

Matching via Character classes



Victor Grazi

ORACLE JAVA CHAMPION, SPEAKER AND GEEK

@vgrazi



Introduction



Predefined character classes `\d`, `\s`, `\w`
and their inverses `\D`, `\S`, `\W`

Defining your own character classes

Ranges of characters

Negation of home made character classes

Unions and intersections of character classes

Alternation



POSIX character classes (US-ASCII only)

<code>\p{Lower}</code>	A lower-case alphabetic character: [a-z]
<code>\p{Upper}</code>	An upper-case alphabetic character: [A-Z]
<code>\p{ASCII}</code>	All ASCII: [\x00-\x7F]
<code>\p{Alpha}</code>	An alphabetic character
<code>\p{Digit}</code>	A decimal digit: [0-9]
<code>\p{Alnum}</code>	An alphanumeric character
<code>\p{Punct}</code>	Punctuation:
<code>\p{Graph}</code>	A visible character: [\p{Alnum}\p{Punct}]
<code>\p{Print}</code>	A printable character: [\p{Graph}\x20]
<code>\p{Blank}</code>	A space or a tab: [\t]
<code>\p{Cntrl}</code>	A control character: [\x00-\x1F\x7F]
<code>\p{XDigit}</code>	A hexadecimal digit: [0-9a-fA-F]
<code>\p{Space}</code>	A whitespace character: [\t\n\x0B\f\r]

java.lang.Character classes (simple java character type)

<code>\p{javaLowerCase}</code>	Equivalent to java.lang.Character.isLowerCase()
<code>\p{javaUpperCase}</code>	Equivalent to java.lang.Character.isUpperCase()
<code>\p{javaWhitespace}</code>	Equivalent to java.lang.Character.isWhitespace()
<code>\p{javaMirrored}</code>	Equivalent to java.lang.Character.isMirrored()

Classes for Unicode scripts, blocks, categories and binary properties

<code>\p{IsLatin}</code>	A Latin script character (script)
<code>\p{InGreek}</code>	A character in the Greek block (block)
<code>\p{Lu}</code>	An uppercase letter (category)
<code>\p{IsAlphabetic}</code>	An alphabetic character (binary property)
<code>\p{Sc}</code>	A currency symbol
<code>\P{InGreek}</code>	Any character except one in the Greek block (negation)
<code>[\p{L}&&[^\p{Lu}]]</code>	Any letter except an uppercase letter (subtraction)



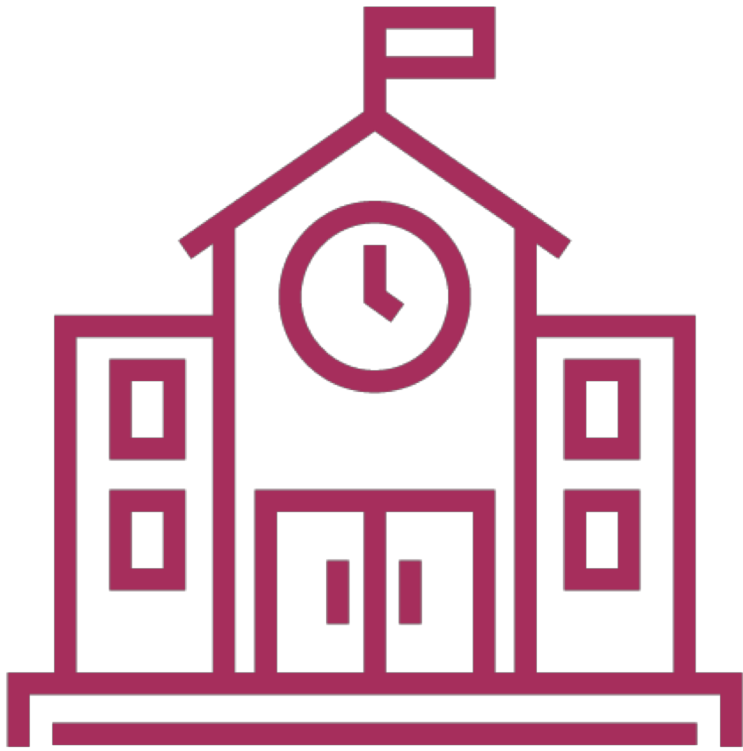


Numeric digits: 0123456789

“Word” characters

Alphanumeric or “_”

Whitespace (“ ”, tabs, or new lines)



Escaped characters \

Character set []

Grouping ()

Concatenation

Anchoring ^\$

Alternation |



```
public void findFile() {  
    File dir = new File(".");  
    FileFilter fileFilter = new RegexFileFilter  
        ("^.*[mM]y-file(-\\d+)?\\.java$");  
    File[] files = dir.listFiles(fileFilter);  
    for (int i = 0; i < files.length; i++) {  
        System.out.println(files[i]);  
    }  
}
```



replaceAll()

Replaces all occurrences of the supplied regex

```
"one + one = 2".replaceAll("one", "1")  
> 1 + 1 = 2
```

replaceFirst()

Replaces just the first occurrence of the supplied regex

```
"one + one = 2".replaceFirst("one", "1")  
> 1 + one = 2
```



Type	Example	RegEx
Digit:	1	\d
whiteSpace:	(Tab, space, new line)	\s
Word character	Alphanumeric or underscore (_)	\w
Anything but a Digit:	a	\D
Anything but a whiteSpace:	1	\S
Anything but a Word character	-	\W



String class Regex support

matches()

split()

replaceAll()

replaceFirst()

*** But not replace()**



Summary



File searching

Validation

Text searching

Text extraction

Expression reformulating

Text cleansing



Summary



Predefined character classes `\d`, `\s`, `\w`
and their inverses `\D`, `\S`, `\W`

Defining your own character classes

Ranges of characters

Negation of home made character classes

Unions and intersections of character classes

Alternation

