# Acerca de

Esta práctica ofrece una librería con dos funciones que se podrán utilizar desde Windows y desde Linux.

La primera función se llama **GetProcessList**, la cual retorna una lista con los *process id* de los procesos que están corriendo actualmente en la máquina. Para invocar esta función, se debe enviar una referencia de una variable entera. En esta variable, al finalizar la función modificará este valor y allí se podrá encontrar el total de procesos corriendo.

La otra función, llamada **CpuUsage**, retorna el porcentaje de uso de la CPU desde la última vez que se utilizó esta función. Es decir, cuando se llama por primera vez, retorna el uso de la CPU desde que se inició la máquina, pero la segunda vez retorna el uso de la CPU desde que se llamó la primera vez la función. Esta función no se debe llamar muy seguido, se recomienda mínimo cada segundo para poder apreciar un cambio significativo en el uso de la CPU.

A continuación se muestra un programa de prueba en linux que utiliza ambas funciones de la librería.

#include <cstdio>

#include "linux/SO.h"

int main(){

int\* processes = NULL;

int numP = 0;

processes = SO::GetProcessList(&numP);

printf("Procesos:\n");

for(int i = 0; i < numP; ++i)

printf("Proceso %d: %d\n",i+1,processes[i]);

printf("%f\n",SO::CpuUsage());

return 0;

}

Compilación en Linux

Para compilar la librería en Linux, debemos correr el makefile. Esto sólo compilará la librería pero no copiará el .so a las carpetas necesarias.

Para compilar e instalar la librería, debemos correr el siguiente comando como sudo. Es necesario correrlo con sudo pues se copiarán archivos en **/usr/lib** y en **/usr/include** y se necesitan permisos para esto.

sudo make install

Esto creará el archivo **libso-eafit.so** el cual se copiará en la carpeta **/usr/lib**. Además de esto, se copiará el archivo **SO.h** en la carpeta **/usr/include/**

Compilación y ejecución del programa de prueba

Una vez corrido el makefile con los permisos necesarios, es fácil compilar el programa de ejemplo, solo es necesario correr el siguiente comando

g++ -lso-eafit practica.cpp -o practica && ./practica

Esto compilará y correrá el programa compilado.

# Compilación en Windows

Para compilar la librería en Windows hay que asegurarse que el símbolo **BUILDING\_DLL** este definido, que se este compilando como un *dll* (algo equivalente a la opción -shared en minGW), y que se deben *linkear* las librerías *psapi* y *Kernel32* (en minGW es *-lpsapi -lKernel32*).

Si se va compilar usando el IDE dev-c++ (fue como lo hicimos nosotros), en las opciones del proyecto bajo la pestaña general se selecciona type como win32 dll. Luego en las pestaña parameters bajo Compiler y C++ compiler se añade la siguiente linea

-DBUILDING\_DLL=1

Bajo Linker se añade esto

--no-export-all-symbols --add-stdcall-alias -lpsapi -lKernel32

Por último bajo la pestaña *build options* hay que asegurarse que el nombre del archivo de salida sea ***so-eafit.dll***

Se generan 2 archivos, uno es **libso-eafit.a** y el otro es **so-eafit.dll**. El primero debe ir a una carpeta donde el compilador busca las librerías (para Dev-c++ esta es la carpeta **/ruta-a-dev-cpp/lib**), y el *.dll* debe ir en la misma carpeta de los ejecutables que usen la librería en *C:\Windows\System32*

Compilación y ejecución del programa de prueba

Ya generado el dll se puede compilar el programa de prueba desde el mismo Dev-c++ como un archivo solo (no hay necesidad de hacer un proyecto), lo único es que hay que ir a Tools > Compiler Options y añadir la linea –lso-eafit en las opciones del linker. Además es necesario que el .h de la librería y el dll estén en la misma carpeta o en las carpetas del sistema (o cualquiera que este definida en el IDE)