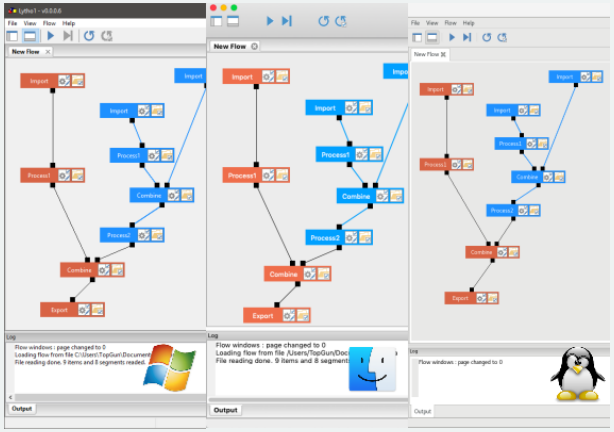
МИЭТ
Национальный исследовательский университет «МИЭТ»
Кафедра ПКИМС

Кроссплатформенная разработка программного обеспечения



Лабораторная работа №1

Нативные API. Win32 API.

Минимальный каркас оконной программы под WinAPI: WinMain

Task_01

```
#include <windows.h>
#include <string.h>

#define szWindowClass "MyWindow"
#define szTitle "A Simple Window"

int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
  WNDCLASSEX wcex;

  wcex.cbSize = sizeof(WNDCLASSEX);
  wcex.style = CS_HREDRAW | CS_VREDRAW;
  wcex.lpfnWndProc = WndProc;
  wcex.cbClsExtra = 0;
  wcex.cbWndExtra = 0;
  wcex.hInstance = hInstance;
  wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_APPLICATION));
  wcex.hCursor = LoadCursor(NULL, IDC_ARROW);
  wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
  wcex.lpszMenuName = NULL;
  wcex.lpszClassName = szWindowClass;
  wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI_APPLICATION));

  if (!RegisterClassEx(&wcex)) {
    MessageBox(NULL, "Can't register window class!", "Win32 API Test", NULL);
    return 1;
  }

  HWND hWnd = CreateWindow(szWindowClass, szTitle, WS_OVERLAPPEDWINDOW, CW_USEDEFAULT,
    CW_USEDEFAULT, 500, 400, NULL, NULL, hInstance, NULL);

  if (!hWnd) {
    MessageBox(NULL, "Can't create window!", "Win32 API Test", NULL);
    return 1;
  }

  ShowWindow(hWnd, SW_SHOWNORMAL);
  UpdateWindow(hWnd);

  MSG msg;
  while (GetMessage(&msg, NULL, 0, 0)) {
    TranslateMessage(&msg);
    DispatchMessage(&msg);
  }

  return msg.wParam;
}
```

Инициализация параметров создаваемого окна

Регистрация окна с нашими параметрами в системе

Создание экземпляра окна

Показать окно

Запуск обработчика событий

\+ оконная процедура

Минимальный каркас оконной программы под WinAPI: WndProc

```
long __stdcall WndProcedure(HWND   hWnd,
                            UINT   Msg,
                            WPARAM wParam,
                            LPARAM lParam) {
  switch (Msg) {
    case WM_DESTROY:
      PostQuitMessage(WM_QUIT);
      break;

    case WM_PAINT:
      ...

    case WM_LBUTTONDOWN:
      ...

    case WM_SIZE:
      ...

    default:
      return DefWindowProc(hWnd, Msg, wParam, lParam);
  }
  return 0;
}
```

Определение координат клика

Задание: при клике левой кнопкой мыши выводить сообщение, в котором будут координаты клика

C++

```
#define WM_LBUTTONDOWN                  0x0201
```

Значение параметра wParam:

MK_CONTROL
MK_SHIFT

Значение параметра lParam:

Координаты в виде двойного слова, вытащить которые можно с помощью макросов:
LOWORD, HIWORD
или
GET_X_PARAM, GET_Y_PARAM

Диалог для вывода информации на экран

Задание: при клике левой кнопкой мыши выводить сообщение, в котором будут координаты клика

C++

```
int WINAPI MessageBox(
  HWND    hWnd,
  LPCTSTR lpText,
  LPCTSTR lpCaption,
  UINT    uType
);
```

Типы кнопок:

```
MB_OK
MB_OKCANCEL
MB_YESNO
MB_YESNOCANCEL
…
```

Типы иконок:

```
MB_ICONEXCLAMATION
MB_ICONWARNING
MB_ICONINFORMATION
MB_ICONASTERISK
MB_ICONQUESTION
…
```

Коды возврата:

```
IDOK
IDYES
IDCANCEL
…
```

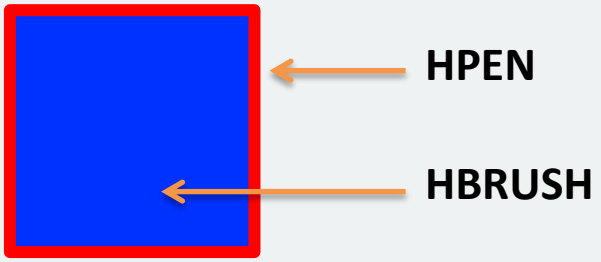
Работа с GDI (1)

Task_02

```
LRESULT CALLBACK WndProc(HWND hWnd,
                         UINT message,
                         WPARAM wParam,
                         LPARAM lParam) {
  PAINTSTRUCT ps;
  HDC hdc;

  switch (message) {

    case WM_PAINT:
      hdc = BeginPaint(hWnd, &ps);

      ...

      TextOut(hdc, 10, 10, "Hello, World!", strlen("Hello, World!"));

      ...

      EndPaint(hWnd, &ps);

    break;
```

Работа с GDI (2)

```
HPEN pen, old_pen;
HBRUSH brush, old_brush;

...

pen = CreatePen(PS_SOLID, 1, RGB(255, 0, 0));
old_pen = (HPEN)SelectObject(hDC, pen);

brush = CreateSolidBrush(RGB(0, 50, 255));
old_brush = (HBRUSH)SelectObject(hDC, brush);

Rectangle(hDC, 10, 0, 150, 150);

SelectObject(hDC, old_pen);
SelectObject(hDC, old_brush);
DeleteObject(pen);
DeleteObject(brush);
```



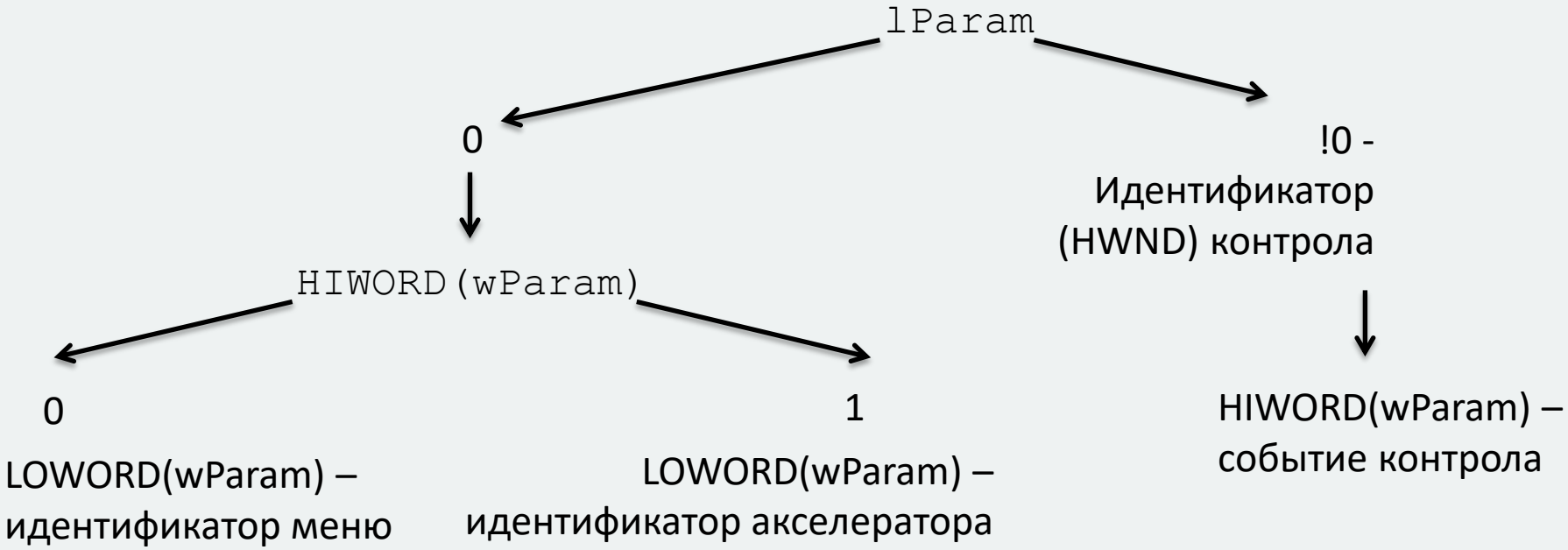
Вызов контекстного меню

```
case WM_RBUTTONDOWN: {
   HMENU popup = CreatePopupMenu();

   AppendMenu(popup, MF_STRING, 0, L"Exit");

   POINT point = { LOWORD(lParam), HIWORD(lParam) };

   ClientToScreen(hWnd, &point);

   TrackPopupMenu(popup,
     TPM_LEFTBUTTON,
     point.x, point.y,
     0,
     hWnd,
     NULL);

   DestroyMenu(popup);
   break;
 }
```

Обработка событий меню: сообщение WM_COMMAND

C++

```
#define WM_COMMAND                    0x0111
```



Написание обработчика событий меню

```
AppendMenu(popup, MF_STRING, 4131, L"Exit");
```

```
case WM_COMMAND:
   if (LOWORD(wParam) == 4131)
     SendMessage(hWnd, WM_CLOSE, 0, 0);
   break;
```

Реализация строки меню окна

```
switch (message) {
  case WM_CREATE:
    HMENU hMenubar = nullptr;
    HMENU hMenu = nullptr;

    hMenubar = CreateMenu();
    hMenu = CreateMenu();

    AppendMenu(hMenu, MF_STRING, 4131, L"&Quit");

    AppendMenu(hMenubar, MF_POPUP, (UINT_PTR)hMenu, L"&File");
    SetMenu(hWnd, hMenubar);
    break;
```

Использование файлов ресурсов (1)

Task_03

Файл - Resource.h:

```
#define IDR_MYMENU 101
#define IDI_MYICON 201

#define IDM_FILE_OPEN 9001
#define IDM_FILE_EXIT 9002
```

Файл - Resource.rc:

```
#include "resource.h"

IDR_MYMENU MENU
BEGIN
  POPUP "&File"
  BEGIN
    MENUITEM "&Open", IDM_FILE_OPEN
    MENUITEM "E&xit", IDM_FILE_EXIT
  END
END

IDI_MYICON ICON "menu_one.ico"
```

Использование файлов ресурсов (2)

```
#include "resource.h"

  ...

  WNDCLASSEX wcex;

  ...

  wcex.lpszMenuName    = MAKEINTRESOURCE(IDR_MYMENU);
```

Создание контролов и обработка событий от них

```
HWND hBtn;
...

 hBtn = CreateWindow(L"button", L"Push me", WS_CHILD | WS_VISIBLE,
                     0, 0, 200, 50, hWnd, NULL,
                     hInstance, NULL);
...

 case WM_COMMAND:
    if (lParam != 0) {
      if ((HWND)lParam == hBtn)
        if (HIWORD(wParam) == BN_CLICKED)
          MessageBox(hWnd, L"Button clicked!", L"Cap", MB_OK);
    }
    else
      if (LOWORD(wParam) == 4131)
        PostMessage(hWnd, WM_CLOSE, 0, 0);
    break;
```

Использование диалога открытия файлов

```
OPENFILENAME ofn;
char szFileName[MAX_PATH] = "";
ZeroMemory(&ofn, sizeof(ofn));
ofn.lStructSize = sizeof(ofn);
ofn.hwndOwner = hWnd;
ofn.lpstrFilter = "Text Files (*.txt)\0*.txt\0All Files (*.*)\0*.*\0";
ofn.lpstrFile = szFileName;
ofn.nMaxFile = MAX_PATH;
ofn.Flags = OFN_EXPLORER | OFN_FILEMUSTEXIST | OFN_HIDEREADONLY;
ofn.lpstrDefExt = "txt";
if (GetOpenFileName(&ofn))
  MessageBox(hWnd, ofn.lpstrFile, "File name", MB_OK);
}
```

Результат возвращается в ofn.lpstrFile. Именно этот файл и нужно открывать и считывать из него данные.
В данном примере я вывожу имя фала в качестве текста сообщения в MessageBox.

Задание на дом

Написать программу с использованием WinAPI, которая рисует схемотехнические элементы: резистор, биполярный npn-транзистор, источник тока. Информация о выбранном элементе и его положении сохраняется в момент закрытия программы и подгружается в момент её запуска.

В программе должны быть пункты меню File и Element.
В меню File слудующие пункты:

- «Open...» - загрузить файл с информацией об элементе (с вызовом диалога)
- «Save As...» - сохранить информацию об элементе в файл (с вызовом диалога)
- Разделитель
- «Exit» - выход с подтверждением

В меню Element следующие пункты:

- resistor – при выборе пункта меню программа рисует резистор
- npn BJT – при выборе пункта меню программа рисует биполярный транзистор
- current source – при выборе пункта меню программа рисует источник тока

Когда двигаем мышкой – в заголовке окна выводятся координаты курсора.
Когда кликаем мышкой – меняется положение рисуемого элемента.