Basic knowledge Test Project

Visual C++ 및 리눅스 컴파일/빌드

- CPU, Memory, HDD 크기 등

- 동작중인 프로세스 리스트 보기

- 특정 폴더 파일 리스트 보기

- 네트워크 주소 확인

# Window Visual C++ version

# Sisinfo.cpp

#include <windows.h>

#include <stdio.h>

#include <tchar.h>

#include <iostream>

void hardware\_info();

// Getting the Hardware info(CPU, Memory, HDD)

void hardware\_info()

{

SYSTEM\_INFO siSysInfo;

GetSystemInfo(&siSysInfo);

printf("\tHardware information \n");

printf("====================================\n");

printf("Number of processors: %u\n", siSysInfo.dwNumberOfProcessors);

printf("Page size: %u\n", siSysInfo.dwPageSize);

printf("Processor type: %u\n", siSysInfo.dwProcessorType);

printf("Minimum application address: %lx\n", siSysInfo.lpMinimumApplicationAddress);

printf("Maximum application address: %lx\n", siSysInfo.lpMaximumApplicationAddress);

printf("Active processor mask: %u\n", siSysInfo.dwActiveProcessorMask);

MEMORYSTATUSEX memInfo;

GlobalMemoryStatusEx(&memInfo);

memInfo.dwLength = sizeof(MEMORYSTATUSEX);

DWORDLONG totalPhysMem = memInfo.ullTotalPhys; // 전체 메모리

DWORDLONG availPhysMem = memInfo.ullAvailPhys; // 남은 메모리

printf("Total Mem: %I64u KB, Available Mem: %I64u KB\n", totalPhysMem / 1024, availPhysMem / 1024); // KB 로 출력

BOOL fResult;

unsigned \_\_int64 i64FreeBytesToCaller, i64TotalBytes, i64FreeBytes;

fResult = GetDiskFreeSpaceEx("C:",

(PULARGE\_INTEGER)&i64FreeBytesToCaller,

(PULARGE\_INTEGER)&i64TotalBytes,

(PULARGE\_INTEGER)&i64FreeBytes);

if (fResult)

{

printf("Available space to caller = %I64u MB\n",

i64FreeBytesToCaller / (1024 \* 1024));

printf("Total space = %I64u MB\n",

i64TotalBytes / (1024 \* 1024));

printf("Free space on drive = %I64u MB\n",

i64FreeBytes / (1024 \* 1024));

} }

int main()

{

hardware\_info();

printf("====================================\n");

}

// \*Source

// https://docs.microsoft.com/ko-kr/windows/win32/sysinfo/getting-hardware-information

// https://stackoverflow.com/questions/11917946/how-do-i-get-available-disk-space-from-windows-using-c

// https://ospace.tistory.com/514

# Output

Hardware information

====================================

Number of processors: 16

Page size: 4096

Processor type: 8664

Minimum application address: 10000

Maximum application address: fffeffff

Active processor mask: 65535

Total Mem: 4126 KB, Available Mem: 14405 KB

Available space to caller = 273136 MB

Total space = 453215 MB

Free space on drive = 273136 MB

====================================

# Process2.cpp

gcc -o process2.exe process2.cpp -lPSAPI

#include <windows.h>

#include <stdio.h>

#include <tchar.h>

#include <psapi.h>

// To ensure correct resolution of symbols, add Psapi.lib to TARGETLIBS

// and compile with -DPSAPI\_VERSION=1

void PrintProcessNameAndID( DWORD processID )

{

TCHAR szProcessName[MAX\_PATH] = TEXT("<unknown>");

// Get a handle to the process.

HANDLE hProcess = OpenProcess( PROCESS\_QUERY\_INFORMATION |

PROCESS\_VM\_READ,

FALSE, processID );

// Get the process name.

if (NULL != hProcess )

{

HMODULE hMod;

DWORD cbNeeded;

if ( EnumProcessModules( hProcess, &hMod, sizeof(hMod),

&cbNeeded) )

{

GetModuleBaseName( hProcess, hMod, szProcessName,

sizeof(szProcessName)/sizeof(TCHAR) );

}

}

// Print the process name and identifier.

\_tprintf( TEXT("%s (PID: %u)\n"), szProcessName, processID );

// Release the handle to the process.

CloseHandle( hProcess ); }

int main(void)

{

// Get the list of process identifiers.

DWORD aProcesses[1024], cbNeeded, cProcesses;

unsigned int i;

if ( !EnumProcesses( aProcesses, sizeof(aProcesses), &cbNeeded ) )

{

return 1;

}

// Calculate how many process identifiers were returned.

cProcesses = cbNeeded / sizeof(DWORD);

// Print the name and process identifier for each process.

for ( i = 0; i < cProcesses; i++ )

{

if( aProcesses[i] != 0 )

{

PrintProcessNameAndID( aProcesses[i] );

}

}

return 0;

}

//Source

//https://stackoverflow.com/questions/11564148/how-to-get-the-starting-base-address-of-a-process-in-c

# Output

Current Process information

====================================

<unknown> (PID: 4)

<unknown> (PID: 172)

<unknown> (PID: 600)

<unknown> (PID: 900)

<unknown> (PID: 752)

<unknown> (PID: 1012)

<unknown> (PID: 1000)

<unknown> (PID: 1144)

<unknown> (PID: 1172)

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

PerfWatson2.exe (PID: 19712)

Microsoft.ServiceHub.Controller.exe (PID: 9288)

ServiceHub.VSDetouredHost.exe (PID: 18704)

ServiceHub.IdentityHost.exe (PID: 20276)

ServiceHub.SettingsHost.exe (PID: 5712)

ServiceHub.Host.CLR.x64.exe (PID: 11504)

ServiceHub.Host.CLR.x86.exe (PID: 1220)

VCPkgSrv.exe (PID: 11292)

cpptools-srv.exe (PID: 2800)

chrome.exe (PID: 20440)

chrome.exe (PID: 13416)

chrome.exe (PID: 13684)

VCPkgSrv.exe (PID: 4032)

<unknown> (PID: 8524)

cpptools-srv.exe (PID: 11448)

MSBuild.exe (PID: 10760)

conhost.exe (PID: 10540)

cpptools-srv.exe (PID: 13984)

cpptools-srv.exe (PID: 11100)

smartscreen.exe (PID: 18840)

WINWORD.EXE (PID: 11940)

<unknown> (PID: 10644)

chrome.exe (PID: 12764)

chrome.exe (PID: 17052)

chrome.exe (PID: 13092)

<unknown> (PID: 20360)

<unknown> (PID: 13872)

<unknown> (PID: 10452)

<unknown> (PID: 16056)

SearchProtocolHost.exe (PID: 1804)

cpptools-srv.exe (PID: 19248)

backgroundTaskHost.exe (PID: 14920)

RuntimeBroker.exe (PID: 10516)

cpptools-srv.exe (PID: 11760)

process2.exe (PID: 20728)

====================================

# Filelist.cpp

#include <iostream>

#include <io.h>

#include <stdio.h>

#include <string>

using namespace std;

int main()

{

string path = "C:\\Users\\teddy\\TechBase\\\*.\*";

struct \_finddata\_t fd;

intptr\_t handle;

if ((handle = \_findfirst(path.c\_str(), &fd)) == -1L)

std::cout << "No file in directory!" << std::endl;

printf("\tFile list information \n");

printf("====================================\n");

do

{

cout << "File: " << fd.name << endl;

} while (\_findnext(handle, &fd) == 0);

\_findclose(handle);

printf("====================================\n");

}

# Output

File list information

====================================

File: .

File: ..

File: .git

File: .gitattributes

File: .gitignore

File: .ipynb\_checkpoints

File: .vs

File: .vscode

File: execute.py

File: filelist.cpp

File: filelist.exe

File: ip.txt

File: linuxs.c

File: network.cpp

File: network.txt

File: process.cpp

File: process2.cpp

File: process2.exe

File: Python\_exe\_ssh.py

File: p\_list.txt

File: README.md

File: sisinfo.cpp

File: visualC.cpp

File: Wincpp.cpp

# Network.cpp

gcc -o network.exe network.cpp -lws2\_32

//네트워크 주소 확인 하기

//#define \_CRT\_SECURE\_NO\_WARNINGS

#include <WinSock2.h>

#include <Windows.h>

#include <stdio.h>

// #pragma warning(disable:4996)

// #pragma comment(lib, "ws2\_32.lib")

FILE\* fp;

void ViewLocalHostIPv4();

int main()

{

WSADATA wsadata;

WSAStartup(MAKEWORD(2, 2), &wsadata);

fp = \_wfopen(L"network.txt", L"w+");

ViewLocalHostIPv4();

WSACleanup();

fclose(fp);

return 0;

}

void ViewLocalHostIPv4()

{

char localhostname[MAX\_PATH];

IN\_ADDR addr = { 0, };

if (gethostname(localhostname, MAX\_PATH) == SOCKET\_ERROR)//호스트 이름 얻어오기

{

return;

}

HOSTENT\* ptr = gethostbyname(localhostname);//호스트 엔트리 얻어오기

printf("Network list information \n");

printf("====================================\n");

while (ptr && ptr->h\_name)

{

if (ptr->h\_addrtype == PF\_INET)//IPv4 주소 타입일 때

{

for (int index = 0; ptr->h\_addr\_list[index]; index++)

{

printf("IP: ");

memcpy(&addr, ptr->h\_addr\_list[index], ptr->h\_length);//메모리 복사

printf("%s\n", inet\_ntoa(addr));

fprintf(fp, "%s\n", inet\_ntoa(addr));

}

}

ptr++;

}

printf("====================================\n");

}

//https://ehpub.co.kr/2-4-%EB%A1%9C%EC%BB%AC-%ED%98%B8%EC%8A%A4%ED%8A%B8-ip-%EC%A3%BC%EC%86%8C-%EC%96%BB%EC%96%B4%EC%98%A4%EA%B8%B0-tcpip-%EC%86%8C%EC%BC%93-%ED%86%B5%EC%8B%A0-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98/

# Output

Network list information

====================================

IP: 192.168.56.1

IP: 172.31.96.1

IP: 10.50.31.220

====================================

# Linux Unix version

#include <stdio.h>

#include <stdlib.h>

#include <sys/sysinfo.h>

#include <sys/utsname.h>

#include <sys/types.h>

#include <dirent.h>

#include <sys/ioctl.h>

#include <net/if.h>

#include <arpa/inet.h>

#include <string.h>

void hardware\_info()

{

struct utsname un;

struct sysinfo sys;

if(sysinfo(&sys) < 0) {

fprintf(stderr, "sysinfo() function error.\n");

exit(1);

}

fprintf(stdout, "===================================== \n");

fprintf(stdout, "\tHardware information \n");

fprintf(stdout, "===================================== \n");

fprintf(stdout,

"업타임: %ld days\n"

"메모리 총계: %ld KB\n"

"사용가능 메모리: %ldKB\n"

"현재 프로세스 갯수: %d\n"

"CPU 갯수: %d\n\n",

((sys.uptime/60)/60)/24, sys.totalram/1024, sys.freeram/1024,

sys.procs, get\_nprocs\_conf());

if(uname(&un) < 0) {

fprintf(stderr, "uname() function error.\n");

exit(1);

}

fprintf(stdout,

"시스템 이름: %s\n"

"노드명: %s\n"

"커널버전: %s\n"

"릴리즈버전: %s\n"

"프로세서타입: %s\n",

un.sysname, un.nodename, un.release, un.version, un.machine); }

void process\_info()

{

fprintf(stdout, "\tProcess information \n");

fprintf(stdout, "===================================== \n");

int r=system("ps");

}

void file\_info()

{

fprintf(stdout, "\tfile list information \n");

fprintf(stdout, "===================================== \n");

DIR \*dir;

struct dirent \*ent;

dir = opendir("./");

if(dir)

{

while (ent = readdir (dir))

printf("%s\n", ent -> d\_name);

}

closedir(dir); }

void network\_info()

{

fprintf(stdout, "\tnetwork information \n");

fprintf(stdout, "===================================== \n");

struct ifreq ifr;

char ipstr[40];

int s = socket(AF\_INET, SOCK\_DGRAM, 0);

strncpy(ifr.ifr\_name, "enp0s3", IFNAMSIZ);

if(ioctl(s, SIOCGIFADDR, &ifr) < 0)

printf("no network");

inet\_ntop(AF\_INET, ifr.ifr\_addr.sa\_data+2, ipstr, sizeof(struct sockaddr));

printf("IP Address : %s\n", ipstr);

}

int main(void)

{

hardware\_info();

fprintf(stdout, "\n===================================== \n");

process\_info();

fprintf(stdout, "\n===================================== \n");

file\_info();

fprintf(stdout, "\n===================================== \n");

network\_info();

# Output

=====================================

Hardware information

=====================================

업타임: 0 days

메모리 총계: 4023440 KB

사용가능 메모리: 911756KB

현재 프로세스 갯수: 491

CPU 갯수: 1

시스템 이름: Linux

노드명: teddy-VirtualBox

커널버전: 5.11.0-40-generic

릴리즈버전: #44~20.04.2-Ubuntu SMP Tue Oct 26 18:07:44 UTC 2021

프로세서타입: x86\_64

=====================================

Process information

=====================================

PID TTY TIME CMD

72874 pts/0 00:00:00 bash

72951 pts/0 00:00:00 linuxs.exe

72952 pts/0 00:00:00 sh

72953 pts/0 00:00:00 ps

=====================================

file list information

=====================================

.gitattributes

.vscode

.git

network.cpp

execute.py

linuxs.c

network.txt

process.cpp

.gitignore

filelist.cpp

.

process2.cpp

sisinfo.cpp

Python\_exe\_ssh.py

linuxs.exe

README.md

ip.txt

visualC.cpp

p\_list.txt

..

Wincpp.cpp

=====================================

network information

=====================================

IP Address : 10.0.2.15