Seneca | SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY

Unix Scripting

Lecturer: Shahdad Shariatmadari

July 2020

Agenda

- Parsing Program Arguments, **Option Parsing**
  - getops

Parsing Program Arguments

- A common task in shell scripting is to parse command line arguments to your script.
- Bash provides the **getopts** built-in function to do just that.
  - **`getopts`** *`optstring name [arg ...]`*
- The **getopts** function takes three parameters.
  - The first is a **specification (optsting)** of which options are valid, listed as a sequence of letters
    - if the script recognizes **-a**, **-f** and **-s**, *optstring* is ***`afs`***
  - The *name* on the **getopts** command line is the name of a shell variable. Each time you invoke **getopts**, it obtains the next option from the *positional parameters* and places the option letter in the shell variable *name*.
  - The third argument to getopts is the **list of arguments and options** to be processed.

Activity: Run this and explain how it works

```
while getopts ":ht" opt; do
   case ${opt} in
     h ) # process option h
        ;;
     t ) # process option t
        ;;
     \? ) echo "Usage: cmd [-h] [-t]"
        ;;
   esac
done
```

the string 'ht' signifies that the options -h and -t are valid.

opt will hold the value of the current option that has been parsed by getopts.

3rd argument of getopts

- When not provided, this defaults to the arguments and options provided to the application ($@). You can provide this third argument to use getopts to parse any list of arguments and options you provide.

Parsing options with arguments

- When **getopts** obtains an option from the script command line, it stores the index of the next argument to be processed in the shell variable **OPTIND**.
- When an option letter has an associated argument (indicated with a : in *optstring*), **getopts** stores the argument as a string in the shell variable **OPTARG**.
  - If an option doesn't take an argument, or **getopts** expects an argument but doesn't find one, **getopts** unsets OPTARG.

OPTIND and OPTARG

- **ENVIRONMENT VARIABLES**
  - ***OPTARG*** stores the value of the option argument found by **getopts**.
  - ***OPTIND*** contains the index of the next argument to be processed.

Activity: Run this and explain how it works

```
while getopts ":t:" opt; do
  case ${opt} in
    t )
      target=$OPTARG
      ;;
    \? )
      echo "Invalid option: $OPTARG" 1>&2
      ;;
    : )
      echo "Invalid option: $OPTARG requires an argument"
1>&2
      ;;
  esac
done
shift $((OPTIND -1))
```

Shifting processed options

- The variable **OPTIND** holds the number of options parsed by the last call to getopts.
- It is common practice to call the ***shift command*** at the end of your processing loop to remove options that have already been handled from **$@**.

Notes

- The special option of two dashes ("**--**") is interpreted by **getopts** as the end of options.
- By default, **getopts** will report a verbose error if it finds an unknown option or a misplaced argument. It also sets the value of *optname* to a question mark ("**?**"). It does not assign a value to **$OPTARG**.
  - If the option is valid but an expected argument is not found, *optname* is set to "**?**", **$OPTARG** is unset, and a verbose error message is printed.
- If you put a colon at the beginning of the *optstring*, getopts runs in "silent error checking mode." It will not report any verbose errors about options or arguments, and you need to perform error checking in your script.

getopt vs getopts

- There is also the external utility **getopt** , which parses long-form arguments, like "--filename" instead of the briefer "-f" form
- **getopt**’s traditional versions can’t handle empty argument strings, or arguments with embedded whitespace

Activity

- Download the script from the BB->CourseDocument->Week10, named **greeting**
- Run the scrip as follow and explain what happen at each step:
  - ./greeting
  - ./greeting -n UNIX
  - ./greeting -t 5
  - ./greeting -t 4 -n Seneca
  - ./greeting -t 0