



WinAPI + DLL

Матрохин Дмитрий Александрович
Голованов Роман Вячеславович

26.11.18



Структура лекции

Основные разделы и подразделы.

❑ WinAPI

- Platform SDK,
- Объекты ядра,
- Описатели объектов ядра
- «Жизненный цикл» объекта

❑ Пользовательский интерфейс

- Окна,
- Взаимодействие с пользователем,
- Оконные сообщения,
- Архитектура обработки сообщений,
 - Очередь
 - Отправка/обработка

❑ DLLs

- Предыстория,
- Жизненный цикл разработки
- Адресное пространство
- Экспортируемые функции
- Поиск dll

❑ Использование dll в коде

- Неявное связывание
- Явное связывание
 - ::Load/::Free Library

❑ Недостатки dll



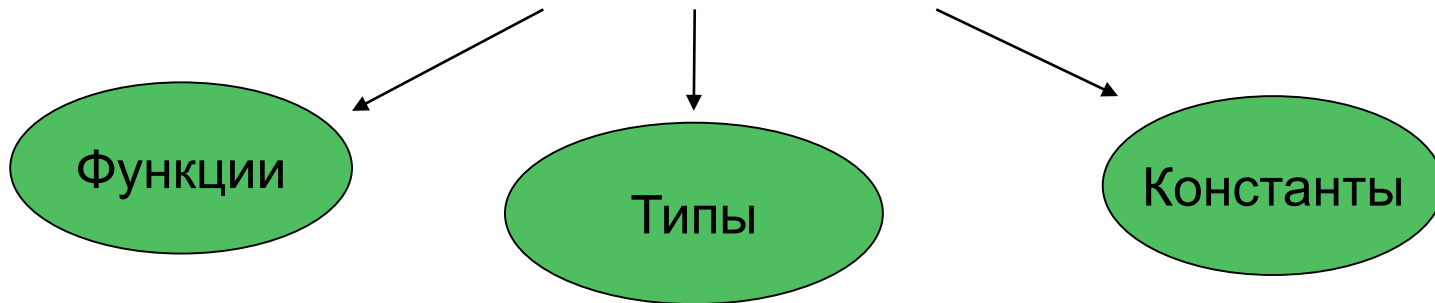
WinAPI

Windows Application Programming Interface

- ❑ Набор разработчика (Platform SDK),
- ❑ Объекты ядра,
- ❑ Описатели объектов ядра
- ❑ «Жизненный цикл» объекта

WinAPI: Platform SDK (Software Development Kit)

Win32 Application Programming Interface



Platform SDK:

- библиотеки
- заголовочные файлы
- примеры
- документация

WinAPI: Объекты ядра

Ресурсы операционной системы

Event

Process

Thread

Mutex

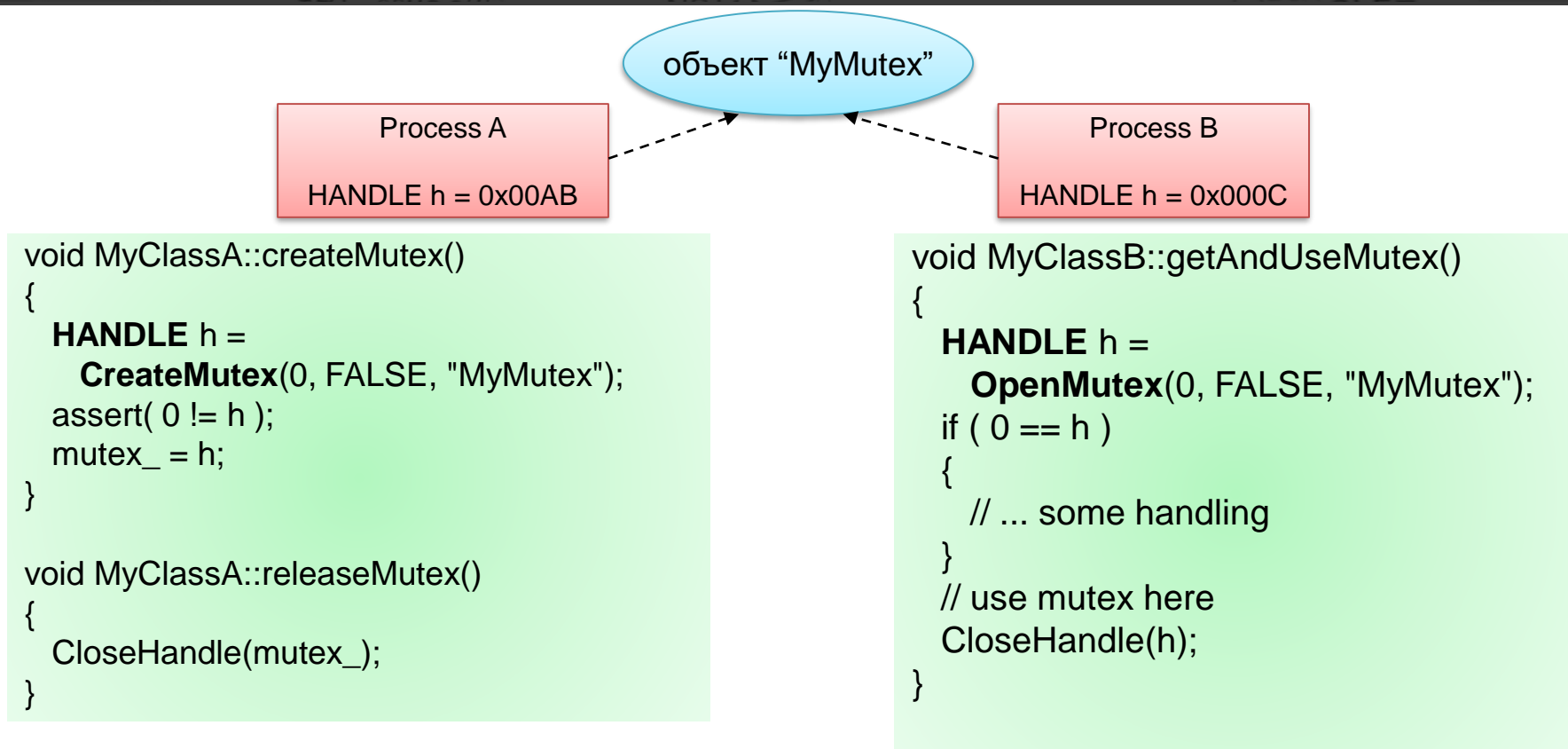
Semaphore

File

CreateEvent(...)
CreateMutex(...)
CreateProcess(...)
CreateThread(...)

OpenEvent(...)
OpenMutex(...)
OpenProcess(...)
OpenThread(...)

WinAPI: Описатели объектов ядра



WinAPI: «Жизненный цикл» объекта: Создание

1. Вызов CreateMutex

Process A

Process B

4. Вызов OpenMutex

3. CreateMutex
возвращает handle

Handle 1

Handle 2

5. OpenMutex
возвращает handle

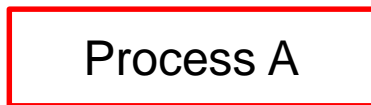
Mutex
“MyMutex”

2. CreateMutex создаёт
mutex в памяти

WinAPI:

«Жизненный цикл» объекта: Удаление

1. Вызов CloseHandle

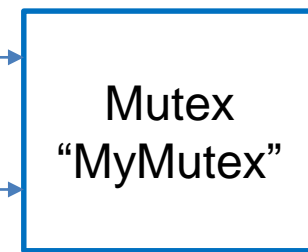


2. CloseHandle
закрывает handle



3. Вызов CloseHandle

4. CloseHandle
закрывает handle



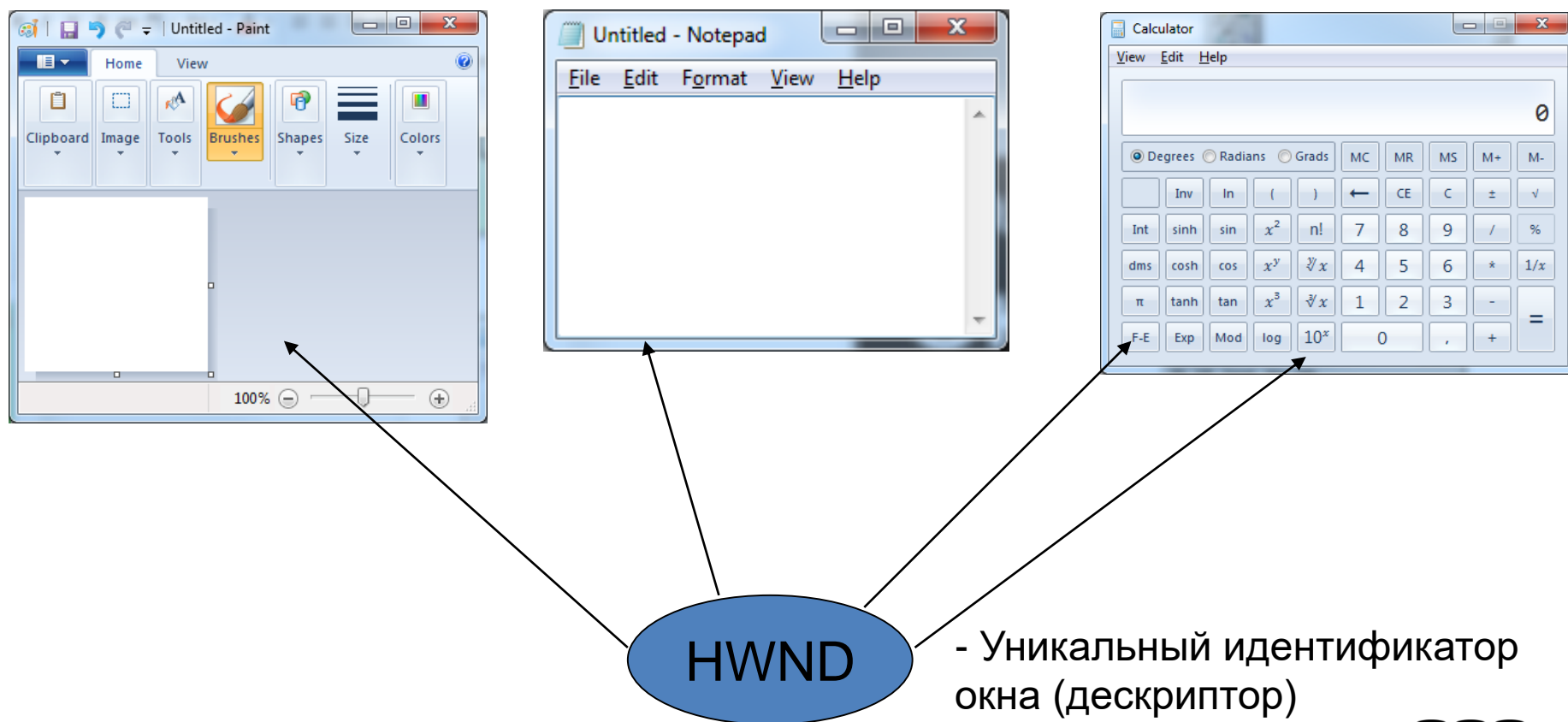
5. После закрытия
последнего handle на объект,
ОС удаляет его из памяти

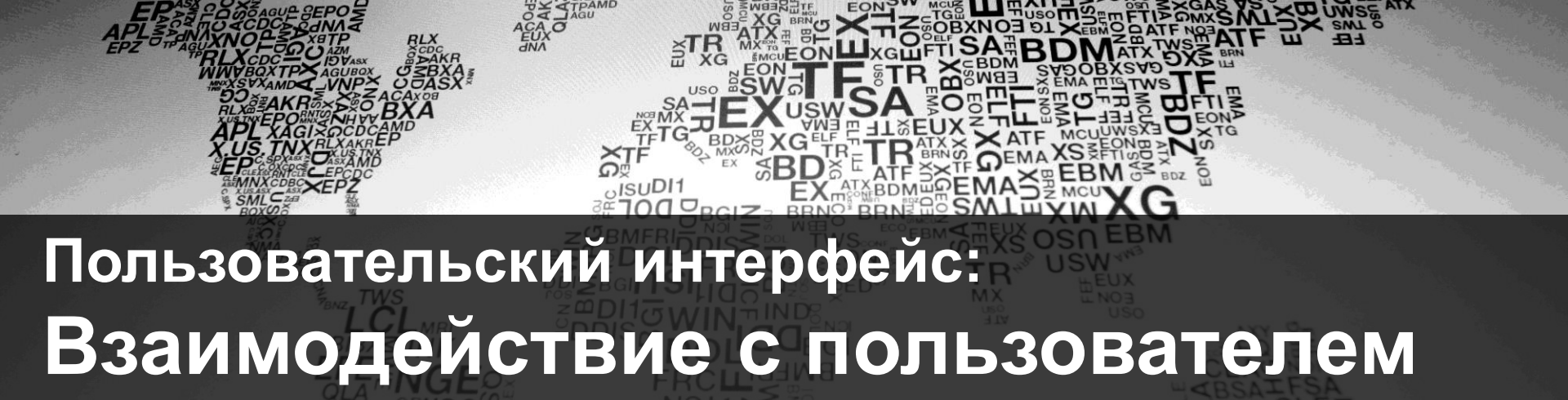


Пользовательский интерфейс

- ❑ Окна,
- ❑ Взаимодействие с пользователем,
- ❑ Оконные сообщения,
- ❑ Архитектура обработки сообщений,
 - Очередь
 - Отправка/обработка

Пользовательский интерфейс: Окна





Пользовательский интерфейс: Взаимодействие с пользователем

Пакетный подход

```
int a;  
int b;  
  
printf("Type a\n");  
scanf("%d", &a);  
printf("Type b\n");  
scanf("%d", &b);  
  
printf("a + b = %d\n", a + b);
```

Событийно-ориентированный подход (Event-driven paradigm)

```
void processEvent(Event e);  
  
while (true) {  
    const Event e = get_event();  
    processEvent(e);  
}  
  
void processEvent(Event e) {  
    switch (e.type) {  
        case keyPressed: /* process keyboard */ break;  
        case mouseClicked: /* process mouse*/ break;  
        case timer: /* process timer*/ break;  
    }  
}
```

Пользовательский интерфейс: Оконные сообщения

```
LRESULT WndProc(HWND hWnd, UINT message,  
                WPARAM wParam, LPARAM lParam)
```

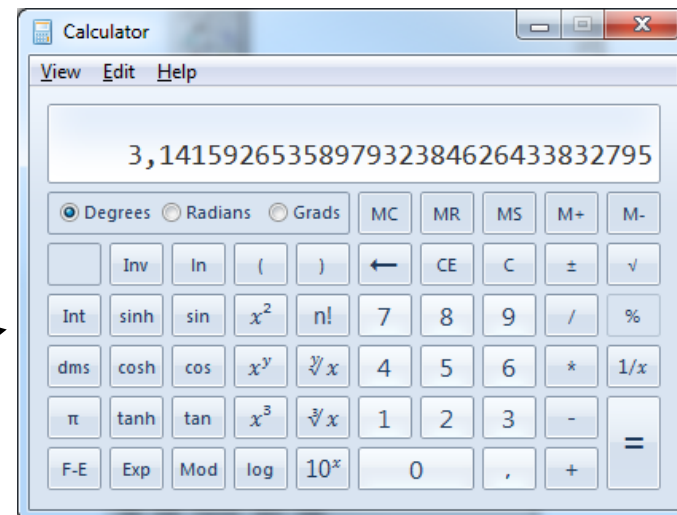
```
{  
    switch (message.msg)  
    {  
        case WM_KEYDOWN:  
            //handle  
            break;  
        case WM_LBUTTONDOWN:  
            // handle  
            break;  
        case WM_TIMER:  
            // handle  
            break;  
        case WM_MOVE:  
            // handle  
            break;  
    }  
}
```

WM_KEYDOWN

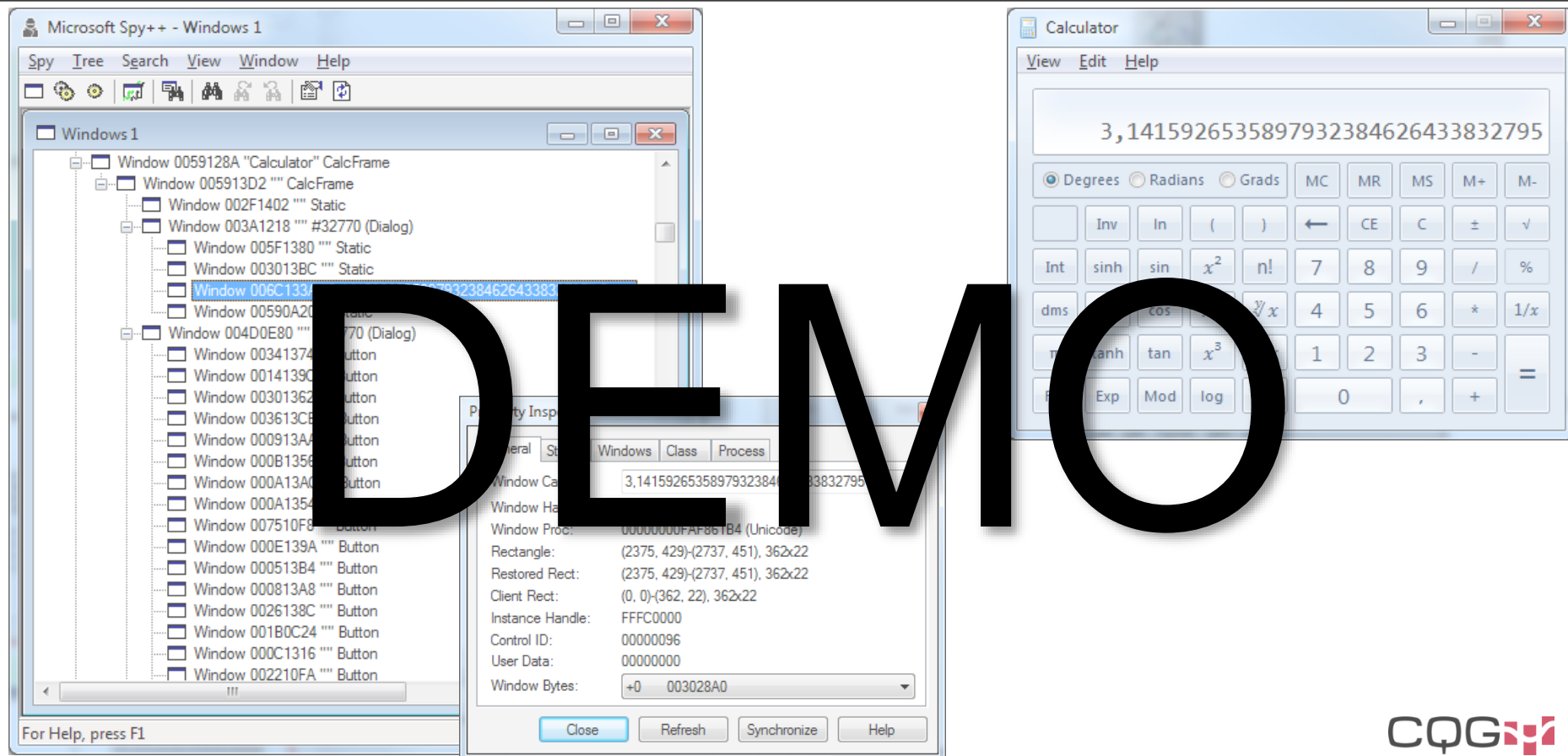
WM_MOVE

WM_TIMER

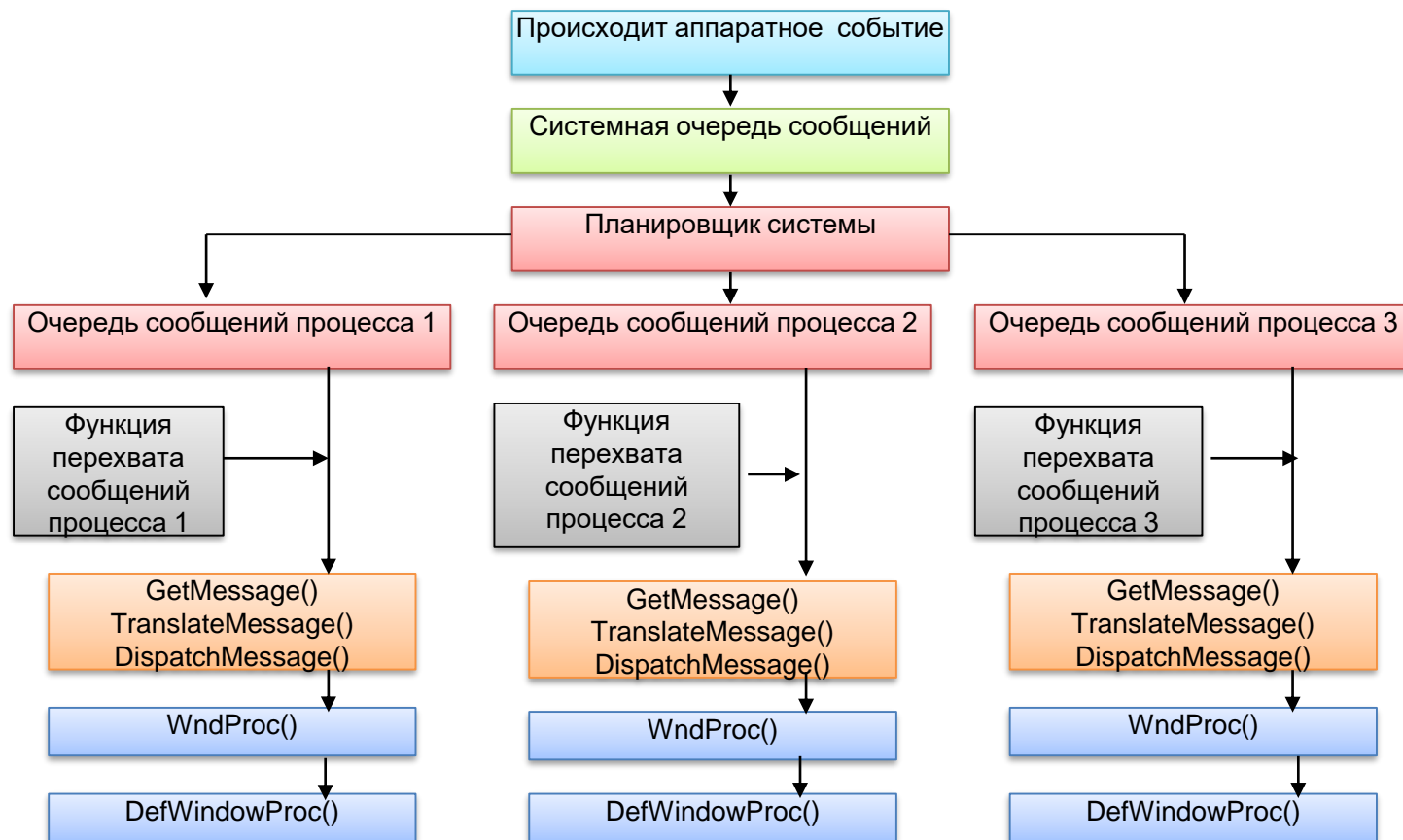
WM_LBUTTONDOWN



Пользовательский интерфейс: Окна в Microsoft Spy++



Пользовательский интерфейс: Архитектура обработки сообщений



Пользовательский интерфейс: Очередь сообщений



```
void RunMessageLoop() {  
    MSG msg;  
    while (GetMessage(&msg, 0, 0, 0)) {  
        TranslateMessage(&msg);  
        DispatchMessage(&msg);  
    }  
}
```

```
LRESULT WindowProc(HWND hwnd, UINT message,  
                    WPARAM wParam, LPARAM lParam) {  
    switch (message) {  
        case WM_KEYDOWN: /* process keyboard */ break;  
        case WM_LBUTTONDOWN: /* process mouse */ break;  
        case WM_TIMER: /* process timer */ break;  
    }  
}
```

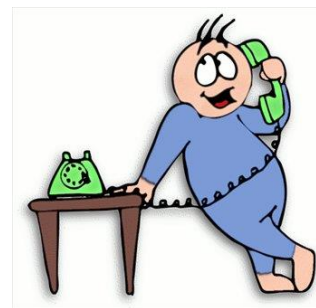
<https://docs.microsoft.com/en-us/windows/desktop/winmsg/about-messages-and-message-queues>

Пользовательский интерфейс: Функции отправки сообщений

BOOL PostMessage(HWND, UINT, WPARAM, LPARAM);



LRESULT SendMessage(HWND, UINT, WPARAM, LPARAM);



Пользовательский интерфейс: Функции обработки сообщений

```
BOOL WINAPI GetMessage( __out LPMMSG lpMsg,  
    __in_opt HWND hWnd,  
    __in UINT wMsgFilterMin,  
    __in UINT wMsgFilterMax );
```

извлекает сообщение из очереди сообщений
вызывающего потока и помещает его в
заданную структуру

```
BOOL WINAPI TranslateMessage( __in const MSG *lpMsg );
```

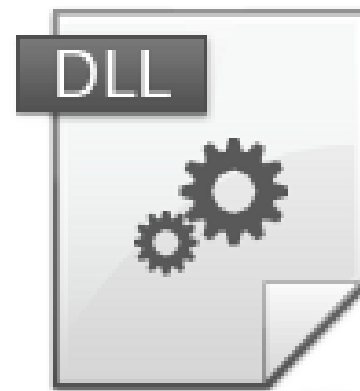
переводит сообщения формата виртуальных
клавиш в сообщения символы

```
LRESULT WINAPI DispatchMessage( __in const MSG *lpmsg );
```

пересылает сообщение оконной процедуре

Динамические библиотеки (dll-ки)

- ❑ Общие сведения
 - Предыстория,
 - Жизненный цикл разработки
 - Адресное пространство
 - Экспортируемые функции
 - Поиск dll
- ❑ Использование dll в коде
 - Неявное связывание
 - Явное связывание
 - ::Load/::Free Library
- ❑ Недостатки dll



Динамические библиотеки: Предыстория

Dynamic Link Library – концепция совместно используемых библиотек в ОС Windows.

- Изначально предназначено для эффективного управления памятью.
- В последствии стало средством для эффективной разработки ПО за счёт модульности.

Пример:

Photoshop.exe

ImageProc.dll

LicenseVerifier.dll

kernel32.dll

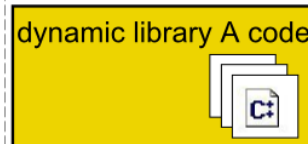
ImageEffects.dll

PluginManager.dll

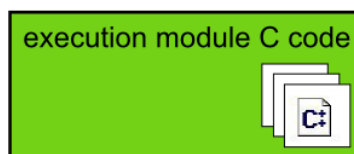
...

Динамические библиотеки: Жизненный цикл разработки

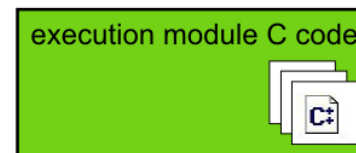
Статические
библиотеки
(LIB-ы)



Динамические
библиотеки
(DLL-ки)



Разработка



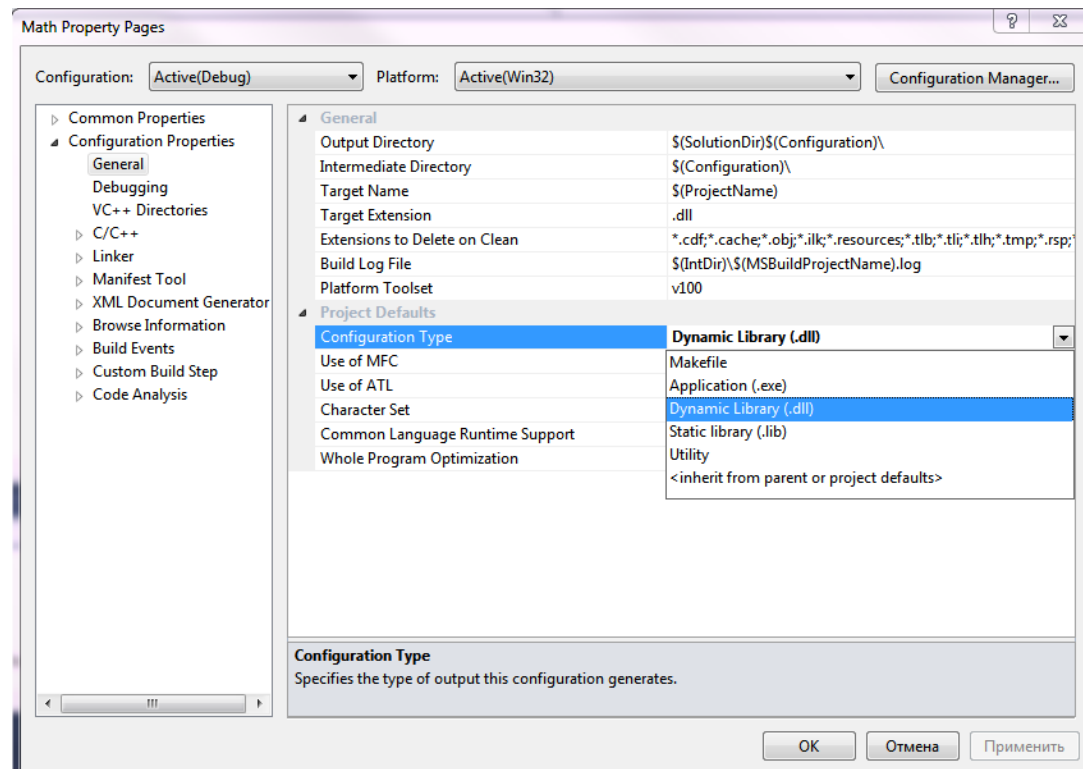
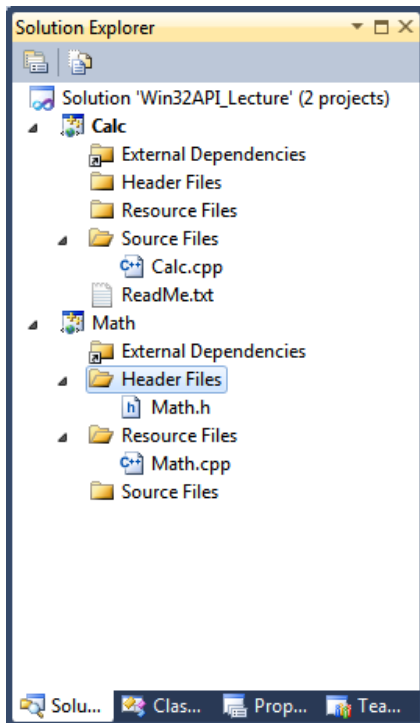
Компоновка



Поставка



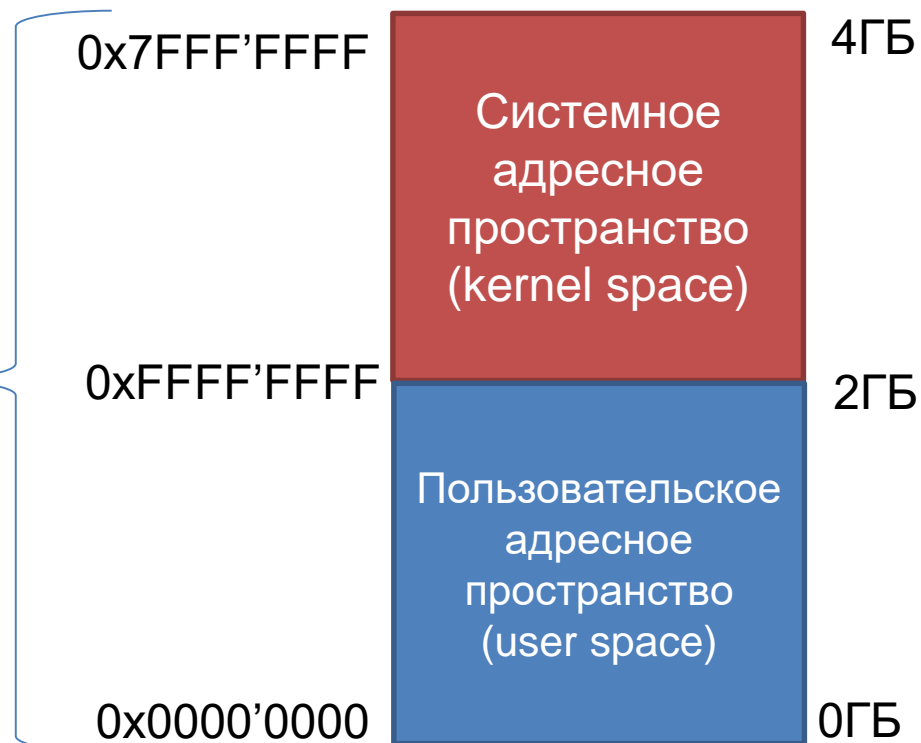
Динамические библиотеки: Жизненный цикл разработки



Динамические библиотеки: Адресное пространство

1. Создание процесса и выделение памяти

Адресное пространство процесса
(4 Гб для 32 бит)



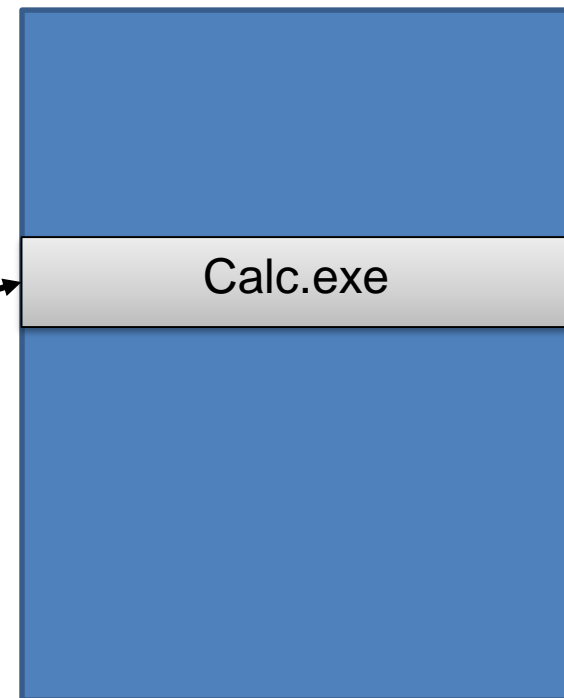
Динамические библиотеки: Адресное пространство

1. Создание процесса и выделение памяти
2. Загрузка EXE в память

Calc.EXE

```
void _cdecl func() {  
00411A20 55          push     ebp  
00411A21 8B EC       mov     ebp,esp  
00411A23 81 EC C0 00 00 00 sub     esp,0C0h  
00411A29 53          push     ebx  
00411A2A 56          push     esi  
00411A2B 57          push     edi  
00411A2C 8D BD 40 FF FF FF lea     edi,[ebp-0C0h]  
00411A32 B9 30 00 00 00 mov     ecx,30h  
00411A37 B8 CC CC CC CC mov     eax,0CCCCCCC  
00411A3C F3 AB       rep stos dword ptr [edi]  
}  
00411A3E 5F          pop      edi  
00411A3F 5E          pop      esi  
00411A40 5B          pop      ebx  
00411A41 8B E5       mov     esp,ebp  
00411A43 5D          pop     ebp  
00411A44 C3          ret
```

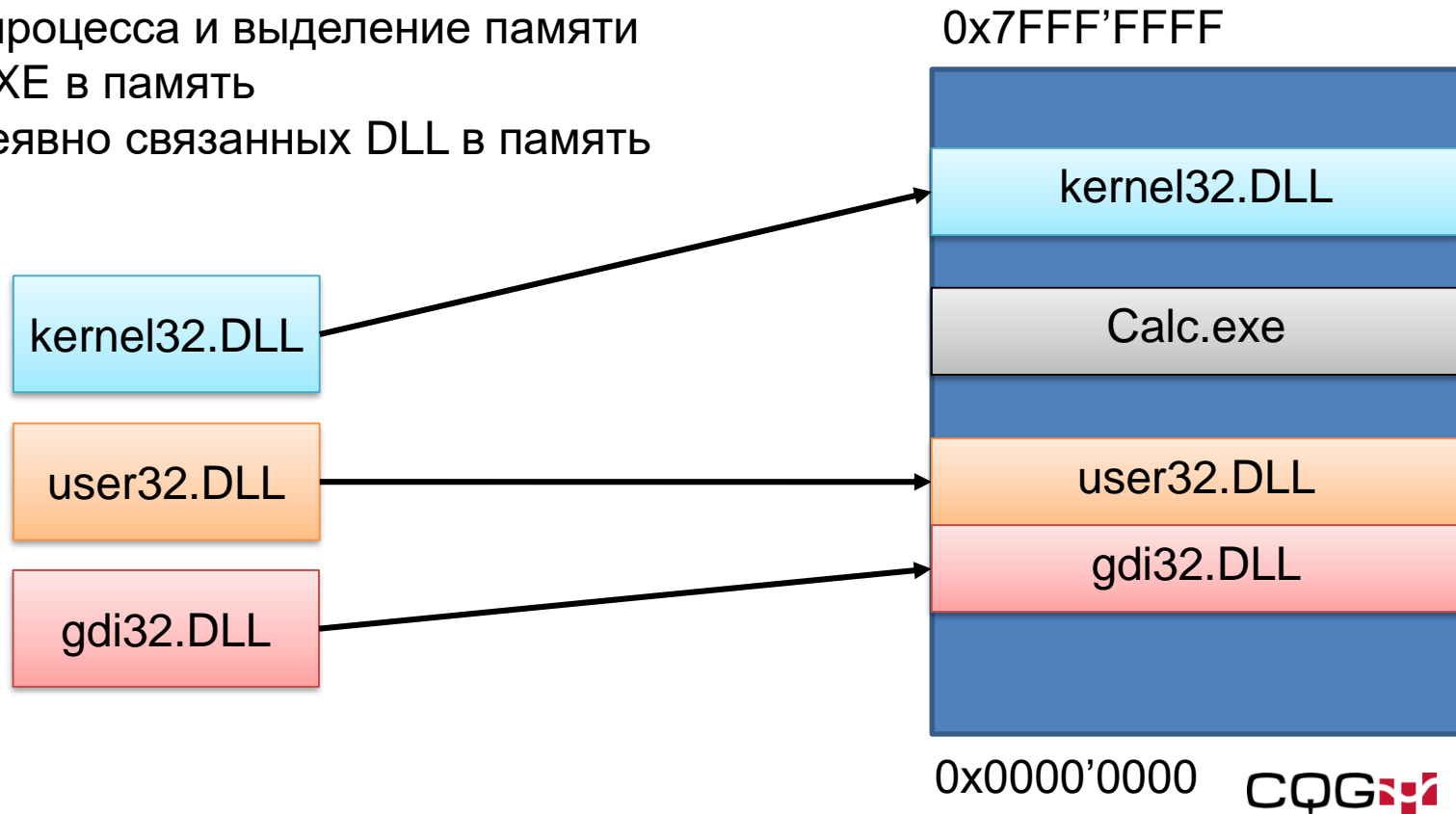
0x7FFF'FFFF



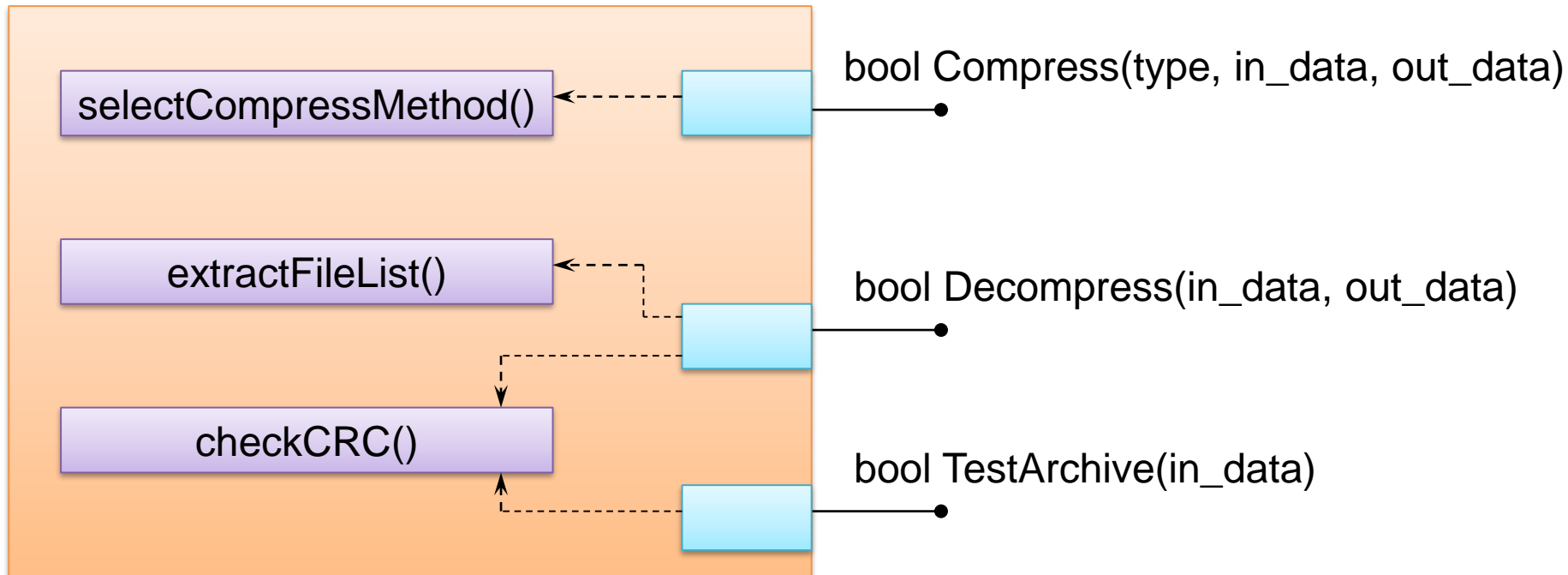
0x0000'0000

Динамические библиотеки: Адресное пространство

1. Создание процесса и выделение памяти
2. Загрузка EXE в память
3. Загрузка неявно связанных DLL в память

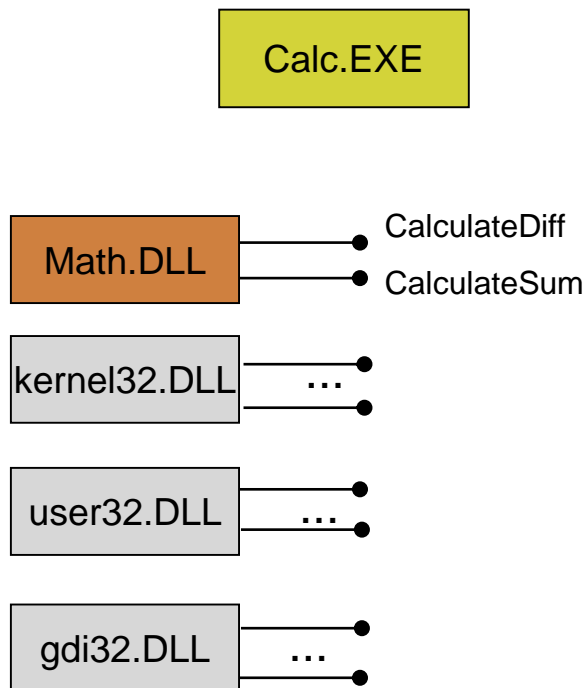


Динамические библиотеки: Экспортируемые функции



Динамические библиотеки: Экспортируемые функции

Адресное пространство процесса



```
D:\Projects\Exportimport>dumpbin -exports Math.dll
Microsoft (R) COFF/PE Dumper Version 7.10.6030
Copyright (C) Microsoft Corporation. All rights reserved.
```

Dump of file Math.dll

File Type: DLL

Section contains the following exports for Math.dll

00000000 characteristics
494018C6 time date stamp Wed Dec 10 22:30:14 2008
0.00 version
1 ordinal base
2 number of functions
2 number of names

ordinal	hint	RVA	name
1	0	0001C514	CalculateDiff
2	1	0001C5BE	CalculateSum

Summary

5000 .data
1000 .idata
7000 .rdata
3000 .reloc
3B000 .text
1B000 .textbss



Динамические библиотеки (dll-ки)

Использование dll в коде



Динамические библиотеки: Явное связывание (код)

calc.cpp

```
#include <iostream>
#include "Windows.h"

typedef int (*CalcFuncPtr)(int a, int b);

int main() {
    HMODULE h = ::LoadLibrary("math.dll");

    CalcFuncPtr calcSum =
        (CalcFuncPtr) ::GetProcAddress(h, "CalculateSum");
    CalcFuncPtr calcDiff =
        (CalcFuncPtr) ::GetProcAddress(h, "CalculateDiff");

    std::cout << calcSum(10, 15) << std::endl;
    std::cout << calcDiff(25, 12) << std::endl;

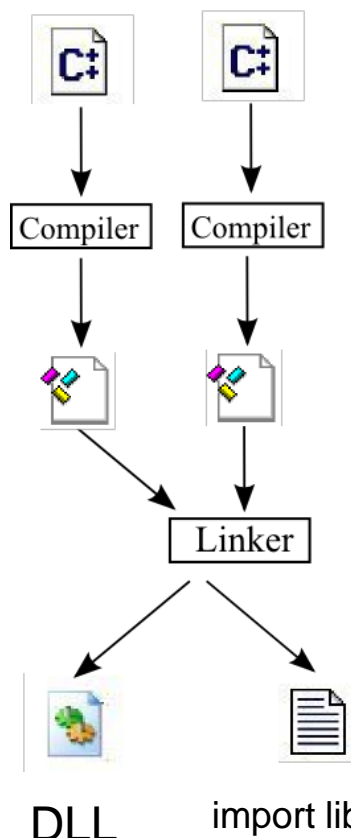
    ::FreeLibrary(h);
    return 0;
}
```

math.cpp

```
extern "C"
_declspec(dllexport) int CalculateSum(int a, int
b)
{
    return a + b;
}

extern "C"
_declspec(dllexport) int CalculateDiff(int a, int b)
{
    return a - b;
}
```


Динамические библиотеки: Явное связывание

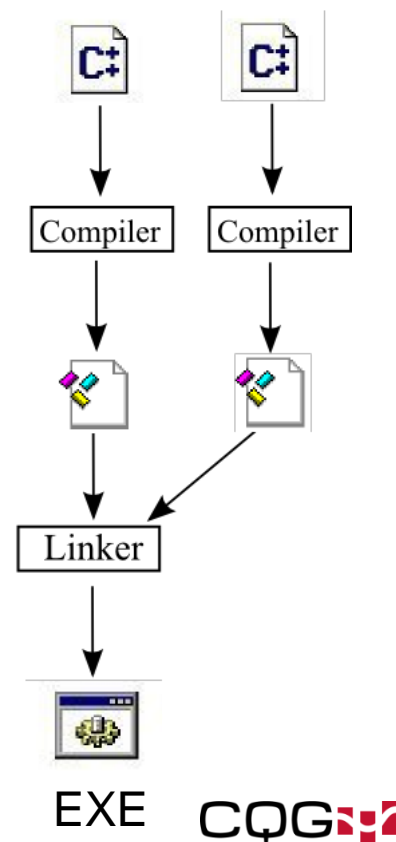


```
D:\Projects\ExportImport>dumpbin -imports Calc.exe
Microsoft (R) COFF/PE Dumper Version 7.10.6030
Copyright (C) Microsoft Corporation. All rights reserved.
```

Dump of file Calc.exe

File Type: EXECUTABLE IMAGE

Section contains the following **imports**:



Динамические библиотеки: Неявное связывание (код)

math.cpp

```
#define MATHAPI extern "C" __declspec(dllexport)
#include "math.h"

int CalculateSum(int a, int b)
{
    return a + b;
}

int CalculateDif(int a, int b)
{
    return a - b;
}
```

math.h

```
#ifndef MATHAPI
#define MATHAPI extern "C" __declspec(dllimport)
#endif

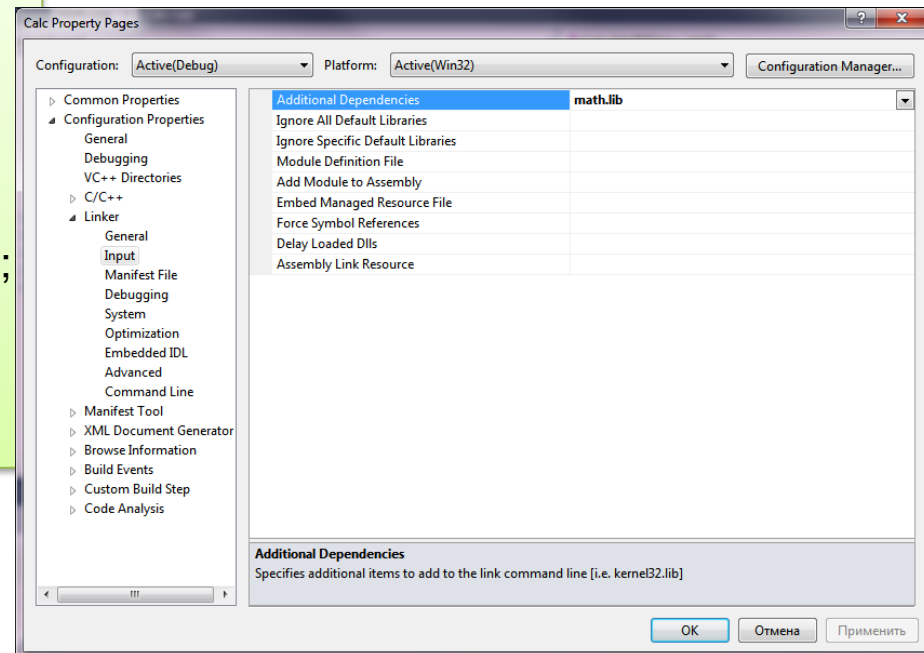
MATHAPI int CalculateSum(int a, int b);
MATHAPI int CalculateDif(int a, int b);
```

Динамические библиотеки: Неявное связывание (код)

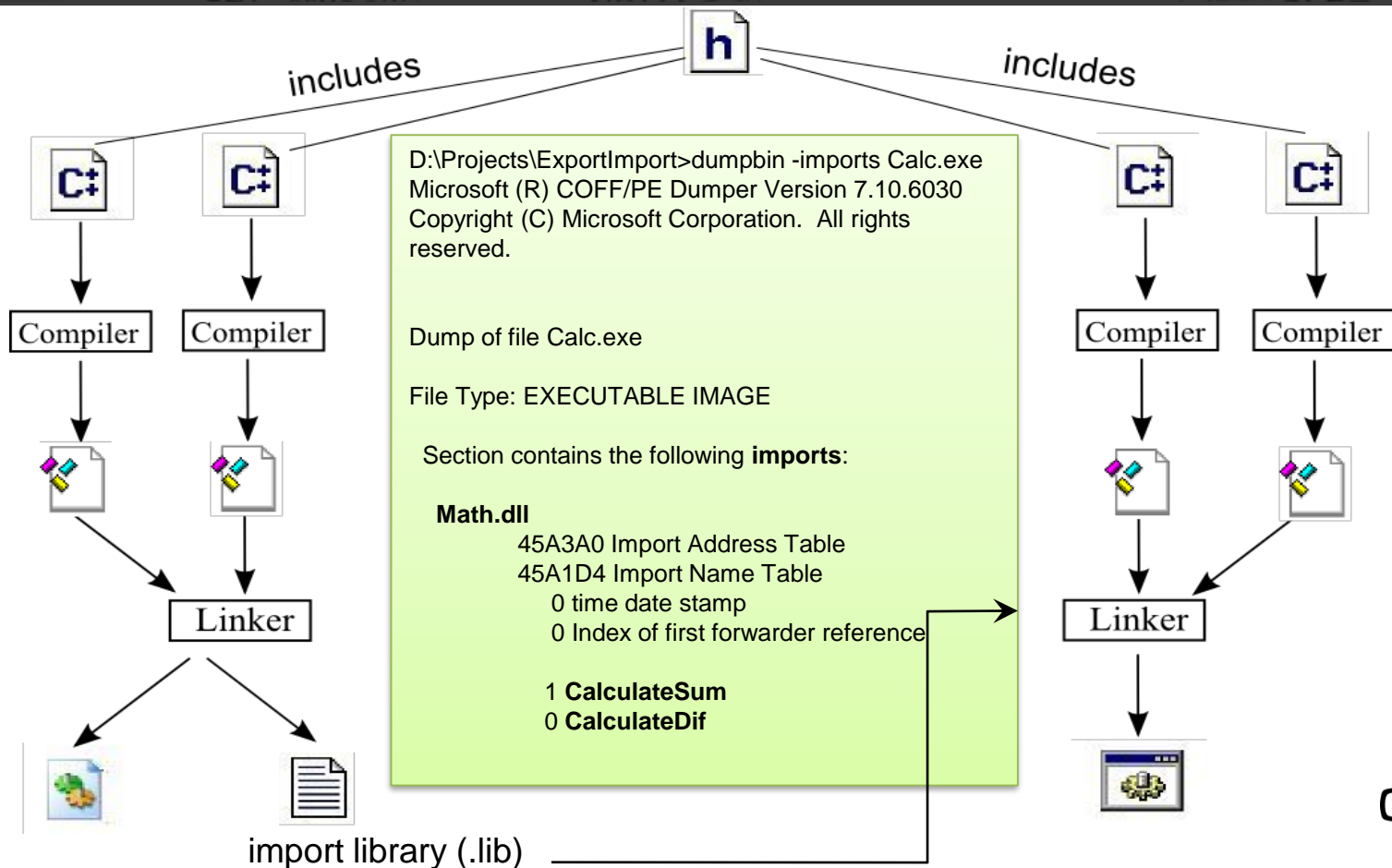
calc.cpp

```
#include <iostream>
#include "math.h"

int main()
{
    std::cout << CalculateSum(10, 15) << std::endl;
    std::cout << CalculateDif(25, 12) << std::endl;
    return 0;
}
```



Динамические библиотеки: Неявное связывание



Динамические библиотеки: Искажение имён (name mangling)

```
#ifndef MATHAPI
#define MATHAPI __declspec(dllimport)
#endif
```

```
MATHAPI int CalculateSum(int a, int b)
```

```
#define MATHAPI __declspec(dllexport)
#include "math.h"
```

```
int CalculateSum(int a, int b) {
    return a + b;
}
```

```
D:\Projects\Exportimport>dumpbin -exports Math.dll
Microsoft (R) COFF/PE Dumper Version 7.10.6030
Copyright (C) Microsoft Corporation. All rights reserved.
```

Dump of file Math.dll

File Type: DLL

Section contains the following exports for Math.dll

ordinal	hint	RVA	name
1	0	00011267	?CalculateSum@@YAHPAUSOMESTRUCT@@@Z

```
#ifndef MATHAPI
#define MATHAPI extern "C" __declspec(dllimport)
#endif
```

```
MATHAPI int CalculateSum(int a, int b)
```

```
#define MATHAPI extern "C" __declspec(dllexport)
#include "math.h"
```

```
int CalculateSum(int a, int b) {
    return a + b;
}
```

```
D:\Projects\Exportimport>dumpbin -exports Math.dll
Microsoft (R) COFF/PE Dumper Version 7.10.6030
Copyright (C) Microsoft Corporation. All rights reserved.
```

Dump of file Math.dll

File Type: DLL

Section contains the following exports for Math.dll

ordinal	hint	RVA	name
1	0	00011267	CalculateSum

Динамические библиотеки: API для функции работы с DLL

Загрузка:

HINSTANCE LoadLibrary(PCTSTR pszDllPathName);

↑
Адрес виртуальной памяти,
по которому спроецирован образ dll файла

Выгрузка:

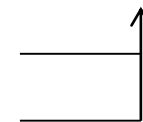
BOOL FreeLibrary(HINSTANCE hinstDll);

HINSTANCE GetModuleHandle(PCTSTR pszModuleHandle);

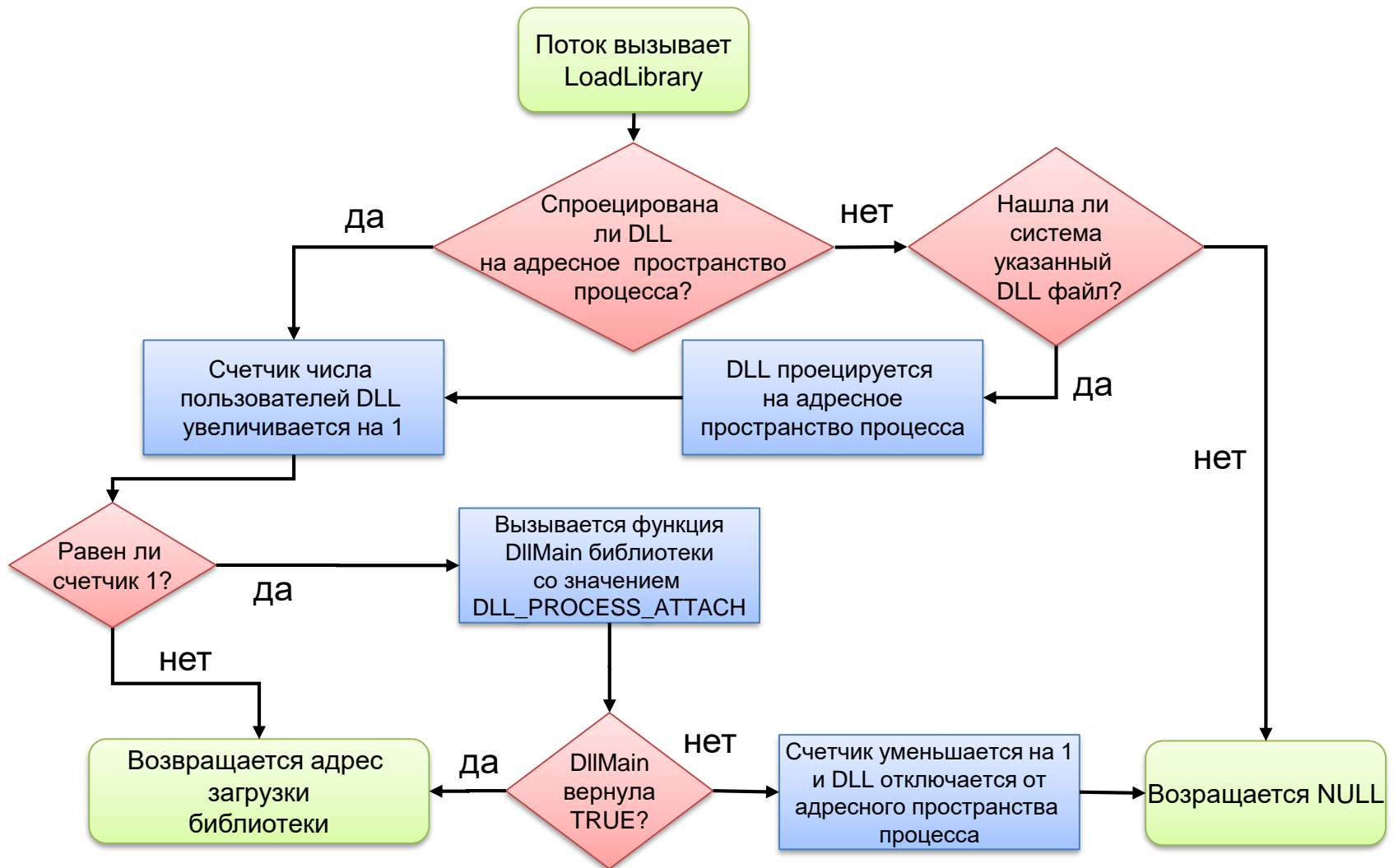
Адрес экспортируемой функции:

FARPROC GetProcAddress(HINSTANCE hinstDll, PCSTR pszSymbolName);

ANSI – имя функции
Порядковый номер (Оригинал)



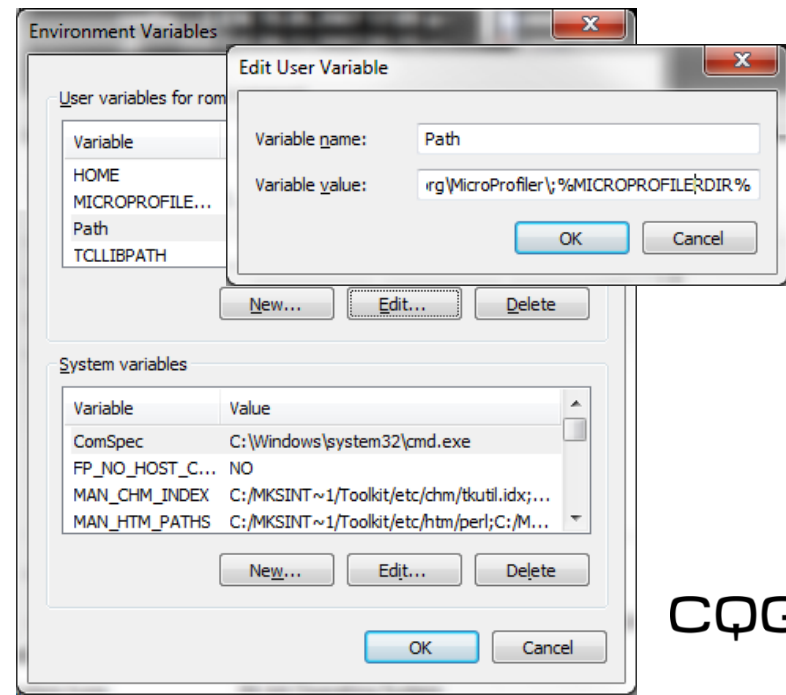
Динамические библиотеки: Алгоритм ::LoadLibrary



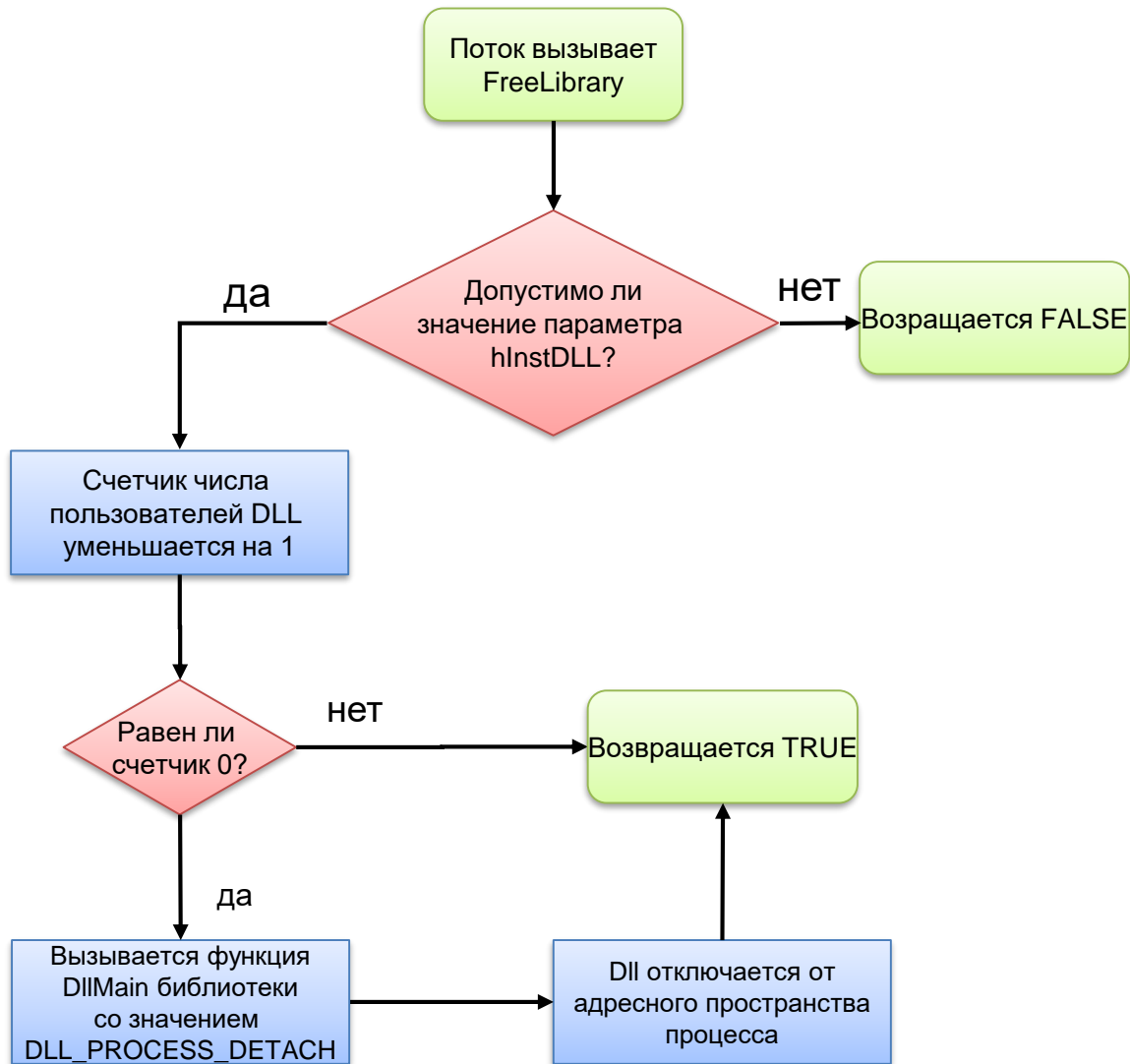
Динамические библиотеки: Алгоритм ::LoadLibrary (поиск dll)

Поиск DLL:

1. Если это известная системная dll, то загрузить из C:\Windows\System32
2. Папка, содержащая EXE-файл
3. Системная папка: C:\Windows\System32 или C:\Windows\SysWOW64
4. 16-bit системная папка: C:\Windows\System
5. Папка Windows: C:\Windows
6. Текущая папка
7. Папки из PATH



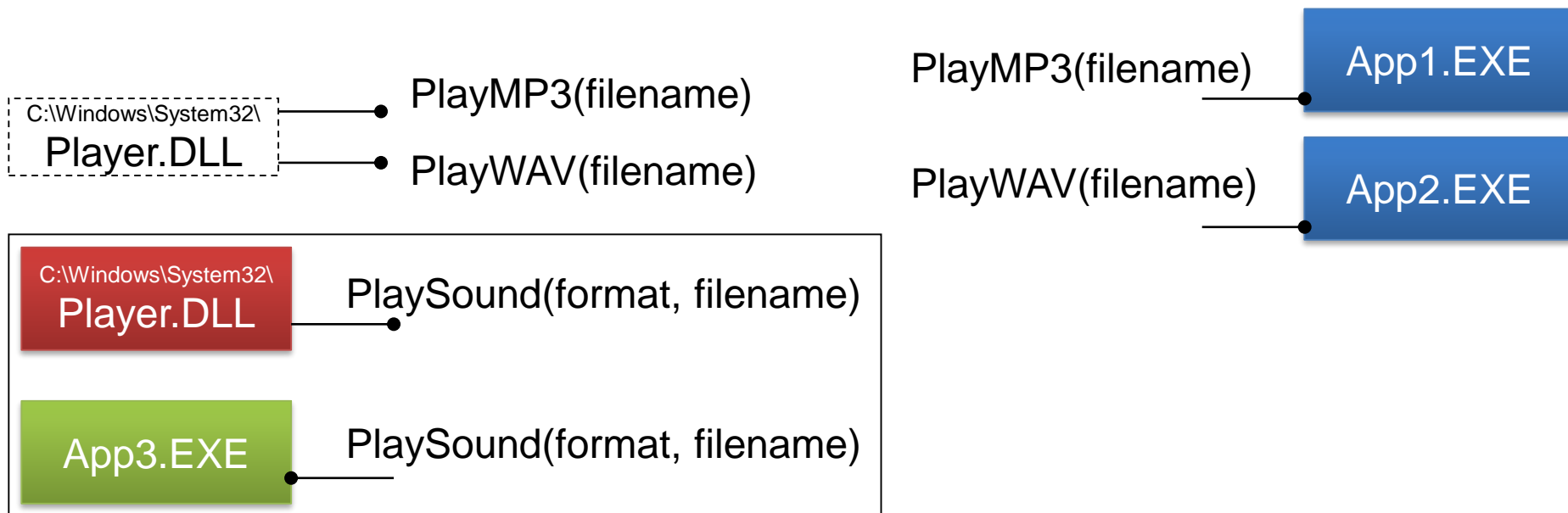
Динамические библиотеки: Алгоритм :: FreeLibrary



Динамические библиотеки: Функция входа/выхода (DllMain)

```
BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpReserved)
{
    switch( fdwReason )
    {
        case DLL_PROCESS_ATTACH: загрузка DLL в адресное пространство процесса
            break;
        case DLL_THREAD_ATTACH: создаётся поток
            break;
        case DLL_THREAD_DETACH: поток корректно завершается
            break;
        case DLL_PROCESS_DETACH: выгрузка DLL из адресного пространства процесса
            break;
    }
    return TRUE;
}
```

Динамические библиотеки: Недостатки DLL



- Не устанавливать DLL для частного использования в общие папки;
- Тщательное проектирование версионности на этапе разработки DLL;
- Использование статически-линкуемых библиотек.

Динамические библиотеки: Использование dumpbin + подмена DLL

```
Developer Command Prompt for VS 2017
D:\my_work\internship\2018\7. WinAPI и DLL\new\demo>z:
Z:\>dumpbin /rawdata /section:.text dll_main.dll
Microsoft (R) COFF/PE Dumper Version 14.16.27024.1
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file dll_main.dll

File Type: DLL

SECTION HEADER #1
.text name
  A virtual size
  1000 virtual address (10001000 to 10001000)
  200 size of raw data
  400 file pointer to raw data (00000400 to 000005FF)
  0 file pointer to relocation table
  0 file pointer to line numbers
  0 number of relocations
  0 number of line numbers
60000020 flags
  Code
  Execute Read
RAW DATA #1
10001000: 55 8B EC B8 00 20 00 00 5D C3
Summary
1000 .text
```

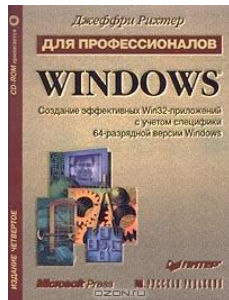
```
Developer Command Prompt for VS 2017
Z:\>dumpbin /disasm /section:.text dll_main.dll
Microsoft (R) COFF/PE Dumper Version 14.16.27024.1
Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file dll_main.dll

File Type: DLL

SECTION HEADER #1
.text name
  A virtual size
  1000 virtual address (10001000 to 10001000)
  200 size of raw data
  400 file pointer to raw data (00000400 to 000005FF)
  0 file pointer to relocation table
  0 file pointer to line numbers
  0 number of relocations
  0 number of line numbers
60000020 flags
  Code
  Execute Read
10001000: 55          push     ebp
10001001: 8B EC       mov     ebp,esp
10001003: B8 00 20 00 10 mov     eax,10002000h
10001008: 5D         pop     ebp
10001009: C3         ret
Summary
1000 .text
```


Дополнительные ресурсы



Джеффри Рихтер. Windows для профессионалов.
Создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows



<https://docs.microsoft.com/en-us/windows/desktop/DLLs/about-dynamic-link-libraries>



CppCon 2017: **James McNellis**
“Everything You Ever Wanted to Know about DLLs”



Q&A