Regular Expressions

Basic regexes

```
while
```

- Matches the string while
- “while” in double quotes does the same thing.

Basic regexes

`/while/`

- Matches any word that has while as a substring
  - *YES* : “`while`", “`whiletrue`", “`swhile`", “`whileN=True`"
  - *NO* : “`shile`", “`whi le`", “`WhIlE`", “`RandomString`", …

Basic regexes

```
[while]
```

- Matches any of the letters w, h, i, l or e.

Quantifiers: * + ?

***** means 0 or more occurrences

- `/abc*/` matches "ab", "abc", "abcc", "abccc", ...
- `/a(bc)*/"` matches "a", "abc", "abcbc", "abcbcbc", ...
- `/a.*a/` matches "aa", "aba", "a8qa", "a!?_a", ...

+ means 1 or more occurrences

- `/a(bc)+/` matches "abc", "abcbc", "abcbcbc", ...
- `/smoo+th/` matches “smooth", “smoooth", “smoooooooooooth", ...

? means 0 or 1 occurrences

- `/coded?/` matches lines with “code" or “coded"
- `/Dan(iel)?/` matches lines with "Dan" or "Daniel"

Character sets

`[ ]` group characters into a *character set*;
will match any single character from the set

- `/[bcd]art/` matches lines with "bart", "cart", and "dart"
- equivalent to `/(b|c|d)art/` but shorter

- inside `[ ]`, most modifier keys act as normal characters
  - `/what[.!*?]*/` matches "what", "what.", "what!", "what?**!", ...

Quick Quiz : Match letter grades e.g. A+, B-, D.
"[ABCDF][+\-]?"

Basic regexes

```
[while]+
```

- Matches any word that only contains the letters w, h, i, l or e.
- Yes: we, he, ill, www, hi, eel,
- No: empty string, whiles

Character ranges

- inside a character set, specify a range of chars with -

  `/[a-z]/` matches any lowercase letter

  `/[a-zA-Z0-9]/` matches any letter or digit

- an initial ^ inside a character set negates it

  `/[^abcd]/` matches any character but a, b, c, or d

  /^[^A]/ - Match a string that does not start with A

- inside a character set, `-` must be escaped to be matched

  `/[\-+]?[0-9]+/` matches optional - or +, followed by at least one digit

Wildcards and anchors – Theses will help with #DRBC and comments

`.` (a dot) matches any character except `\n`

`/.oo.y/` matches `"Doocy"`, `"goofy"`, `"LooPy"`, …

use `\.` to literally match a dot `.` Character

`^` matches the beginning of a line; `$` the end

`/^if$/` matches lines that consist entirely of `if`

`\<` demands that pattern is the beginning of a *word*;

`\>` demands that pattern is the end of a word

`/\<for\>/` matches lines that contain the word `"for"`

Special characters

| means OR

- `/abc|def|g/` matches lines with `"abc"`, `"def"`, or `"g"`
- precedence: `^Subject|Date:` vs. `^(Subject|Date):`
- There's no AND & symbol. Why not?

( ) are for grouping

- `/(Homer|Marge) Simpson/` matches lines containing `"Homer Simpson"` or `"Marge Simpson"`

\ starts an escape sequence

- many characters must be escaped: `/ \ $ . [ ] ( ) ^ * + ?`
- `"\.\\n"` matches lines containing `".\n"`

Replacing with back-references

- you can use back-references when replacing text:
  - refer to captures as ***$number*** in the replacement string
  - Example: to swap a last name with a first name:

```
var name = “Quill,    Peter";
name = name.replace(/(\w+),\s+(\w+)/, "$2 $1");
// “Peter Quill"
```

- *Quick Quiz* : Reformat phone numbers from 250-478-8048 format to (250) 478.8048 format.

A Few “Got-Ya”s

[+-*/]

[+\-*/] – You need to escape the -

[\+\-*\/] – To make it consistent, I would escape all of the chars.

Flex always matches the longest string, so “.*”

The “cat” sat on “the” mat.

Summary

dog	matches the string "dog"
[dog]	matches matches one character: a "d" an "o" or a "g"
[dog]*	matches matches a string of zero or more characters from the set {"d" an "o" or a "g"}
(dog|cat)	matches the string "dog" or the string "cat"
dog.*cat	matches the string "dog" followed by the string "cat" somewhere later in the string
x(dog|cat)x	matches the string "dog" or the string "cat" between two "x"s
xx*	matches a string of one or more "x"s
x+	matches a string of one or more "x"s
x(dog|cat)?x	matches two "x"s with optionally the string "dog" or the string "cat" between the "x"'s
[aeiou]	matches a single vowel
[A-Z]+	matches a string of one or more uppercase characters
[az-]+	matches a string of one characters from the set or three characters "a", "z", "-"
[^a-z]+	matches a string of one or more characters that are not lowercaase letters
"[a-z]"	in flex matches exactly the five character string "[a-z]"
[a-zA-Z][a-zA-Z0-9]*	matches a letter optionally followed by letters or digits
[1-9][0-9]*|0	matches a positive integer with no leading zero except when the number is zero
[+-]?[0-9]+	matches an integer with optional sign (note that leading zeroes are allowed
([0-9].)*	matches an even number of characters where every odd numbered character is a digit
[+-]?[1-9][0-9]*|0	matches an integer with no leading zero except when the number is zero. The number may have an optional sign
[\^\+\-\:*\]]	matches one of the 6 characters: "^", "+", "-", ":", "*", "]"