ใบงานที่ 2 Message Map

# **วัตถุประสงค์**

1. เพิ่ม Message Map ในโปรแกรม

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\_02)

· ส่วนที่เหลือ ให้คงไว้ตามที่ปรากฏ กด OK

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

· ตั้งชื่อไฟล์เป็น EasyMain2.cpp

· กด Add เพื่อเพิ่มไฟล์

4. พิมพ์โปรแกรมดังต่อไปนี้ลงในไฟล์ EasyMain2.cpp

#include <windows.h>

LONG WINAPI WndProc (HWND, UINT, WPARAM, LPARAM);

int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance,

LPSTR lpszCmdLine, int nCmdShow)

{

WNDCLASS wc;

HWND hwnd;

MSG msg;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. Define Windows class \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

wc.style = 0; // Class style

wc.lpfnWndProc = (WNDPROC) WndProc; // Window procedure address

wc.cbClsExtra = 0; // Class extra bytes

wc.cbWndExtra = 0; // Window extra bytes

wc.hInstance = hInstance; // Instance handle

wc.hIcon = LoadIcon (NULL, IDI\_WINLOGO); // Icon handle

wc.hCursor = LoadCursor (NULL, IDC\_ARROW); // Cursor handle

wc.hbrBackground = (HBRUSH) (COLOR\_WINDOW + 1); // Background color

wc.lpszMenuName = NULL; // Menu name

wc.lpszClassName = "MyWndClass"; // WNDCLASS name

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. Register the Windows class \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

RegisterClass (&wc);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3. Create window \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

hwnd = CreateWindow (

"MyWndClass", // WNDCLASS name

"SDK Application", // Window title

WS\_OVERLAPPEDWINDOW, // Window style

CW\_USEDEFAULT, // Horizontal position

CW\_USEDEFAULT, // Vertical position

CW\_USEDEFAULT, // Initial width

CW\_USEDEFAULT, // Initial height

HWND\_DESKTOP, // Handle of parent window

NULL, // Menu handle

hInstance, // Application's instance handle

NULL // Window-creation data

);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4. Display the window \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ShowWindow (hwnd, nCmdShow);

UpdateWindow (hwnd);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5. Message loop \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

while (GetMessage (&msg, NULL, 0, 0)) {

TranslateMessage (&msg);

DispatchMessage (&msg);

}

return msg.wParam;

}

LRESULT CALLBACK WndProc (HWND hwnd, UINT message, WPARAM wParam,

LPARAM lParam)

{

PAINTSTRUCT ps;

HDC hdc;

switch (message) {

case WM\_PAINT:

hdc = BeginPaint (hwnd, &ps);

Ellipse (hdc, 10, 10, 200, 100);

EndPaint (hwnd, &ps);

return 0;

case WM\_DESTROY:

PostQuitMessage (0);

return 0;

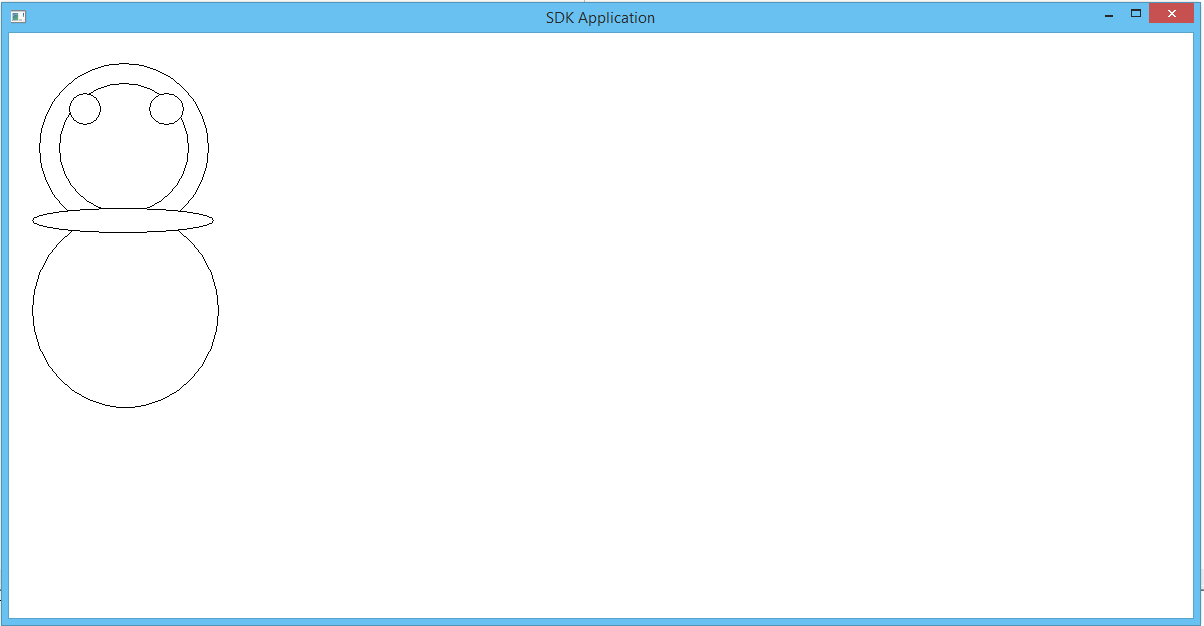
}

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

}

4. กดปุ่ม F5 เพื่อดูผลการทำงานของโปรแกรม

5. บันทึกผล



6. ปรับค่า wc (WNDCLASS) มา 5 ค่า

7. บันทึกผล

8. ปรับค่าพารามิเตอร์ของฟังก์ชั่น CreateWindow() มา 5 ค่า

9. บันทึกผล

10. เปลี่ยนฟังก์ชั่น Ellipse() เป็นฟังก์ชันวาดรูป

1. จุด SetPixel()
2. วงกลม
3. สี่เหลี่ยม
4. สามเหลี่ยม
5. เส้นตรง
6. เส้นโค้ง

11. แก้ไขฟังก์ชั่นในการวาดรูป

1. เปลี่ยนสีรูปทรง
2. เปลี่ยนสีเส้น
3. เปลี่ยนลักษณะเส้น
4. เปลี่ยนสีพื้นหลัง

12. เขียนโปรแกรมวาดรูป Doremon

# บันทึกผล

ตารางการเปลี่ยนค่า WNDCLASS

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.hbrBackground = (HBRUSH) (COLOR\_WINDOW + 1); // Background color | wc.hbrBackground = (HBRUSH) (COLOR\_WINDOW + 9); // Background color |
| ผลการทดลอง | |
| สีพื้นหลังมีการเปลี่ยนแปลงจากสีขาวเป็นสีฟ้า | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.hCursor = LoadCursor (NULL, IDC\_ARROW); // Cursor handle | wc.hCursor = LoadCursor (NULL, IDC\_HELP); // Cursor handle |
| ผลการทดลอง | |
| เปลี่ยนรูป cursor จากเครื่องหมายลูกศรเป็นเครื่องหมายลูกศรและมีเครื่องหมายคำถามห้อยอยู่ | |
| รูป | |
| https://scontent.fbkk1-3.fna.fbcdn.net/v/t34.0-12/21100607_1838073879542708_642968417_n.jpg?oh=58ab028478524b03e1dc4ef35dccece8&oe=59A0FC98 | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.hIcon = LoadIcon (NULL,IDI\_WINLOGO); // Icon handle | wc.hIcon = LoadIcon (NULL,IDI\_QUESTION); // Icon handle |
| ผลการทดลอง | |
| เปลี่ยนไอคอนมุมบนทางซ้ายจากไอคอนรูปวินโดวเป็นไอคอนที่มีเครื่องหมายคำถามพื้นหลังสีน้ำเงิน | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.style = 0; // Class style | wc.style = CS\_NOCLOSE; // Class style |
| ผลการทดลอง | |
| ไม่สามารถกดเครื่องหมายกากบาทปิดโปรแกรมได้ | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.lpszClassName = "MyWndClass"; // WNDCLASS name | wc.lpszClassName = "Naphorn"; // WNDCLASS name |
| ผลการทดลอง | |
| เมื่อเปลี่ยนค่า โปรแกรมจะสามารถรันได้เมื่อเปลี่ยนใน CreateWindow ด้วย | |
| รูป | |
|  | |

ตารางการเปลี่ยนค่าพารามิเตอร์ของฟังก์ชั่น CreateWindow()

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| "SDK Application", // Window title | "Naphorn Chokpokasombat", // Window title |
| ผลการทดลอง | |
| เปลี่ยนชื่อหน้าต่างจาก SDK Application เป็น Naphorn Chokpokasombat | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| WS\_OVERLAPPEDWINDOW, // Window style | WS\_OVERLAPPEDWINDOW | WS\_VSCROLL, // Window style |
| ผลการทดลอง | |
| มีแถบเลื่อนขึ้นลงทางด้านขวาเพิ่มขึ้นมา | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| WS\_OVERLAPPEDWINDOW, // Window style | 0, // Window style |
| ผลการทดลอง | |
| แถบสํญลักษณ์ข้างบนหายไป | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| CW\_USEDEFAULT, // Horizontal position  CW\_USEDEFAULT, // Vertical position | 0, // Horizontal position  0, // Vertical position |
| ผลการทดลอง | |
| ระยะห่างจากขอบตามที่กำหนด Horizontal แกน x (แนวนอน), Vertical แกน y (แกนตั้ง) | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| CW\_USEDEFAULT, // Initial width  CW\_USEDEFAULT, // Initial height | 600, // Initial width  120, // Initial height |
| ผลการทดลอง | |
| ปรับขนาดหน้าต่างตามที่ต้องการ width คือ ความกว้าง, height คือ ความสูง | |
| รูป | |
|  | |

ตารางการเปลี่ยนฟังก์ชั่นในการวาดรูป

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| Ellipse (hdc, 10, 10, 200, 100); | SetPixel(hdc,500,501,RGB(0,255,0)); |
| ผลการทดลอง | |
| ใช้ SetPixel จะทำให้เกิดจุดขึ้นมานั้น สามารถเปลี่ยนสีได้ | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| SetPixel(hdc,500,501,RGB(0,255,0)); | Ellipse (hdc, 20, 20, 100, 100); |
| ผลการทดลอง | |
| กำหนดให้ค่า x,y แกนเท่ากัน จะทำให้รูปออกมาเป็นวงกลม | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| Ellipse (hdc, 20, 20, 100, 100); | Rectangle (hdc, 10, 10, 200, 200); |
| ผลการทดลอง | |
| ใช้ Rectangle และกำหนดระยะห่างขอบตามแกน x, แกน y เท่ากัน และความกว้างกับความยาวเท่ากัน จะทำให้เกิดรูปสี่เหลี่ยม | |
| รูป | |
|  | |
| ค่าเดิม | ค่าใหม่ |
| Rectangle (hdc, 10, 10, 200, 200); | LineTo(hdc, 130, 250); |
| ผลการทดลอง | |
| ใช้ LineTo ในการทำให้เกิดเส้นตรงตามที่เรากำหนด | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| LineTo(hdc, 130, 250); | LineTo(hdc, 130, 250);  LineTo(hdc, 130, 0);  LineTo(hdc, 0, 0); |
| ผลการทดลอง | |
| ใช้ LineTo ทั้ง 3 เส้นต่อกัน ทำให้เกิดเป็นรูปสามเหลี่ยม | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| LineTo(hdc, 130, 250);  LineTo(hdc, 130, 0);  LineTo(hdc, 0, 0); | Arc(hdc,200,200,400,400,200,200,400,400); |
| ผลการทดลอง | |
| ใช้ Arc ทำให้เกิดเส้นโค้ง | |
| รูป | |
|  | |

ตารางการเปลี่ยนค่าพารามิเตอร์ของฟังก์ชั่นในการวาดรูป

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| Rectangle (hdc, 200, 200, 400, 400);  Arc(hdc,200,200,400,400,200,200,400,400); | SelectObject(hdc,CreateSolidBrush(RGB(10,950,600)));  Rectangle (hdc, 200, 200, 400, 400);  Arc(hdc,200,200,400,400,200,200,400,400); |
| ผลการทดลอง | |
| รูปสี่เหลี่ยมที่มีเส้นโค้งอยู่ข้างในจะมีพื้นหลังเป็นสีเขียว | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| SelectObject(hdc,CreateSolidBrush(RGB(10,950,600)));  Rectangle (hdc, 200, 200, 400, 400);  Arc(hdc,200,200,400,400,200,200,400,400); | SelectObject(hdc,CreateSolidBrush(RGB(10,950,600)));  SelectObject(hdc,CreatePen(PS\_DASHDOT, 3, RGB(10,500,100)));  Rectangle (hdc, 200, 200, 400, 400);  Arc(hdc,200,200,400,400,200,200,400,400); |
| ผลการทดลอง | |
| เกิดเส้นขอบรูปทรงที่มีสีเขียวอ่อน | |
| รูป | |
|  | |
| ค่าเดิม | ค่าใหม่ |
| SelectObject(hdc,CreatePen(PS\_DASHDOT, 3, RGB(10,500,100))); | SelectObject(hdc,CreatePen(PS\_DOT, 1, RGB(10,500,100))); |
| ผลการทดลอง | |
| เกิดเส้นที่มีลักษณะเป็นเส้นประ | |
| รูป | |
|  | |

|  |  |
| --- | --- |
| ค่าเดิม | ค่าใหม่ |
| wc.hbrBackground = (HBRUSH) (COLOR\_WINDOW + 9); // Background color | wc.hbrBackground = (HBRUSH) (COLOR\_WINDOW + 12); // Background color |
| ผลการทดลอง | |
| พื้นหลังจากสีฟ้าเปลี่ยนเป็นสีเทา | |
| รูป | |
|  | |

โปรแกรมวาดรูป Doreamon

|  |
| --- |
| โปรแกรม |
| case WM\_PAINT:  hdc = BeginPaint (hwnd, &ps);  //Rectangle (hdc, 40, 40, 260, 260);  //Rectangle (hdc, 40, 600, 260, 260);  //Leg  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 50, 410, 150, 450);  Ellipse (hdc, 150, 410, 250, 450);  //Arm  SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));  Ellipse (hdc, 25, 262, 125, 315);  Ellipse (hdc, 175, 262, 275, 315);  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 5, 305, 50, 270);  Ellipse (hdc, 295, 305, 250, 270);  //Body  SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));  SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));  Ellipse (hdc, 58, 238, 240, 430);  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 75, 245, 225, 390);  //Bag  Chord(hdc,92,370,208,220,40,362,322,400);  //Face  SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));  Ellipse (hdc, 40, 40, 260, 260);  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 60, 90, 240, 260);  //Eye1  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 90, 55, 150, 128);  Ellipse (hdc, 150, 55, 210, 128);  //Eye2  SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));  Ellipse (hdc, 95, 80, 145, 128);  Ellipse (hdc, 155, 80, 205, 128);  //Eye3  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 128, 105, 142, 121);  Ellipse (hdc, 158, 105, 172, 121);  //Nose Red  SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));  Ellipse (hdc, 165, 116, 135, 144);  //Nose Shadow  SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));  Ellipse (hdc, 162, 131, 152, 142);  //FaceLine  MoveToEx(hdc, 150, 143, NULL); LineTo(hdc, 150, 190);  ///Light  MoveToEx(hdc, 65, 132, NULL); LineTo(hdc, 135, 146);  MoveToEx(hdc, 55, 152, NULL); LineTo(hdc, 135, 158);  MoveToEx(hdc, 65, 175, NULL); LineTo(hdc, 135, 170);  ///Right  MoveToEx(hdc, 235, 132, NULL); LineTo(hdc, 165, 146);  MoveToEx(hdc, 235, 152, NULL); LineTo(hdc, 165, 158);  MoveToEx(hdc, 235, 175, NULL); LineTo(hdc, 165, 170);  //Lip  SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));  Chord(hdc,84,100,215,240,101,185,300,217);  //Neck  SelectObject(hdc,CreatePen(PS\_DASH, 17, RGB(255,0,0)));  MoveToEx(hdc, 92, 255, NULL); LineTo(hdc, 205, 255); //Ring  SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));  SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));  Ellipse (hdc, 163, 268, 138, 245);  EndPaint (hwnd, &ps);  return 0; |
| ผลการทดลอง |
| สร้างรูป Doreamon จากฟังก์ชันหลายๆ ฟังก์ชัน เช่น ฟังก์ชันวงรี, เส้นโค้ง, วงกลม, ครึ่งวงกลมและเส้นตรง แล้วทำการลงสีตามส่วนต่างๆ |
| รูป |
|  |

# สรุปผลการทดลอง

จากการทดลองนี้คือ การเขียนโค้ดสามารถโดยใช้ฟังก์ชันในการวาดรูปได้ เช่น ฟังก์ชัน Ellipse (วงรี) หากเราปรับแกน x, y เท่ากันและความกว้าง, ความยาวเท่ากัน จะทำให้เกิดรูปวงกลม, ฟังก์ชัน SetPixel (จุด), ฟังก์ชัน Arc (เส้นโค้ง), ฟังก์ชัน Rectangle (สี่เหลี่ยม), ฟังก์ชัน LineTo (เส้นตรง) และฟังก์ชัน Chord (ครึ่งวงกลม)

# คำถามท้ายการทดลอง

1. อธิบายการทำงานของโปรแกรมในข้อ 4

ตอบ เพื่อที่เราจะสามารถสร้าง Windows ได้ เราต้องทำการวางพื้นฐานกับตัวข้อมูล การตั้งแบบแสดงผล ตั้งที่เก็บข้อมูล ตั้งไอคอน ตั้งเคอร์เซอร์ กำหนดและปรับแต่งสีพื้นหลัง ตั้งชื่อ Class Name จึงจะสามารถลงทะเบียนเพื่อให้จดจำได้ แล้วต่อไปจะเป็นการสร้างหน้าต่างแสดงผล ไม่ว่าจะการตั้งชื่อ ตั้งค่าขนาดตามแนวแกน ความกว้าง ความยาว เมื่อสร้างหน้าต่างเสร็จ จึงทำให้หน้าต่างนี้มีการแสดงผล และอัพเดทผล การแสดงผลนี้จะเป็นไปตามเงื่อนไขที่ใช้ ซึ่งในโปรแกรมนี้ใช้เงื่อนไข switch { case… } ในการทำฟังก์ชันเกี่ยวกับรูปร่าง ก็คือ วงรี(Ellipse) ซึ่งวงรีก็สามารถกำหนดค่าได้ตามแนวแกน ความกว้าง และความยาว และอีกฟังก์ชันคือ ฟังก์ชันการประมวลผลข้อมูล

1. บอกพารามิเตอร์ของ WNDCLASS

* Style ประเภท UINT : การระบุ class styles ของหน้าต่างคลาส
* IpfnWndProc ประเภท WNDPROC : การบอกขั้นตอนที่ได้ทำเก็บไว้ที่ WindowProc
* cdClsExtra ประเภท int : จำนวนไบต์ที่จัดสรรตามโครงสร้างระดับหน้าต่าง โดยระบบเริ่มต้นไบต์เป็นศูนย์
* cdWndExtra ประเภท int : จำนวนไบต์ที่จัดสรรตามตัวอย่างหน้าต่าง ระบบไบต์เริ่มต้นเป็นศูนย์
* hInstance ประเภท : HINSTANCE : หมายเลขอ้างอิงไปยังอินสแตนด์ที่ประกอบด้วยขั้นตอนของหน้าต่างสำหรับคลาส
* hIcon ประเภท HICON : หมายเลขอ้างอิงไปยังไอคอนชั้นเรียน ถ้าสมาชิกนี้เป็น NULL ระบบจะมีไอคอนเริ่มต้น
* hCursor ประเภท HCURSOR : หมายเลขอ้างอิงไปยังเคอร์เซอร์ชั้นเรียน ถ้าสมาชิกนี้เป็น NULL แอพพลิเคชันต้องกำหนดรูปเคอร์เซอร์อย่างชัดเจน เมื่อเมาส์เลื่อนไปมาที่หน้าต่างแอพพลิเคชัน
* hbrBackground ประเภท HBRUSH : สำหรับแปรงพื้นหลังของชั้นเรียน
* lpszMenuName ประเภท LPCTSTR : ชื่อของเมนูคลาส ถ้าสมาชิกนี้เป็น NULL หน้าต่างที่อยู่ในคลาสนี้ไม่มีเมนูเริ่มต้น
* lpszClassName ประเภท LPCTSTR : เป็นสตริงจะระบุชื่อคลาสหน้าต่าง ชื่อคลาสสามารถเป็นชื่อที่ลงทะเบียนกับ RegisterClass หรือชื่อคลาสที่กำหนดไว้ล่วงหน้า ความยาวสูงสุดคือ 256 ถ้ามีค่ามากกว่าความยาวสูงสุด RegisterClass จะไม่ทำงาน

1. บอกพารามิเตอร์ และหน้าที่ของฟังก์ชั่น CreateWidow

* lpClassName ประเภท LPCTSTR : ถ้าคำสั่งสูงต้องเป็นศูนย์ แต่ถ้าเป็นสตริงจะระบุชื่อคลาสหน้าต่าง
* lpWindowName ประเภท LPCTSTR : ชื่อหน้าต่าง หากสไตล์หน้าต่างระบุแถบชื่อตาม lpWindowName จะปรากฏในแถบชื่อเรื่อง
* dwStyle ประเภท DWORD : สไตล์ของหน้าต่างที่สร้างขึ้น
* x ประเภท int : ตำแหน่งแนวนอนเริ่มต้นของหน้าต่าง
* y ประเภท int : ตำแหน่งแนวตั้งเริ่มต้นของหน้าต่าง
* nWidth ประเภท int : ความกว้างในหน่วยอุปกรณ์ของหน้าต่าง
* nHeight ประเภท int : ความสูงในหน่วยอุปกรณ์ของหน้าต่าง
* hWndParent ประเภท HWND : หมายเลขอ้างอิงไปยังหน้าต่างที่กำลังจะสร้างขึ้น
* hMenu ประเภท HMENU : หมายเลขอ้างอิงไปยังเมนูขึ้นอยู่กับลักษณะหน้าต่าง สามารถเป็นค่า NULL ได้
* hInstance ประเภท HINSTANCE : จัดการกับอินสแตนด์ที่เชื่อมโยงกับหน้าต่าง
* lpParam ประเภท LPVOID : อาจเป็นค่า NULL ถ้าไม่มีข้อมูลเพิ่มเติม

1. บอกหน้าที่ของฟังก์ชั่น RegisterClass(), ShowWindow(), UpdateWindow(), TranslateMessage(), DispatchMessage()

* ฟังก์ชัน RegisterClass() : ลงทะเบียนหน้าต่างคลาส สำหรับการใช้งานที่เรียกไปยังฟังก์ชัน CreateWindow
* ฟังก์ชัน ShowWindow() : การตั้งค่าการแสดงของหน้าต่างที่ระบุ ควบคุมหน้าต่างที่จะแสดง
* ฟังก์ชัน UpdateWindow() : ฟังก์ชันการปรับปรุงพื้นที่ของหน้าต่างที่ระบุโดยการส่งพารามิเตอร์ไปที่หน้าต่าง
* ฟังก์ชัน TranslateMessage() : การแปลข้อความเสมือนคีย์เป็นข้อความตัวอักษร
* ฟังก์ชัน DispatchMessage() : การส่งข้อความไปยังขั้นตอนของหน้าต่าง โดยปกติจะใช้เพื่อส่งข้อความที่ดึงข้อมูล

1. บอกชื่อและหน้าที่ WM\_MESSAGE มา 10 ชื่อ

* WM\_NULL : ไม่มีการดำเนินการ โปรแกรมจะส่งข้อความนี้ไป หากต้องการโพสต์ข้อความที่หน้าต่างผู้รับจะไม่สนใจ
* WM\_DESTROY : ข้อความนี้จะส่งไปเมื่อหน้าต่างถูกทำลาย จะถูกส่งไปยังหน้าต่างขั้นตอนของหน้าต่างที่ถูกทำลาย หลังจากหน้าต่างถูกลบออกจากหน้าจอ
* WM\_CREATE : ข้อความจะถูกส่ง เมื่อมีการร้องขอโปรแกรมประยุกต์ที่หน้าต่างที่ถูกสร้างขึ้นโดยการเรียกฟังก์ชัน CreateWindow
* WM\_QUIT : การบอกถึงคำขอที่ประยุกต์ใช้และสร้างขึ้นเมื่อแอพพลิเคชันเรียกใช้ PostQuitMessage
* WM\_GETICON : ข้อความที่ส่งไปที่หน้าต่างเพื่อดึงข้อมูลไปยังไอคอนขนาดใหญ่หรือขนาดเล็กที่เชื่อมโยงกับหน้าต่าง ระบบจะแสดงไอคอนขนาดใหญ่ในกล่องโต้ตอบ ALT + TAB และไอคอนขนาดเล็กในคำอธิบายภาพของหน้าต่าง
* WM\_MOVE : เป็นข้อความที่ถูกส่งหลังจากย้ายหน้าต่างแล้ว
* WM\_NCCREATE : เป็นข้อความที่ส่งก่อนข้อความ WM\_CREATE เมื่อสร้างหน้าต่างขึ้นมา
* WM\_FONTCHANGE : โปรแกรมจะส่งข้อความนี้ไปยังหน้าต่างระดับบนสุดทั้งหมดในระบบหลังจากเปลี่ยนแหล่งข้อมูลแบบอักษร
* WM\_PAINT : ข้อความนี้จะถูกส่งเมื่อระบบทำการร้องขอในการวาดส่วนของหน้าต่างของโปรแกรม ข้อความถูกส่งไปเมื่อเรียกใช้ฟังก์ชัน UpdateWindow
* WM\_ENDSESSION : ข้อความแจ้งการประยุกต์ใช้ช่วงที่กำลังจะสิ้นสุด

1. เขียนโปรแกรมสร้าง Doraemon จำนวน 5 ตัว

case WM\_PAINT:

hdc = BeginPaint (hwnd, &ps);

//Leg

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 50, 410, 150, 450);

Ellipse (hdc, 150, 410, 250, 450);

//Arm

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 25, 262, 125, 315);

Ellipse (hdc, 175, 262, 275, 315);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 5, 305, 50, 270);

Ellipse (hdc, 295, 305, 250, 270);

//Body

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 58, 238, 240, 430);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 75, 245, 225, 390);

//Bag

Chord(hdc,92,370,208,220,40,362,322,400);

//Face

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 40, 40, 260, 260);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 60, 90, 240, 260);

//Eye1

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 90, 55, 150, 128);

Ellipse (hdc, 150, 55, 210, 128);

//Eye2

SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));

Ellipse (hdc, 95, 80, 145, 128);

Ellipse (hdc, 155, 80, 205, 128);

//Eye3

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 128, 105, 142, 121);

Ellipse (hdc, 158, 105, 172, 121);

//Nose Red

SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));

Ellipse (hdc, 165, 116, 135, 144);

//Nose Shadow

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 162, 131, 152, 142);

//FaceLine

MoveToEx(hdc, 150, 143, NULL); LineTo(hdc, 150, 190);

///Light

MoveToEx(hdc, 65, 132, NULL); LineTo(hdc, 135, 146);

MoveToEx(hdc, 55, 152, NULL); LineTo(hdc, 135, 158);

MoveToEx(hdc, 65, 175, NULL); LineTo(hdc, 135, 170);

///Right

MoveToEx(hdc, 235, 132, NULL); LineTo(hdc, 165, 146);

MoveToEx(hdc, 235, 152, NULL); LineTo(hdc, 165, 158);

MoveToEx(hdc, 235, 175, NULL); LineTo(hdc, 165, 170);

//Lip

SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));

Chord(hdc,84,100,215,240,101,185,300,217);

//Neck

SelectObject(hdc,CreatePen(PS\_DASH, 17, RGB(255,0,0)));

MoveToEx(hdc, 92, 255, NULL); LineTo(hdc, 205, 255);

//Ring

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));

Ellipse (hdc, 163, 268, 138, 245);

/\*\*2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

//Leg

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 340, 215, 395, 240);

Ellipse (hdc, 395, 215, 450, 240);

//Arm

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 320, 140, 400, 170);

Ellipse (hdc, 395, 140, 470, 170);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 315, 142, 335, 170);

Ellipse (hdc, 475, 142, 455, 170);

//Body

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 340, 133, 450, 230);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 351, 135, 440, 205);

//Bag

Chord(hdc,365,190,425,135,240,145,420,160);

//Face

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 330, 10, 460, 140);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 345, 45, 445, 140);

//Eye1

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 370, 30, 395, 62);

Ellipse (hdc, 395, 30, 420, 62);

//Eye2

SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));

Ellipse (hdc, 373, 40, 392, 62);

Ellipse (hdc, 398, 40, 417, 62);

//Eye3

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 383, 52, 390, 59);

Ellipse (hdc, 400, 52, 407, 59);

//Nose Red

SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));

Ellipse (hdc, 402, 58, 388, 71);

//Nose Shadow

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 400, 65, 396, 69);

//FaceLine

MoveToEx(hdc, 395, 70, NULL); LineTo(hdc, 395, 100);

///Light

MoveToEx(hdc, 350, 68, NULL); LineTo(hdc, 388, 74);

MoveToEx(hdc, 348, 79, NULL); LineTo(hdc, 388, 81);

MoveToEx(hdc, 350, 91, NULL); LineTo(hdc, 388, 87);

///Right

MoveToEx(hdc, 440, 68, NULL); LineTo(hdc, 402, 74);

MoveToEx(hdc, 442, 79, NULL); LineTo(hdc, 402, 81);

MoveToEx(hdc, 440, 91, NULL); LineTo(hdc, 402, 87);

//Lip

SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));

Chord(hdc,355,35,435,125,200,182,602,184);

//Neck

SelectObject(hdc,CreatePen(PS\_DASH, 10, RGB(255,0,0)));

MoveToEx(hdc, 365, 138, NULL);

LineTo(hdc, 425, 138);

//Ring

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));

Ellipse (hdc, 388, 145, 403, 130);

/\*\*3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

//Leg

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+340, 215, 180+395, 240);

Ellipse (hdc, 180+395, 215, 180+450, 240);

//Arm

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+320, 140, 180+400, 170);

Ellipse (hdc, 180+395, 140, 180+470, 170);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+315, 142, 180+335, 170);

Ellipse (hdc, 180+475, 142, 180+455, 170);

//Body

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+340, 133, 180+450, 230);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+351, 135, 180+440, 205);

//Bag

Chord(hdc,180+365,190,180+425,135,180+240,145,180+420,160);

//Face

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+330, 10, 180+460,140);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+345, 45, 180+445, 140);

//Eye1

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+370, 30, 180+395, 62);

Ellipse (hdc, 180+395, 30, 180+420, 62);

//Eye2

SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));

Ellipse (hdc, 180+373, 40, 180+392, 62);

Ellipse (hdc, 180+398, 40, 180+417, 62);

//Eye3

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+383, 52, 180+390, 59);

Ellipse (hdc, 180+400, 52, 180+407, 59);

//Nose Red

SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));

Ellipse (hdc, 180+402, 58, 180+388, 71);

//Nose Shadow

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+400, 65, 180+396, 69);

//FaceLine

MoveToEx(hdc, 180+395, 70, NULL); LineTo(hdc, 180+395, 100);

///Light

MoveToEx(hdc, 180+350, 68, NULL); LineTo(hdc, 180+388, 74);

MoveToEx(hdc, 180+348, 79, NULL); LineTo(hdc, 180+388, 81);

MoveToEx(hdc, 180+350, 91, NULL); LineTo(hdc, 180+388, 87);

///Right

MoveToEx(hdc, 180+440, 68, NULL); LineTo(hdc, 180+402, 74);

MoveToEx(hdc, 180+442, 79, NULL); LineTo(hdc, 180+402, 81);

MoveToEx(hdc, 180+440, 91, NULL); LineTo(hdc, 180+402, 87);

//Lip

SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));

Chord(hdc,180+355,35,180+435,125,180+200,182,180+602,184);

//Neck

SelectObject(hdc,CreatePen(PS\_DASH, 10, RGB(255,0,0)));

MoveToEx(hdc, 180+365, 138, NULL);

LineTo(hdc, 180+425, 138);

//Ring

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));

Ellipse (hdc, 180+388, 145, 180+403, 130);

//////\*\*4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

//Leg

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+340, 215, 180+180+395, 240);

Ellipse (hdc, 180+180+395, 215, 180+180+450, 240);

//Arm

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+180+320, 140, 180+180+400, 170);

Ellipse (hdc, 180+180+395, 140, 180+180+470, 170);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+315, 142, 180+180+335, 170);

Ellipse (hdc, 180+180+475, 142, 180+180+455, 170);

//Body

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+180+340, 133, 180+180+450, 230);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+351, 135, 180+180+440, 205);

//Bag

Chord(hdc,180+180+365,190,180+180+425,135,180+180+240,145,180+180+420,160);

//Face

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 180+180+330, 10, 180+180+460,140);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+345, 45, 180+180+445, 140);

//Eye1

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+370, 30, 180+180+395, 62);

Ellipse (hdc, 180+180+395, 30, 180+180+420, 62);

//Eye2

SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));

Ellipse (hdc, 180+180+373, 40, 180+180+392, 62);

Ellipse (hdc, 180+180+398, 40, 180+180+417, 62);

//Eye3

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+383, 52, 180+180+390, 59);

Ellipse (hdc, 180+180+400, 52, 180+180+407, 59);

//Nose Red

SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));

Ellipse (hdc, 180+180+402, 58, 180+180+388, 71);

//Nose Shadow

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 180+180+400, 65, 180+180+396, 69);

//FaceLine

MoveToEx(hdc, 180+180+395, 70, NULL); LineTo(hdc, 180+180+395, 100);

///Light

MoveToEx(hdc, 180+180+350, 68, NULL); LineTo(hdc, 180+180+388, 74);

MoveToEx(hdc, 180+180+348, 79, NULL); LineTo(hdc, 180+180+388, 81);

MoveToEx(hdc, 180+180+350, 91, NULL); LineTo(hdc, 180+180+388, 87);

///Right

MoveToEx(hdc, 180+180+440, 68, NULL); LineTo(hdc, 180+180+402, 74);

MoveToEx(hdc, 180+180+442, 79, NULL); LineTo(hdc, 180+180+402, 81);

MoveToEx(hdc, 180+180+440, 91, NULL); LineTo(hdc, 180+180+402, 87);

//Lip

SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));

Chord(hdc,180+180+355,35,180+180+435,125,180+180+200,182,180+180+602,184);

//Neck

SelectObject(hdc,CreatePen(PS\_DASH, 10, RGB(255,0,0)));

MoveToEx(hdc, 180+180+365, 138, NULL);

LineTo(hdc, 180+180+425, 138);

//Ring

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));

Ellipse (hdc, 180+180+388, 145, 180+180+403, 130);

//////\*\*4\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

//Leg

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+340, 215, 170+180+180+395, 240);

Ellipse (hdc, 170+180+180+395, 215, 170+180+180+450, 240);

//Arm

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 170+180+180+320, 140, 170+180+180+400, 170);

Ellipse (hdc, 170+180+180+395, 140, 170+180+180+470, 170);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+315, 142, 170+180+180+335, 170);

Ellipse (hdc, 170+180+180+475, 142, 170+180+180+455, 170);

//Body

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 170+180+180+340, 133, 170+180+180+450, 230);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+351, 135, 170+180+180+440, 205);

//Bag

Chord(hdc,170+180+180+365,190,170+180+180+425,135,170+180+180+240,145,170+180+180+420,160);

//Face

SelectObject(hdc,CreateSolidBrush(RGB(27,134,240)));

Ellipse (hdc, 170+180+180+330, 10, 170+180+180+460,140);

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+345, 45, 170+180+180+445, 140);

//Eye1

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+370, 30, 170+180+180+395, 62);

Ellipse (hdc, 170+180+180+395, 30, 170+180+180+420, 62);

//Eye2

SelectObject(hdc,CreateSolidBrush(RGB(0,0,0)));

Ellipse (hdc, 170+180+180+373, 40, 170+180+180+392, 62);

Ellipse (hdc, 170+180+180+398, 40, 170+180+180+417, 62);

//Eye3

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+383, 52, 170+180+180+390, 59);

Ellipse (hdc, 170+180+180+400, 52, 170+180+180+407, 59);

//Nose Red

SelectObject(hdc,CreateSolidBrush(RGB(255,0,0)));

Ellipse (hdc, 170+180+180+402, 58, 170+180+180+388, 71);

//Nose Shadow

SelectObject(hdc,CreateSolidBrush(RGB(255,255,255)));

Ellipse (hdc, 170+180+180+400, 65, 170+180+180+396, 69);

//FaceLine

MoveToEx(hdc, 170+180+180+395, 70, NULL); LineTo(hdc, 170+180+180+395, 100);

///Light

MoveToEx(hdc, 170+180+180+350, 68, NULL); LineTo(hdc, 170+180+180+388, 74);

MoveToEx(hdc, 170+180+180+348, 79, NULL); LineTo(hdc, 170+180+180+388, 81);

MoveToEx(hdc, 170+180+180+350, 91, NULL); LineTo(hdc, 170+180+180+388, 87);

///Right

MoveToEx(hdc, 170+180+180+440, 68, NULL); LineTo(hdc, 170+180+180+402, 74);

MoveToEx(hdc, 170+180+180+442, 79, NULL); LineTo(hdc, 170+180+180+402, 81);

MoveToEx(hdc, 170+180+180+440, 91, NULL); LineTo(hdc, 170+180+180+402, 87);

//Lip

SelectObject(hdc,CreateSolidBrush(RGB(255,70,0)));

Chord(hdc,170+180+180+355,35,170+180+180+435,125,170+180+180+200,182,170+180+180+602,184);

//Neck

SelectObject(hdc,CreatePen(PS\_DASH, 10, RGB(255,0,0)));

MoveToEx(hdc, 170+180+180+365, 138, NULL);

LineTo(hdc, 170+180+180+425, 138);

//Ring

SelectObject(hdc,CreatePen(PS\_SOLID, 1, RGB(0,0,0)));

SelectObject(hdc,CreateSolidBrush(RGB(255,255,0)));

Ellipse (hdc, 170+180+180+388, 145, 170+180+180+403, 130);

EndPaint (hwnd, &ps);

return 0;