Short bash and perl summary

Axel Angel

1 Shell Variables

- $* Expands to the positional parameters, starting from one. When the expansion occurs within double quotes, it expands to a single word with the value of each parameter separated by the first character of the IFS special variable.
- $@ Expands to the positional parameters, starting from one. When the expansion occurs within double quotes, each parameter expands to a separate word.
- $# Expands to the number of positional parameters in decimal.
- $? Expands to the exit status of the most recently executed foreground pipeline.
- $- A hyphen expands to the current option flags as specified upon invocation, by the set built-in command, or those set by the shell itself (such as the -i).
- $$ Expands to the process ID of the shell.
- $! Expands to the process ID of the most recently executed background (asynchronous) command.
- $0 Expands to the name of the shell or shell script.
- $_ The underscore variable is set at shell startup and contains the absolute file name of the shell or script being executed as passed in the argument list. Subsequently, it expands to the last argument to the previous command, after expansion. It is also set to the full pathname of each command executed and placed in the environment exported to that command. When checking mail, this parameter holds the name of the mail file.
- $& This variable holds the string that was matched by the last successful pattern match.
- $‘ This variable holds the string that preceded whatever was matched by the last successful pattern match.
- $’ This variable holds the string that followed whatever was matched by the last successful pattern match.

2 Perl Variables

- $_ The default parameter for a lot of functions.
- $. Holds the current record or line number of the file handle that was last read. It is read-only and will be reset to 0 when the file handle is closed.
- $/ Holds the input record separator. The record separator is usually the newline character. However, if $/ is undefined, the Perl reads the entire file as one input file.
- $, The output separator for the print() function. Normally, this variable is an empty string. However, setting $, to a newline might be useful if you need to print each element in the parameter list on a separate line.
- $\ Added as an invisible last element to the parameters passed to the print() function. Normally, an empty string, but if you want to add a newline or some other suffix to everything that is printed, you can assign the suffix to $\.
- $$ This UNIX-based variable holds the process number of the process running the Perl interpreter.
- $? Holds the status of the last pipe close, back-quote string, or system() function. You can find more information about the $? variable in Chapter 13, "Handling Exceptions and Signals."
- $! When used in a numeric context, holds the current value of errno. If used in a string context, will hold the error string associated with errno. For more information about errno, see Chapter 13, "Handling Exceptions."
- %ENV This hash variable contains entries for your current environment variables. Changing or adding an entry affects only the current process or a child process, never the parent process. See the section "Example: Using the
- $ARGV Holds the name of the current file being read when using the diamond operator (¡¿).
- @ARGV This array variable holds a list of the command line arguments. You can use $#ARGV to determine the number of arguments minus one.
- $0 This variable holds the name of the file containing the Perl script being executed.

3 Perl Run Options

- -s : permet un traitement rudimentaire des options passes au script
- -d : debug mode
- -e : pour donner directement une ligne de commande
- -n : ajoute le code suivant autour de votre programme : LINE: while (¡¿)
- -p : similaire -n, affiche en plus la ligne lue chaque itration.
- -a (avec -n ou -p) : ajoute @F = split( );
- -Fpattern (avec -a) utilise lexpression rgulire pattern plutt que le blanc pour faire le split du -a

- -l (L minuscule) : ajoute un chomp; si utilis avec -n ou -p et fait $\=$/
- -i ou -iextension : les fichiers lus par ¡¿ doivent tre dit sur
- place , cest--dire sont modifis.
- -w : dafficher des messages dalerte du compilateur
- -T : force lactivation des vrifications de marquage (scurit)

4 Perl Functions

- chomp VARIABLE
- chop VARIABLE
- close FILEHANDLE
- delete EXPR
- die LIST
- glob EXPR
- index STR,SUBSTR,POSITION
- join EXPR,LIST
- keys HASH
- map BLOCK LIST
- open FILEHANDLE,EXPR
- pop ARRAY
- push ARRAY,LIST
- readline EXPR
- reverse LIST
- shift ARRAY
- sort SUBNAME LIST
- splice ARRAY,OFFSET,LENGTH,LIST
- split /PATTERN/,EXPR,LIMIT
- substr EXPR,OFFSET,LENGTH,REPLACEMENT
- values HASH
- warn LIST

5 Perl Regex

- match m//
- replace s///

5.1 Defined Characters

- \d : [0-9]
- \s : [\t\r\n\f] (un blanc)
- \w : [0-9a-zA-Z_] (alphanumrique ou _)
- \D : ngation de \d : [ô-9]
- \S : ngation de \s
- \W : ngation de \w
- \b : limite de mot (par exemple entre \w et \W)

5.2 Options

- i : ignore case (majuscules/minuscules)
- s : . (point) correspond nimporte quel caractre y compris \n
- m : considre la chane traite comme une srie de plusieurs lignes. êt $ correspondent nimporte quel dbut/fin de ligne (i.e. avec \n). Si lon veut indiquer vraiment le dbut ou la fin de la chane, on utilisera dans ce cas respectivement \A et \Z [\n tolr en fin de chane] ou \z [strict]
- x : permet de mettre des commentaires et des espaces dans les expressions rgulires afin de les rendre plus lisibles.
- o : compile les expressions rgulires contenant des variables (dont le contenu ne doit pas changer au cours des futures valuations)
- g (comme “global”) : cherche/remplace toutes les occurrences et retourne:
  - dans un contexte de tableau, le tableau de toutes les occurrences ;
  - dans un contexte scalaire, la prochaine occurrence depuis le dernier appel de cet expression rgulire (utilisation multiple). La fonction pos (applique la chane traite) retourne la position de fin de loccurrence courante.
- e (comme eval) : option supplmentaire loprateur de substitution s/// . Cette option value (au sens perl) la chane de remplacement et utilise cette valeur comme remplacement.

5.3 Special characters

```
| ( ) [ ] { } \ / $ + * ? .
```

5.4 Others

- $x = /machin/; # recherche machin dans $_ et met le rsultat (vrai/faux) dans $x
- $x =∼ /machin/; # recherche machin dans $x et ne garde pas le rsultat mais retourne:
  - dans un contexte scalaire : vrai ou faux en fonction que lexpression correspond ou non
  - dans un contexte de tableau : les contenus des diffrents groupes, i.e. ($1, $2, $3, etc.)