Системное программирование

Дедов Никита, 251003

Лабораторная №2

#include <windows.h>

#include <math.h>

#include "strsafe.h"

#define NUM\_ROWS 3

#define NUM\_COLUMNS 4

#define M\_PI 3.14159265358979323846

#define RADIUS 200

#define ELL\_WIDTH 550

#define ELL\_HEIGHT 300

LRESULT CALLBACK WndProc(HWND hWnd, UINT message,

WPARAM wParam, LPARAM lParam);

int ChangeString(char\* string, int width, HDC hdc);

void DrawRotatedText(HDC hdc, const char\* text, int x, int y, double angle);

int tangent\_angle(double x, double y, double width, double height);

int APIENTRY WinMain(HINSTANCE hInstance,

HINSTANCE hPrevInstance, LPTSTR lpCmdLine, int nCmdShow)

{

WNDCLASSEX wcex; HWND hWnd; MSG msg;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_DBLCLKS;

wcex.lpfnWndProc = 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 = "Lab2";

wcex.hIconSm = wcex.hIcon;

RegisterClassEx(&wcex);

hWnd = CreateWindow("Lab2", L"Second lab",

WS\_OVERLAPPEDWINDOW, CW\_USEDEFAULT, 0,

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

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

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

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return (int)msg.wParam;

}

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

{

char\* table[NUM\_ROWS][NUM\_COLUMNS] = {

{L"We're no strangers to love\nYou know the rules and so do I",

L"A full commitment's what I'm thinking of\nYou wouldn't get this from any other guy",

L"I just wanna tell you how I'm feeling\nGotta make you understand",

L"Never gonna give you up\nNever gonna let you down"},

{L"Never gonna run around and desert you\nNever gonna make you cry",

L"Never gonna say goodbye\nNever gonna tell a lie and hurt you",

L"We've known each other for so long\nYour heart's been aching but you're too shy to say it",

L"Inside we both know what's been going on\nWe know the game and we're gonna play it"},

{L"And if you ask me how I'm feeling\nDon't tell me you're too blind to see",

L"Never gonna give you up\nNever gonna let you down",

L"Never gonna run around and desert you\nNever gonna make you cry",

L"Never gonna say goodbye\nNever gonna tell a lie and hurt you"}

};

static int rowHeights[NUM\_ROWS] = { 0 };

static int colWidth[NUM\_COLUMNS] = { 0 };

static RECT clientRect;

GetClientRect(hwnd, &clientRect);

switch (message) {

case WM\_SIZE: {

for (int i = 0; i < NUM\_ROWS; i++)

{

HDC hdc = GetDC(hwnd);

int rem = (clientRect.right - clientRect.left) % NUM\_COLUMNS;

int maxHeight = 0;

for (int j = 0; j < NUM\_COLUMNS; j++)

{

char tmp[256];

size\_t convertedChars = wcstombs(tmp, table[i][j], sizeof(tmp));

colWidth[j] = (clientRect.right - clientRect.left) / NUM\_COLUMNS;

if (rem > 0)

{

colWidth[j]++;

rem--;

}

int l = colWidth[j];

int Height = ChangeString(tmp, colWidth[j] - 2, hdc);

if (Height > maxHeight)

maxHeight = Height;

}

rowHeights[i] = maxHeight;

}

InvalidateRect(hwnd, NULL, TRUE);

} break;

case WM\_PAINT: {

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hwnd, &ps);

HBRUSH hBrush = CreateSolidBrush(RGB(255, 255, 255));

SelectObject(hdc, hBrush);

for (int i = 0; i < NUM\_ROWS; i++)

{

for (int j = 0; j < NUM\_COLUMNS; j++)

{

RECT rect;

rect.left = 0;

for (int k = 0; k < j; k++)

rect.left += colWidth[k] - 1;

rect.right = rect.left + colWidth[j] + j;

rect.top = 0;

for (int k = 0; k < i; k++)

rect.top += rowHeights[k];

rect.bottom = rect.top + rowHeights[i] + 1;

Rectangle(hdc, rect.left, rect.top, rect.right, rect.bottom);

rect.top += 2;

rect.left += 1;

rect.right -= 1;

DrawText(hdc, table[i][j], -1, &rect, DT\_CENTER | DT\_VCENTER | DT\_WORDBREAK);

}

}

int hTable = 0;

for (int i = 0; i < NUM\_ROWS; hTable += rowHeights[i], i++);

RECT rect;

rect.left = 50;

rect.top = hTable + 50;

rect.right = rect.left + RADIUS \* 2;

rect.bottom = rect.top + RADIUS \* 2;

int centerX = (rect.left + rect.right) / 2;

int centerY = (rect.top + rect.bottom) / 2;

Ellipse(hdc, rect.left, rect.top, rect.right, rect.bottom);

const char\* text = "Circle is round apparently...Which is weird, but okay I guess";

int textLength = strlen(text);

double angleStep = 0.1;// 2 \* M\_PI / textLength;

for (int i = 0; i < textLength; i++) {

double angle = i \* angleStep + M\_PI;

int angleDeg = 2700 - (round)(angle \* 180 / M\_PI) \* 10;

int x = (int)(centerX + (RADIUS - 0) \* cos(angle));

int y = (int)(centerY + (RADIUS - 0) \* sin(angle));

HGDIOBJ hfnt, hfntPrev;

WCHAR lpszRotate[2];

lpszRotate[0] = (WCHAR)text[i];

lpszRotate[1] = L'\0';

HRESULT hr;

size\_t pcch = 2;

PLOGFONT plf = (PLOGFONT)LocalAlloc(LPTR, sizeof(LOGFONT));

hr = StringCchCopy(plf->lfFaceName, 6, TEXT("Arial"));

plf->lfWeight = FW\_NORMAL;

SetBkMode(hdc, TRANSPARENT);

plf->lfEscapement = angleDeg;

hfnt = CreateFontIndirect(plf);

hfntPrev = SelectObject(hdc, hfnt);

hr = StringCchLength(lpszRotate, 2, &pcch);

TextOut(hdc, x, y, lpszRotate, pcch);

SelectObject(hdc, hfntPrev);

DeleteObject(hfnt);

SetBkMode(hdc, OPAQUE);

LocalFree((LOCALHANDLE)plf);

}

DeleteObject(hBrush);

EndPaint(hwnd, &ps);

} break;

case WM\_DESTROY: {

PostQuitMessage(0);

} break;

default:

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

}

return 0;

}

int ChangeString(char\* string, int width, HDC hdc)

{

int heightOfText = GetTextHeight(hdc, string);

int height = heightOfText + 5;

int strLen = strlen(string);

int indOfNewLine = -1;

int indOfSpace = -1;

char\* newLine = (char\*)malloc((strLen + 1) \* sizeof(char));

memset(newLine, 0, ((strLen + 1) \* sizeof(char)));

int i = 0, k = 0;

while (i < strLen)

{

for (int j = 0; i < strLen && string[i] != '\n' && string[i] != ' '; i++, j++, k++)

{

newLine[k] = string[indOfSpace + 1 + j];

}

SIZE size;

GetTextExtentPoint32A(hdc, newLine, strlen(newLine), &size);

if (size.cx > width || string[i] == '\n')

{

if (size.cx > width && string[i] == '\n')

height += heightOfText;

byte isOneWord = 1;

for (int i = 0; i < strlen(newLine) && isOneWord; i++)

if (newLine[i] == ' ')

isOneWord = 0;

if (isOneWord)

{

indOfNewLine = i;

indOfSpace = i;

}

else if (string[i] != '\n')

{

indOfNewLine = indOfSpace;

i = indOfSpace;

}

else

{

indOfNewLine = i;

indOfSpace = i;

}

k = 0;

memset(newLine, 0, ((strLen + 1) \* sizeof(char)));

height += heightOfText;

}

else

{

newLine[strlen(newLine)] = ' ';

k++;

indOfSpace = i;

}

i++;

}

return height;

}

int GetTextHeight(HDC hdc, const char\* str) {

TEXTMETRIC tm;

GetTextMetrics(hdc, &tm);

return tm.tmHeight + tm.tmExternalLeading;

}

int tangent\_angle(double x, double y, double width, double height) {

double dy\_dx = -(height \* height \* x) / (width \* width \* y);

double angle = atan(dy\_dx);

int res = (round)(angle \* (180.0 / M\_PI));

if (y >= 0)

res = res + 180;

return res;

}