至少有is preceded by at least nm 个获取得子表达式，则 nm 为后向引用。如果 \nm 之前至少有 n 个获取，则 n 为一个后跟文字 m 的后向引用。如果前面的条件都不满足，若 n 和 m 均为八进制数字 (0-7)，则 \nm 将匹配八进制转义值 nm。 \nml 如果 n 为八进制数字 (0-3)，且 m 和 l 均为八进制数字 (0-7)，则匹配八进制转义值 nml。 \un 匹配 n，其中 n 是一个用四个十六进制数字表示的 Unicode 字符。例如， \u00A9 匹配版权符号 (?)。