COMP3001 2012/13 Perl Coursework 1 Report

Patrick Naish
pn3g10@soton.ac.uk

October 25, 2012

1 Approach

For locating datasheets, it was clear that recursion is the best option within perl itself to traverse folder structures. The full path at the current directory is mapped onto each file/directory within it, except for the special "." and ".." directories. For brevity, regex matching and substitution is used to manipulate paths. The subroutine pushes pdfs and docs into arrays, and recurses on folders.

For matching the items present in both arrays, a hashmap is built up for the contents of both arrays. Grepping through for keys in the map with a value greater than 1 shows which keys are present in both arrays, and therefore is a dual-format datasheet.

For parsing logfiles, a hashmap for the array of dual-format datasheets is first built up. Regex matching is then used to identify GET requests to docs and pdfs. The unnecessary (non-path) text is split out with another regex, and the EOL character is chomped. The line is then appended with a BR tag, to match those stored. If the hashmap has a defined value for the current line, then it is a dual-format datasheet and can be counted. This is not hugely optimised, but does not take unreasonably long to execute.

2 Difficulty

The CGI section was reasonably straightforward, due to the provided examples and online references. Linuxproj temporarily could not locate CGI.pm, causing some stress, but eventually it began working again. I also forgot to set execute permissions on the script originally. Recursing through folders was also fairly intuitive, being a problem we have encountered before.

I found it hard to think of reasonable ways to match array values, not being hugely familiar with the use of hashmaps. After reading in more detail about them, however, this was also not too difficult. The main issue I encountered was with passing array parameters to subroutines, which still feels counterintuitive to me.