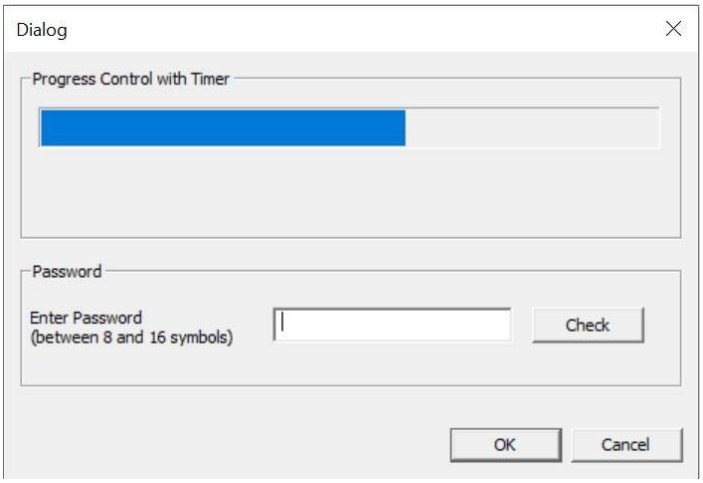
Lab #09

Обработка на съобщения (събития) от клавиатура, таймер и мишка. Примерни програми.

Solution/Project: нов

1. Създаваме нов проект LabKeyMouTmr от тип C++/Windows/Desktop/Application (VS2019)
2. Създаваме нов диалог IDD_KEYMOUTMR, меню елемент IDM_TESTDIALOG, управляваща функция DlgKeyMouTmrFunc и активиране на диалога в WndProc съгласно Lab03-05



ID на контроли, използвани в примерите
- Progress Control: IDC_PROGRESS1
- Edit Control: IDC_ED_PASS
- Button: IDC_BTN_CHECK

3. Съобщенията при натискане и отпускане на бутон на мишката са:
   a. за ляв бутон – `WM_LBUTTONDOWN, WM_LBUTTONUP`
   b. за десен бутон – `WM_RBUTTONDOWN, WM_RBUTTONUP`

   В зависимост от активния прозорец съобщенията се обработват от съответната управляваща функция.

4. Задача: при натискане на десния бутон на мишката в прозореца на приложението да се изчисли дължината на линията от точката на натискане, до точка с координати { 100,100 } и да се изведе съобщение за резултата. Точката най-горе, най-ляво е с координати { 0,0 }, ос X е надясно, ос Y е надолу.
   a. В Windows има структура от тип точка: POINT с два члена x и y
   b. Създайте функция
      `int LenOfLine(POINT pt1, POINT pt2)`
      за изчисляване на разстоянието между две точки по теоремата на Питагор. Ще е необходимо да използвате библиотека `#include "math.h"`
   c. Обработването на съобщението за натиснат десен клавиш на мишката се извършва във функцията `WndProc`.

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
       WCHAR szText[MAX_LOADSTRING];
       POINT pt;
       int iLen;
    switch (message)
```

```
{
    case WM_COMMAND:
            ……

    case WM_RBUTTONUP:
            // в помощна променлива получаваме точката на натискане на десния бутон на мишката
            pt = { LOWORD(lParam), HIWORD(lParam) };
            iLen = LenOfLine(pt, { 100,100 }); // в iLen получаваме резултата от LenOfLine
            // форматираме и извеждаме съобщение за дължината
            StringCbPrintf(szText, ARRAYSIZE(szText),
                    L"Point of click {%i,%i}.\nDistance to target %i.", pt.x, pt.y, iLen);
            MessageBox(hWnd, szText, L"Info", MB_OK);
            break;
```

5. Задача: в прозореца на приложението натискаме левия бутон на мишката, преместваме я и отпускаме бутона. Да се изчисли дължината на линията между точката на натискане и точката на отпускане на левия бутон на мишката.
   Решението зависи от две събития:
   - Натискане: трябва да запазим данните за точката на натискане
   - Отпускане: калкулираме дължината и извеждаме съобщение за резултата

```
POINT ptLMDown;      // the LM button down, global variable
……..

        case WM_LBUTTONDOWN:
                ptLMDown = { LOWORD(lParam), HIWORD(lParam) };
                break;
        case WM_LBUTTONUP:
                pt = { LOWORD(lParam), HIWORD(lParam) };
                iLen = LenOfLine(pt, ptLMDown);
                StringCbPrintf(szText, ARRAYSIZE(szText),
                        L"Line {%i,%i} - {%i,%i}.\nLength of line %i.",
                        ptLMDown.x, ptLMDown.y, pt.x, pt.y, iLen);
                MessageBox(hWnd, szText, L"Info", MB_OK);
                break;
```

6. Съобщенията при натискане и отпускане на клавиш са:
   a. за бутон без натиснат Alt – `WM_KEYDOWN, WM_KEYUP`
   b. за бутон без натиснат Alt – `WM_SYSKEYDOWN, WM_SYSKEYUP` натиснат
   c. натисnатия бутон се проверява за съответната стойност:
      https://docs.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes

   В зависимост от активния прозорец съобщенията се обработват от съответната управляваща функция.

7. Задача: бутоните F5, F6, F7 да се използват за:
   a. F5 – максимизиране на основния прозорец
   b. F6 – възстановяване на нормалния размер на основния прозорец
   c. F7 – минимизиране на основния прозорец
   d. Изпълнение

```
    case WM_KEYDOWN:
            if (wParam == VK_F5)
                    ShowWindow(hWnd, SW_MAXIMIZE);
            if (wParam == VK_F6)
                    ShowWindow(hWnd, SW_RESTORE);
            if (wParam == VK_F7)
                    ShowWindow(hWnd, SW_MINIMIZE);
            break;
```

8. Таймерите се използват за автоматизиране на процеси. Използват се две функции SetTimer за стартиране и KillTimer за спиране. Всеки таймер има идентификатор по аналогия с контролите в диалог. Времето за отброяване е в милисекунди. При завършване трябва да се обработи съобщение WM_TIMER.
9. Задача: при натискане и задържане на ляв бутон на мишката в диалога IDD_KEYMOUTMR да се инициализира progress control и да стартира таймер, който да извършва увеличаване на progress control на всяка 0.1 sec. При отпускане на левия бутон на мишката действието на таймера се прекратява.

   Обработването се извършва в управляващата функция на диалога `DlgKeyMouTmrFunc`.

```
case WM_LBUTTONDOWN:
	SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_SETRANGE, NULL, MAKELPARAM(0, 100));
	SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_SETSTEP, (WPARAM)1, NULL);
	SetTimer(hDlg, TIMER1, 100, NULL);
	return (INT_PTR)TRUE;
case WM_LBUTTONUP:
	KillTimer(hDlg, TIMER1);
	return (INT_PTR)TRUE;
case WM_TIMER:
	SendDlgItemMessage(hDlg, IDC_PROGRESS1, PBM_STEPIT, NULL, NULL);
	return (INT_PTR)TRUE;
```

10. Задача: реализирайте задачата от предната точка, като промените изпълнението за десен бутон на мишката и намаляване на всяка 1 sec на progress control с граници 0-10.
11. Допълнение към Edit Control – контрол на дължината на въведения текст. Ще използваме полето за парола.
    a. Активираме Property Password=True.
    b. При инициализация на диалога установяваме максималната дължина на 16 и променяме символа за парола от „*“ на „o”

```
switch (message)
	{
	case WM_INITDIALOG:
		SendDlgItemMessage(hDlg, IDC_ED_PASS, EM_SETLIMITTEXT, (WPARAM)16, NULL);
		SendDlgItemMessage(hDlg, IDC_ED_PASS, EM_SETPASSWORDCHAR, (WPARAM)'o', NULL);
```

    c. При натискане на бутона за проверка извличаме дължината на въведения текст

```
		case IDC_BTN_CHECK:
			wTmp = SendDlgItemMessage(hDlg, IDC_ED_PASS, EM_LINELENGTH, NULL, NULL);
			if (wTmp < 8)
				MessageBox(hDlg, L"The minimum length is 8!", L"Error", MB_OK);
			return (INT_PTR)TRUE;
```