**<10.20>**

#pragma once

#ifndef \_\_QUEUE\_H\_\_

#define \_\_QUEUE\_H\_\_

#define Q\_MAX\_COUNT 100

template<typename T>

struct Node

{

T data;

Node\* pNext = nullptr;

Node\* pPrev = nullptr;

};

template<typename T>

class CQueue

{

public:

explicit CQueue() {};

virtual ~CQueue() {};

public:

bool IsEmpty(void) const

{

return (head == nullptr || tail == nullptr);

}

void Enqueue(T \_data)

{

Node<T>\* nNode = new Node<T>;

nNode->data = \_data;

if (IsEmpty())

{

head = nNode;

tail = nNode;

}

else

{

tail->pNext = nNode;

nNode->pPrev = tail;

tail = nNode;

}

++Count;

if (Count >= Q\_MAX\_COUNT)

{

int tmp;

Dequeue(tmp);

}

}

bool Dequeue(T& data)

{

if (IsEmpty())

return false;

data = head->data;

head->pNext->pPrev = nullptr;

Node<T>\* dNode = head;

head = head->pNext;

delete dNode;

dNode = nullptr;

--Count;

return data;

};

int GetSize()

{

return Count;

}

Node<T>\* Peek()

{

return head;

}

private:

T data;

Node<T>\* head = nullptr;

Node<T>\* tail = nullptr;

int Count = 0;

};

#endif

**WinProc**

// WindowsProjectGraph.cpp : 애플리케이션에 대한 진입점을 정의합니다.

//

#include "framework.h"

#include "WindowsProjectGraph.h"

#include "Queue.h"

//다이얼로그 타이머 ID

#define DIALOG\_TIMER 1

// x축 증가량

int xlapse = 0;

//큐

CQueue<int> queue;

// 전역 변수:

HINSTANCE hInst; // 현재 인스턴스입니다.

// 이 코드 모듈에 포함된 함수의 선언을 전달합니다:

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ int nCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

//윈도우클래스 구조체등록

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_WINDOWSPROJECTGRAPH));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = nullptr;

wcex.lpszClassName = L"CMyDialog";

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

RegisterClassExW(&wcex);

hInst = hInstance; // 인스턴스 핸들을 전역 변수에 저장합니다.

//윈도우 생성

HWND hWnd = CreateWindowW(L"CMyDialog", L"CMyDialog\_Title", WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);

if (!hWnd)

return FALSE;

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

MSG msg;

// 기본 메시지 루프입니다:

while (GetMessage(&msg, nullptr, 0, 0))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return (int)msg.wParam;

}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_CREATE:

{

//========================================

// 타이머 셋팅

//========================================

SetTimer(hWnd, DIALOG\_TIMER, 20, NULL);

}

case WM\_TIMER:

{

//wParam으로 타이머 ID를 구분한다.

switch (wParam)

{

//타이머에서 일정타임마다 큐에 자료형을 넣는다.

case DIALOG\_TIMER:

{

//위아래 반전을 위함(모니터 좌표는 0,0이 좌측상단이므로)

queue.Enqueue(100 - rand() % 100);

//넣고 InvalidateRect()로 WM\_PAINT호출

InvalidateRect(hWnd, NULL, TRUE);

}

}

}

case WM\_COMMAND:

{

int wmId = LOWORD(wParam);

// 메뉴 선택을 구문 분석합니다:

switch (wmId)

{

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

}

break;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

// TODO: 여기에 hdc를 사용하는 그리기 코드를 추가합니다...

RECT WinRect;

GetClientRect(hWnd, &WinRect);

Node<int>\* tNode = queue.Peek();

xlapse = 0;

//시작은 중간지점에서부터 ( 좀이상함 )

MoveToEx(hdc, 0, (WinRect.bottom, 2), NULL);

//데이터가 무조건 0이나 100으로 들어가도 그래프에 굴곡이생긴다.

//처음 MoveTO할때부터 큐에있는 좌표를가지고 하면될것.. 크게상관없어보임

//반복문을 돌면서 그린e다.

while (tNode != nullptr)

{

// x축의 증가량

int xPos = (WinRect.right) / 100 + xlapse;

// y축은 윈도우 높이 \* Qtmp의 백분율 값

int yPos = (WinRect.bottom / 100) \* tNode->data;

//SetPixel(hdc, xPos, yPos, RGB(255, 0, 0));

LineTo(hdc, xPos, yPos);

// 시간경과에따른 x,y좌표의 이동

xlapse += WinRect.right / 100;

tNode = tNode->pNext;

}

EndPaint(hWnd, &ps);

}

break;

case WM\_DESTROY:

//타이머 종료

KillTimer(hWnd, DIALOG\_TIMER);

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// 정보 대화 상자의 메시지 처리기입니다.

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}