Decreto ejemplo Decreto N° 0
Decreto ejemplo
Fecha de sanción: 2022-07-13
Fecha de publicacion: 2022-07-28
Resumen: Resumen

Cap´ıtulo 1 Introducci´on al lenguaje de programaci´on C++ 1.1. Lenguaje C++ C++ es un lenguaje de programaci´on, creado a mediados de 1980 por Bjarne Strousstrup, como extensi´on del lenguaje C. Este lenguaje abarca tres paradigmas de la programaci´on: 1. Programaci´on Estructurada 2. Programaci´on Gen´erica 3. Programaci´on Orientada a Objetos En la actualidad, C++ es un lenguaje vers´atil, potente y general. Su ´exito entre los programadores le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones, ya sea en Windows o GNU Linux, que es el sistema operativo en el cual basaremos este tutorial. 1.2. C++ en un entorno Linux Comenzaremos diciendo que los programas se pueden escribir en cualquier editor de textos de GNU, entre ellos se encuentran emacs, vim, kate, gedit, nan, guardando dichos archivos con extensi´on .cpp, los cuales ser´an compilados en GNU/linux utilizando el compilador GNU de C++, llamado gcc que puede compilar C, C++, y que adem´as se apega al estandar ANSI, permitiendo la portabilidad de estos c´odigos. Dicho compilador se invoca con el comando gcc. 3 Para compilar ponemos la siguiente l´ınea en una terminal previamente ubicada en el directorio que contiene nuestro archivo: g++ programa.cpp -o programa.out -o indica el nombre del archivo de salida el cual ser´a el ejecutable de nuestro proyecto. Luego para ejecutar, escribimos sobre la l´ınea de comandos: ./programa.out y entonces podremos ejecutar nuestro programa. Cuando creamos peque˜nos programas la compilaci´on de ´estos es muy f´acil, pero cuando se trabaja con proyectos grandes, con varios archivos fuente la compilaci´on resulta m´as dificil, por lo que Linux proporciona la utiler´ıa make de GNU, el cual busca un archivo make donde encontrar´a toda la informaci´on que necesita para crear el ejecutable, si encuentra el archivo busca la palabra makefile o Makefile, que son nombres predeterminados. Los archivos make contienen informaci´on acerca de la compilaci´on y enlace del programa, con una sintaxis muy especifica. Un makefile se define como una lista de normas y dependencias con sus correspondientes comandos para crear objetivos, utilizando dichas normas y dependencias. Un archivo Makefile es un archivo de texto en el cual se distinguen cuatro tipos b´asicos de declaraciones: 1. Comentarios: Al igual que en los programas, contribuyen a un mejor entendimiento de las reglas definidas en el archivo. Los comentarios se inician con el caracter #, y se ignora todo lo que contin´ue despu´es de ella, hasta el final de l´ınea. 2. Variables: Se definen utilizando el siguiente formato: nombre = dato De esta forma, se simplifica el uso de los archivos Makefile. Para obtener el valor se emplea la variable encerrada entre par´entesis y con el caracter $ al inicio, en este caso todas las instancias de $(nombre) ser´an reemplazadas por datos. Por ejemplo, la siguiente definici´on SRC = main.c origina la siguiente l´ınea: gcc $ SRC y ser´a interpretada como: gcc main.c Sin embargo, pueden contener m´as de un elemento. Por ejemplo: objects = programa 1.o programa 2.o programa 3.o \ programa 4.o programa 5.o 4 programa: $(objects) gcc -o programa $(objects) Hay que notar que make hace distinci´on entre may´usculas y min´usculas. 3. Reglas Expl´ıcitas: Estas le indican a make qu´e archivos dependen de otros, as´ı como los comandos requeridos para compilar un archivo en particular. Su formato es: archivoDestino: archivosOrigen Esta regla indica que, para crear archivoDestino, make debe ejecutar comandos sobre los archivos ar