#pragma once

//add an application manifest that indicates that ComCtl32.dll version 6 should be used

#pragma comment(linker,"\"/manifestdependency:type='win32' \

name='Microsoft.Windows.Common-Controls' version='6.0.0.0' \

processorArchitecture='\*' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#include "Circle.h"

#include "Rectangle.h"

#include "FilledRectangle.h"

#include "Set.h"

#include <vector>

#define WSeparatorWidth 2

//defining menu id

#define AboutOnMenuClicked 1

#define SaveOnMenuClicked 2

#define LoadOnMenuClicked 3

//defining vidget id

#define RadioButtonSP 10

#define RadioButtonEP 11

#define RadioButtonSetObs 12

#define RadioButtonSelectObs 13

#define StartPointXEdit 14

#define StartPointYEdit 15

#define SetStartPoint 16

#define EndPointXEdit 17

#define EndPointYEdit 18

#define SetEndPoint 19

#define ObsPointA\_xEdit 20

#define ObsPointA\_yEdit 21

#define ObsPointD\_xEdit 22

#define ObsPointD\_yEdit 23

#define SetObsPoint 24

#define SetRemoveObsButton 25

//defining constant

#define WHight 600

#define WWidth 800

#define StartAndEndPointSize 5

#define CaptureSize 6

// other defines

#define GetLastObs (\*(--ObsSet.end()))

#define PumpkinColor RGB(255, 117, 24)

#define EndPointColor RGB(236, 40, 40)

#define StartPointColor RGB(45, 160, 226)

#define CaptureColor RGB(255, 255, 0)

#include <string>

HWND hwnd {0};

HWND cnvs {0};

WNDPROC wpOrigCanvasProc;

bool SetObsLButtonPressed{ false };

bool SelCptrLButtonPressed{ false };

class Rectangle\* SelectedObs{ nullptr };

class Rectangle\* ObsSetCollition(int x, int y);

bool RectCollition(int x, int y, POINT A, POINT C);

bool CapturesIsActive{ false };

class Circle\* SelectedCptr{ nullptr };

class Circle\* CptrSetColision(int x, int y);

HDC cnvsDC {0};

RECT cnvsRCT {0};

PAINTSTRUCT cnvsPS;

//HBRUSH BlueSolidBrush = CreateSolidBrush(RGB(27, 27, 179));

HBRUSH PumpkinSolidBrush = CreateSolidBrush(PumpkinColor);

HWND StartPointX{ 0 }, StartPointY{ 0 };

HWND EndPointX{ 0 }, EndPointY{ 0 };

HWND ObsPointA\_x{ 0 }, ObsPointA\_y{ 0 }, ObsPointD\_x{ 0 }, ObsPointD\_y{ 0 };

class Circle start(cnvs, { WWidth/4, WHight/5 }, StartAndEndPointSize, StartPointColor);

class Circle end(cnvs, { WWidth/2, WHight/3 }, StartAndEndPointSize, EndPointColor);

Set<class Rectangle> ObsSet;

// arr of capture to resize & remove selected obstacle

std::vector<Circle\*> CptrArr (

{ new Circle(cnvs, { WWidth / 2, WHight / 3 }, CaptureSize, CaptureColor, PS\_SOLID, true),

new Circle(cnvs, { WWidth / 2, WHight / 3 }, CaptureSize, CaptureColor, PS\_SOLID, true),

new Circle(cnvs, { WWidth / 2, WHight / 3 }, CaptureSize, CaptureColor, PS\_SOLID, true),

new Circle(cnvs, { WWidth / 2, WHight / 3 }, CaptureSize, CaptureColor, PS\_SOLID, true) }

);

std::vector<RECT> CptrRct(4, {0,0,CaptureSize,CaptureSize});

inline LRESULT CALLBACK SoftwareMainProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

inline LRESULT CALLBACK CanvasSubclassProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

inline WNDCLASSEX CALLBACK NewWindowClass

(

HBRUSH hBrush, HCURSOR hCursor, HINSTANCE hInstance, HICON hIcon, LPCWSTR name, WNDPROC Proc

);

inline void MainWndAddMenu(HWND hwnd);

inline void MainWndAddWidgets(HWND hwnd);

void canvas\_redraw();

void SetGoalPoint(UINT uGOAL);

//inline/\*?\*/ void CreateWindowSeparator(int y, int CWidth);

void MessageBoxError(const std::exception& ex);