Compilando con el GCC

Sebastián Santisi

1. Introducción

Este pequeño instructivo es solamente a título informativo como para poder empezar a compilar con el GCC. Sólo abarcará lo básico; se recomienda leer la (abundante) documentación que acompaña a la aplicación para conocer los demás comandos y opciones que pueden usarse.

En primer lugar se presupone que el GCC está instalado.

2. Compilando

El programa que nos servirá para compilar se llama `gcc`. Supongamos que queremos compilar el código fuente `prueba.c` y que el mismo se encuentra en la carpeta `/home/user/7502/`. Lo compilaremos de la siguiente manera (es importante recordar que el GCC una aplicación desarrollada ser ejecutada en una terminal y, como todas las herramientas del universo Unix, distingue entre minúsculas y mayúsculas en los argumentos):

UNIX

```
user@loc 7502$ gcc -o prueba prueba.c -Wall -pedantic -std=c99
```

Esto compilará nuestro código fuente `prueba.c` creando el ejecutable `prueba` (el cual podremos ejecutar en la terminal con `./prueba`). Ahora bien, ¿qué significa cada parámetro?...

- `-o`: (output) Indica que el siguiente argumento será el nombre con el cual queremos que se compile nuestro programa.
- `-Wall`: (warning all) Indica que queremos que el compilador nos muestre todas las advertencias que considere necesarias respecto de nuestro fuente.
- `-std=c99`: Indica (si estamos compilando un fuente C) que queremos que compile según el standard ISO-C99 (es lo que requerimos en este curso).
- `-pedantic`: Indica que queremos que el compilador se ponga "pedante" con los mensajes de error y advertencias.

Estos argumentos que se han presentado alcanzan como para poder comenzar a trabajar con el compilador. Se presenta una lista de otros argumentos útiles (no salva de leer la documentación que acompaña al compilador ;)) que acepta el programa:

- `-x c`: Obliga al compilador a compilar nuestro fuente como si fuera en C. El GCC compila muchos otros lenguajes y se vale de la extensión para saber en cuál compilar .c lo reconoce como C.
- `-O1`: Optimiza la compilación para espacio y velocidad.
- `-O2`: Optimiza más, agregando más *flags*.
- `-O3`: Optimiza todavía más.
- `-Os`: Optimiza la compilación para que el ejecutable ocupe menos espacio (selecciona automáticamente los *flags* entre `-O[123]`).
- `-Ofast`: (excluyente con el anterior) Optimiza la compilación para velocidad. No se preocupa mucho por los estándares, mientras la velocidad aumente. (Como `-O3` y más *flags*).
- `-E`: Solamente preprocesa el código; no compila.
- `-S`: Solamente genera el código assembler; no compila.
- `-c`: Solamente genera el código objeto; no enlaza.
- `-g`: Compila a nuestro programa guardando información para *debugging*.
- `-lm`: Enlaza la biblioteca matemática (esto no se hace por omisión para economizar espacio).
- `--help`: Esta opción nos muestra un pequeño panorama de los argumentos que toma el programa.

2003-09-09
Sebastián Santisi
santisis@7502.com.ar

Última revisión: 2019-03-26 (Sebastián Santisi)
Revisión: 2012-01-09 (Patricio Moreno)
Revisión: 2005-03-24 (Sebastián Santisi)