Misc Gnu Documentation

8 September 2008

Eric Hamilton

Contents

1 Using the GETOPT function from The GNU C Library

Here are the details about how to call the GETOPT function. To use this facility, your program must include the header file `unistd.h`.

1.1 Variable: int `opterr`

If the value of this variable is nonzero, then GETOPT prints an error message to the standard error stream if it encounters an unknown option character or an option with a missing required argument. This is the default behavior. If you set this variable to zero, GETOPT does not print any messages, but it still returns the character `?` to indicate an error.

1.2 Variable: int `optopt`

When GETOPT encounters an unknown option character or an option with a missing required argument, it stores that option character in this variable. You can use this for providing your own diagnostic messages.

1.3 Variable: int `optind`

This variable is set by GETOPT to the index of the next element of the `argv` array to be processed. Once GETOPT has found all of the option arguments, you can use this variable to determine where the remaining non-option arguments begin. The initial value of this variable is `1`.

[1] `$Header: d:/Binder0/GetOpt/RCS/GnuDoc.tex,v 1.1 2008-09-05 07:41:37-07 Hamilton Exp Hamilton $`
[2] `$Header: d:/Binder0/GetOpt/RCS/GetOpt.inc,v 1.2 2008-09-08 10:45:38-07 Hamilton Exp Hamilton $`

1.4 Variable: char * optarg

This variable is set by GETOPT to point at the value of the option argument, for those options that accept arguments.

1.5 Function: int GETOPT (int argc, char **argv, const char *options)

The GETOPT function gets the next option argument from the argument list specified by the `argv` and `argc` arguments. Normally these values come directly from the arguments received by MAIN.

The options argument is a string that specifies the option characters that are valid for this program. An option character in this string can be followed by a colon (‘`:`’) to indicate that it takes a required argument. If an option character is followed by two colons (‘`::`’), its argument is optional; this is a GNU extension.

GETOPT has three ways to deal with options that follow non-options `argv` elements. The special argument ‘`--`’ forces in all cases the end of option scanning.

- The default is to permute the contents of `argv` while scanning it so that eventually all the non-options are at the end. This allows options to be given in any order, even with programs that were not written to expect this.
- If the options argument string begins with a hyphen (‘`-`’), this is treated specially. It permits arguments that are not options to be returned as if they were associated with option character ‘`\1`’.
- POSIX demands the following behavior: The first non-option stops option processing. This mode is selected by either setting the environment variable `POSIXLY_CORRECT` or beginning the options argument string with a plus sign (‘`+`’).

The GETOPT function returns the option character for the next command line option. When no more option arguments are available, it returns `-1`. There may still be more non-option arguments; you must compare the external variable `optind` against the `argc` parameter to check this.

If the option has an argument, GETOPT returns the argument by storing it in the variable `optarg`. You don’t ordinarily need to copy the `optarg` string, since it is a pointer into the original `argv` array, not into a static area that might be overwritten.

If GETOPT finds an option character in `argv` that was not included in options, or a missing option argument, it returns ‘`?`’ and sets the external variable `optopt` to the actual option character. If the first character of options is a colon (‘`:`’), then GETOPT returns ‘`:`’ instead of ‘?’ to indicate a missing option argument. In addition, if the external variable `opterr` is nonzero (which is the default), GETOPT prints an error message.

1.6 Example of Parsing Arguments with GETOPT, The GNU C Library

Here is an example showing how GETOPT is typically used. The key points to notice are:

- Normally, GETOPT is called in a loop. When GETOPT returns `-1`, indicating no more options are present, the loop terminates.
- A switch statement is used to dispatch on the return value from GETOPT. In typical use, each case just sets a variable that is used later in the program.
- A second loop is used to process the remaining non-option arguments.

```
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int
main (int argc, char **argv)
{
  int aflag = 0;
  int bflag = 0;
  char *cvalue = NULL;
  int index;
  int c;

  opterr = 0;

  while ((c = getopt (argc, argv, "abc:")) != -1)
    switch (c)
      {
      case 'a':
        aflag = 1;
        break;
      case 'b':
        bflag = 1;
        break;
      case 'c':
        cvalue = optarg;
        break;
      case '?':
        if (optopt == 'c')
          fprintf (stderr, "Option -%c requires an argument.\n", optopt);
        else if (isprint (optopt))
          fprintf (stderr, "Unknown option `-%c'.\n", optopt);
        else
          fprintf (stderr,
                   "Unknown option character `\\x%x'.\n",
                   optopt);
        return 1;
```

```
      default:
        abort ();
      }

  printf ("aflag = %d, bflag = %d, cvalue = %s\n",
          aflag, bflag, cvalue);

  for (index = optind; index < argc; index++)
    printf ("Non-option argument %s\n", argv[index]);
  return 0;
}
```

Here are some examples showing what this program prints with different combinations of arguments:

```
% testopt
aflag = 0, bflag = 0, cvalue = (null)

% testopt -a -b
aflag = 1, bflag = 1, cvalue = (null)

% testopt -ab
aflag = 1, bflag = 1, cvalue = (null)

% testopt -c foo
aflag = 0, bflag = 0, cvalue = foo

% testopt -cfoo
aflag = 0, bflag = 0, cvalue = foo

% testopt arg1
aflag = 0, bflag = 0, cvalue = (null)
Non-option argument arg1

% testopt -a arg1
aflag = 1, bflag = 0, cvalue = (null)
Non-option argument arg1

% testopt -c foo arg1
aflag = 0, bflag = 0, cvalue = foo
Non-option argument arg1

% testopt -a -- -b
aflag = 1, bflag = 0, cvalue = (null)
Non-option argument -b

% testopt -a -
aflag = 1, bflag = 0, cvalue = (null)
Non-option argument -
```

Index