ใบงานที่ 2 Keyboard and Mouse

# **วัตถุประสงค์**

1. ตรวจจับการทำงานของ Keyboard และ Mouse

2. ใช้เครื่องมือต่างๆ ในการสร้าง Application

# **เนื้อหาเบื้องต้น**

https://github.com/Desktop-Programming-Lab-2559/LAB-02/blob/master/LabIntro.md

# **ลำดับการทดลอง**

1. เรียกโปรแกรม Microsoft Visual Studio

2. สร้าง Project ใหม่ โดยเลือกเมนู File >> New >> Project… (Ctrl+Shift+N)

· ช่อง Templates: ให้เลือก Visual C++ และเลือกชนิด project เป็น Empty Project

· ช่อง Name: ให้ใส่ชื่อของ Project เป็น EasyWin32

· ช่อง Location: ให้เลือกตำแหน่งที่จะสร้าง Project (D:\Student\_Code\LAB\_01\_03)

· ส่วนที่เหลือ ให้คงไว้ตามที่ปรากฏ กด OK

3. เพิ่ม source code ให้กับ project โดยการเลือกเมนู PROJECT >> Add New Item…

· ตั้งชื่อไฟล์เป็น main.cpp

· กด Add เพื่อเพิ่มไฟล์

4. พิมพ์โปรแกรมดังต่อไปนี้ลงในไฟล์ main.cpp

|  |
| --- |
| #include <windows.h>  #include <tchar.h>  // Global Variables:  HINSTANCE hInst; // current instance  TCHAR szTitle[]="Easywin32"; // The title bar text  TCHAR szWindowClass[]="WinApp"; // the class name  // Foward declarations of functions included in this code module:  ATOM MyRegisterClass(HINSTANCE hInstance);  BOOL InitInstance(HINSTANCE, int);  LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  int APIENTRY WinMain(HINSTANCE hInstance,  HINSTANCE hPrevInstance,  LPSTR lpCmdLine,  int nCmdShow)  {  // TODO: Place code here.  MSG msg;  // Register Class  MyRegisterClass(hInstance);  // Perform application initialization:  if (!InitInstance (hInstance, nCmdShow))  {  return FALSE;  }  // Main message loop:  while (GetMessage(&msg, NULL, 0, 0))  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return msg.wParam;  }  //  // FUNCTION: MyRegisterClass()  //  // PURPOSE: Registers the window class.  //  // COMMENTS:  //  // This function and its usage is only necessary if you want this code  // to be compatible with Win32 systems prior to the 'RegisterClassEx'  // function that was added to Windows 95.  //  ATOM MyRegisterClass(HINSTANCE hInstance)  {  WNDCLASSEX wcex;  wcex.cbSize = sizeof(WNDCLASSEX);  wcex.style = CS\_HREDRAW | CS\_VREDRAW;  wcex.lpfnWndProc = (WNDPROC)WndProc;  wcex.cbClsExtra = 0;  wcex.cbWndExtra = 0;  wcex.hInstance = hInstance;  wcex.hIcon = LoadIcon (NULL, IDI\_APPLICATION);  wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);  wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW+1);  wcex.lpszMenuName = NULL;  wcex.lpszClassName = szWindowClass;  wcex.hIconSm = LoadIcon (NULL, IDI\_APPLICATION);  return RegisterClassEx(&wcex);  }  //  // FUNCTION: InitInstance(HANDLE, int)  //  // PURPOSE: Saves instance handle and creates main window  //  // COMMENTS:  //  // In this function, we save the instance handle in a global variable and  // create and display the main program window.  //  BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)  {  HWND hWnd;  hInst = hInstance; // Store instance handle in our global variable  hWnd = CreateWindow(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,  CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);  if (!hWnd)  {  return FALSE;  }  ShowWindow(hWnd, nCmdShow);  UpdateWindow(hWnd);  return TRUE;  }  //  // FUNCTION: WndProc(HWND, unsigned, WORD, LONG)  //  // PURPOSE: Processes messages for the main window.  //  // WM\_COMMAND - process the application menu  // WM\_PAINT - Paint the main window  // WM\_DESTROY - post a quit message and return  //  //  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hdc;  RECT rt;  char szHello[]="Hello, C-Free!";    switch (message)  {  case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...    GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  EndPaint(hWnd, &ps);  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

5. ทำการทดสอบโปรแกรม/บันทึกผล

|  |
| --- |
| ผลการทดลอง |
|  |
| รูป |
|  |

6. เพิ่ม Function Marker() ในโปรแกรม

|  |
| --- |
| void Marker(LONG x, LONG y, HWND hwnd)  {  HDC hdc;  HPEN hPen;  hdc = GetDC(hwnd);  hPen = CreatePen(PS\_DOT,1,RGB(255,0,0));    SelectObject(hdc, hPen);  MoveToEx(hdc, (int) x - 10, (int) y, (LPPOINT) NULL);  LineTo(hdc, (int) x + 10, (int) y);  MoveToEx(hdc, (int) x, (int) y - 10, (LPPOINT) NULL);  LineTo(hdc, (int) x, (int) y + 10);  DeleteObject(hPen);  ReleaseDC(hwnd, hdc);  } |

7. แก้ไขโปรแกรม

|  |  |
| --- | --- |
| case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...  GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  EndPaint(hWnd, &ps);  break; | case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...  GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  Marker(50,50,hWnd);  EndPaint(hWnd, &ps);  break; |

8. ทำการทดสอบโปรแกรม/บันทึกผล

|  |
| --- |
| ผลการทดลอง |
|  |
| รูป |
|  |

9. เพิ่มโปรแกรมตามตาราง

|  |
| --- |
| case WM\_LBUTTONDOWN:  fDraw = TRUE;  ptPrevious.x = LOWORD(lParam);  ptPrevious.y = HIWORD(lParam);  return 0L;    case WM\_LBUTTONUP:  if (fDraw)  {  hdc = GetDC(hWnd);  MoveToEx(hdc, ptPrevious.x, ptPrevious.y, NULL);  LineTo(hdc, LOWORD(lParam), HIWORD(lParam));  ReleaseDC(hWnd, hdc);  }  fDraw = FALSE;  return 0L;    case WM\_MOUSEMOVE:  if (fDraw)  {  hdc = GetDC(hWnd);  MoveToEx(hdc, ptPrevious.x, ptPrevious.y, NULL);  LineTo(hdc, ptPrevious.x = LOWORD(lParam),  ptPrevious.y = HIWORD(lParam));  ReleaseDC(hWnd, hdc);  }  return 0L; |

10. เพิ่มตัวแปร BOOL fDraw = FALSE; POINT ptPrevious; ไว้ที่ Global Area

11. ทำการทดสอบโปรแกรม/บันทึกผล

|  |
| --- |
| ผลการทดลอง |
|  |
| รูป |
|  |

12. เพิ่มตัวแปร COLORREF Color[]={RGB(255,0,0),RGB(0,255,0),RGB(0,0,255),RGB(0,0,0)}; ไว้ที่ Global Area

13. แก้ไข Function Marker() ตามตาราง

|  |
| --- |
| void Marker(LONG x, LONG y,int Index, HWND hwnd)  {  HDC hdc;  HPEN hPen;  hdc = GetDC(hwnd);  hPen = CreatePen(PS\_DOT,1,Color[Index]);    SelectObject(hdc, hPen);  MoveToEx(hdc, (int) x - 10, (int) y, (LPPOINT) NULL);  LineTo(hdc, (int) x + 10, (int) y);  MoveToEx(hdc, (int) x, (int) y - 10, (LPPOINT) NULL);  LineTo(hdc, (int) x, (int) y + 10);  DeleteObject(hPen);  ReleaseDC(hwnd, hdc);  } |

14. แก้ไขโปรแกรมตามตาราง

|  |
| --- |
| case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...    GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  Marker(50,50,Index,hWnd);  EndPaint(hWnd, &ps);  break; |

15. ทำการทดสอบโปรแกรม/บันทึกผล

|  |
| --- |
| ผลการทดลอง |
| เพิ่มตัวแปล Index เข้ามาเก็บค่าสี  แต่ยังไม่มีการเปลี่ยนแปลง |
| รูป |
|  |

16. ทำการเพิ่มโปรแกรมตามตาราง

|  |
| --- |
| case WM\_KEYDOWN:  switch (wParam)  {  case VK\_LEFT:    // Process the LEFT ARROW key.    if(Index<4)  {  Index++;  }else  {  Index=0;  }  InvalidateRect(hWnd, NULL, TRUE);  default:  break;  }  break; |

17. ทำการทดสอบโปรแกรม/บันทึกผล

|  |
| --- |
| ผลการทดลอง |
| จากการทดลองสรุปว่า คำสั่งในชุดนี้สามารถลบ วัตถุที่เราเขียนขึ้น ได้ |
| รูป |
|  |

# สรุปผลการทดลอง

จากการทดลอง สรุปว่า ใบงานที่ 2 Keyboard and Mouse เป็นการเขียนโปรแกรม โดยโปรแกรมจะทำงานเมื่อมีการเปลี่ยนแปลงของ เมาส์ คีย์บอร์ด เช่น 1. หากมีการคลิกเมาส์ ให้โปรแกรม แสดงผลเป็นเส้นตามที่เราลากเมาส์ไป หากเราปล่อยเมาส์จะทำการสิ้นสุดการทำงาน 2.หากมีการแปลี่ยนแปลงของแป้นพิมพ์จะทำให้วัตถุที่เราวาด หายไป

# คำถามท้ายการทดลอง

1. เขียนโปรแกรมวาดรูปที่สามารถเปลี่ยนสีปากกาได้

|  |
| --- |
| ผลการทดลอง |
| ได้ทำการ เพิ่ม code SelectObject(hdc,CreatePen(PS\_SOLID,1,Color[Index])); ในชุดคำสั่ง case WM\_MOUSEMOVE: โดยจะทำการเปลี่ยนสีของวัตถุที่เรา วาดบนหน้าจอ โดยกดเปลี่ยนสีที่ลูกศรบนแป้นพิมพ์ |
| รูป |
|  |

1. เขียนโปรแกรมพิมพ์ข้อมูลจากคีย์บอร์ด แล้วแสดงผลที่หน้าจอ

|  |
| --- |
| ผลการทดลอง |
| ได้ทำการ เพิ่ม code case WM\_CHAR:  hdc = GetDC(hWnd);  Augson[textindext] = (TCHAR)wParam;  textindext++;  TextOut(hdc,250,250,Augson,50);  ReleaseDC(hWnd,hdc);  ซึ่งฟังก์ชัน WM\_CHAR เป็นฟังก์ชั่นในการเขียนตัวอักษร |
| รูป |
|  |