



Regular Expressions

Lecture 10

Michael J Wise

Regular Expressions

- A regular expression is any string that you want to match with another string. However, regular expressions can also contain “wild-cards”, allowing multiple target strings to match.
- Unfortunately, UNIX has two different formats for regular expressions
 - *Shell pattern matching, .a.k.a. globbing, i.e. the regular expressions you have already seen in connection with filenames and the case statement.*
 - *A number of UNIX utilities, e.g. `grep`, use a different, expanded format for regular expressions derived from `ed` (predecessor to `vi`).*

Wild-card Patterns

ed	shell	Description
.	?	Single character
[]	[]	Single character from set or range(s)
[^]	[^]	Single character NOT from this set/range(s)
*		Zero or more occurrences of preceding letter
.*	*	Zero or more occurrences of any letter
^		Start of a matching string
\$		End of a matching string
\	\	Take special meaning away from next letter
\(\)		Capture match for later reuse

The last of these is a more advanced facility, but one that I use a lot (more in a bit)

Examples of Ed-Style Patterns

- `malloc * (` # No space before (
– *Note one or more spaces*
 - `.` *
– *Any string (include empty string)*
 - `\.`
– *Dot (as such)*
 - `[a-zA-Z] [a-zA-Z] *`
– *Alphabetic string*
 - `[0-9] [0-9] * \. [0-9] [0-9] *`
– *Floating point number*
 - `^$`
– *Empty line*
-

Capturing Matches

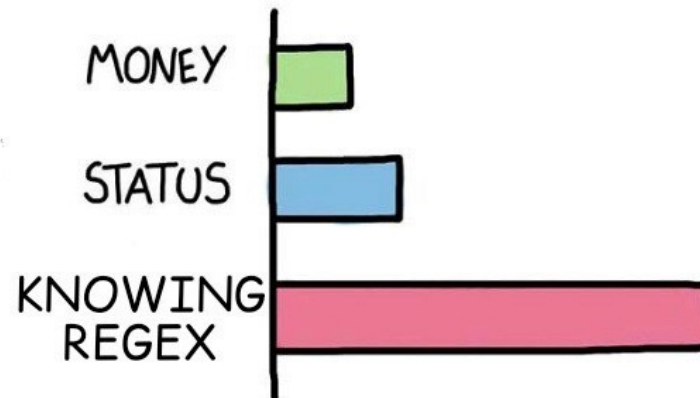
- `sed` and `grep` (but not `awk`) provide a way of recording the match you have made and then recalling it for use in later comparisons (or substitutions)
- `\(\)` around a regular expression records the string that matched that regular expression. You can record up to 9 regular expression matches in a single operation.
- `\<N>` is used to refer to one of the recorded matches, that is `\1` refers to the first match recorded, `\2` for the second, and so on.

Capturing Matches

- Example: `\ ([a-z]) \ ([a-z]) \1\2` matches a string where the first letter is repeated as the third letter and the second repeated as the fourth, e.g. `abab`, but not `abba`
 - *Does `xxxx` match?*

Regex Power

WHAT GIVES PEOPLE
FEELINGS OF POWER



Neetish Raj, <https://neetishop.medium.com/best-learning-path-to-master-regex-for-javascript-developers-d928960a9d14>

grep

`grep [<options>] <regular-expression> <file> ...`

- As we saw before, you can search with an ordinary string, but in reality it's a regular expression
- Useful options:
 - `-i` Make comparisons case insensitive (“i” matches “I”)
 - `-n` Prepend the numbers of the matching lines
 - `-v` Invert the match so only non-matching lines are reported

Examples

What lines do these patterns match:

- `grep '^\. [VABL] [LIE] $' file`
- `grep -v 'warning:' errs | grep -v 'In function'`
- `find . -name Makefile -exec grep awk ' {} ' \; -
print`

Demo

- Let's use `grep` to extract lines from `Alice_in_Wonderland.txt`
- Extract the lines that mention both Alice and cat
- Extract the lines that mention oyster or mystery
- In the spirit of Wordle, I want to get a list of 5 letter words from the Alice text.
 - *How about 5 letter palindromes (i.e. words such as radar that read the same right to left, as left to right).*