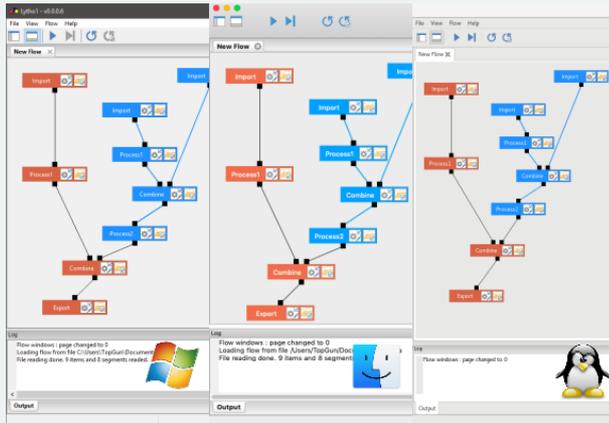




# Кроссплатформенная разработка программного обеспечения

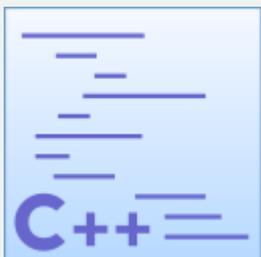


Лабораторная работа №3

Общие вопросы разработки ПО под различные платформы.

# Возможности использования кода в различных проектах

Исходный код



Скомпилированный код

Статическая библиотека



Qt5Core.lib

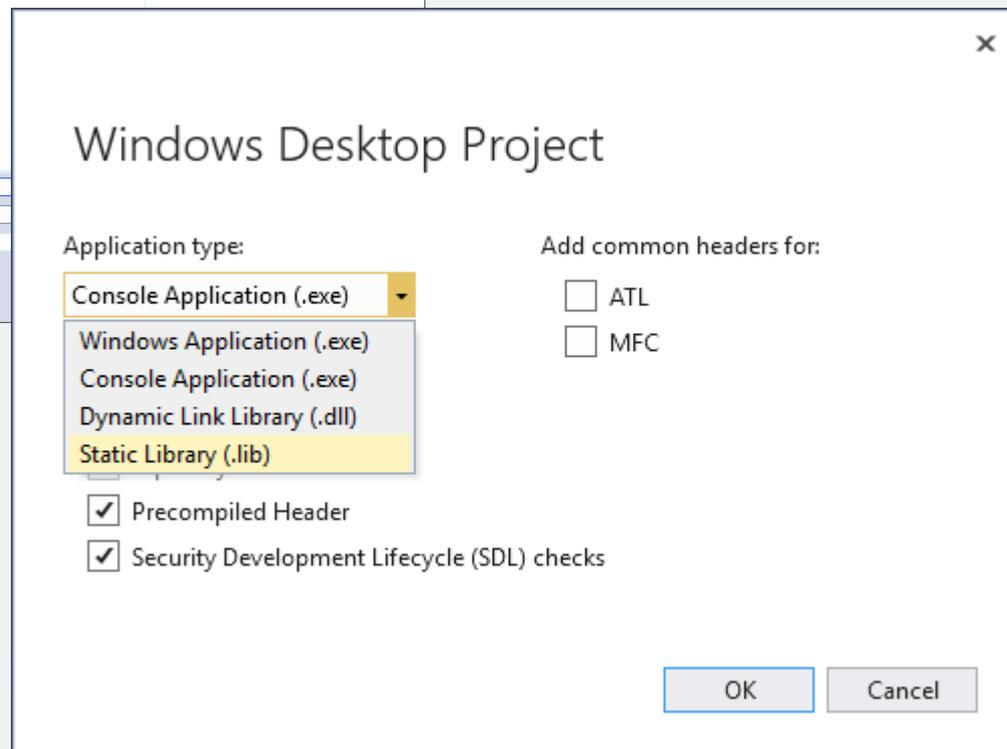
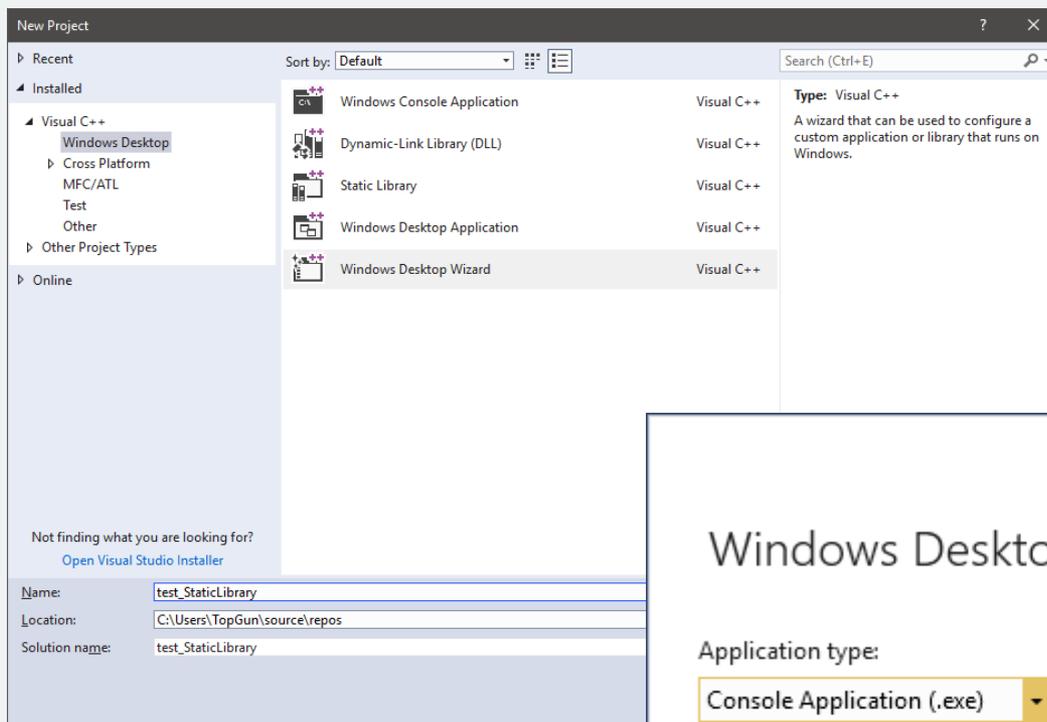
Динамическая библиотека



Qt5Core.dll

Обычно нужен заголовочный файл

# Создание статической библиотеки: создание проекта





## Создание статической библиотеки: код библиотеки

Файл main.cpp:

```
#include <iostream>
#include <locale>

void print_message() {
    setlocale(LC_CTYPE, "rus");
    std::cout << "Привет из статической библиотеки!" << std::endl;
}
```

Лог компилятора:

```
1>----- Rebuild All started: Project: test_StaticLibrary, Configuration: Debug Win32 -----
1>main.cpp
1>test_StaticLibrary.vcxproj -> C:\Users\...\test_StaticLibrary\Debug\test_StaticLibrary.lib
===== Rebuild All: 1 succeeded, 0 failed, 0 skipped =====
```



## Использование статической библиотеки: написание кода

Код:

```
#include <iostream>

#pragma comment(lib, "test_StaticLibrary.lib")

void print_message();

int main(int argc, char *argv[]) {
    print_message();
    return EXIT_SUCCESS;
}
```

# Использование статической библиотеки: подключение в проекте

test\_StaticLibCaller Property Pages

Configuration: Active(Debug) Platform: Active(Win32) Configuration Manager...

Configuration Properties

- General
- Debugging
- VC++ Directories
- C/C++
- Linker**
  - General
  - Input**
  - Manifest File
  - Debugging
  - System
  - Optimization
  - Embedded IDL
  - Windows Metadata
  - Advanced
  - All Options
  - Command Line
- Manifest Tool
- XML Document Generator
- Browse Information
- Build Events
- Custom Build Step
- Code Analysis

**Additional Dependencies** kernel32.lib;user32.lib;gdi32.lib;winspool.lib;comdlg32.lib;advapi32.lib

- Ignore All Default Libraries
- Ignore Specific Default Libraries
- Module Definition File
- Add Module to Assembly
- Embed Managed Resource File
- Force Symbol References
- Delay Loaded DLLs
- Assembly Link Resource

**Additional Dependencies**

test\_StaticLibrary.lib

Evaluated value:

test\_StaticLibrary.lib  
%(AdditionalDependencies)

Inherited values:

kernel32.lib  
user32.lib  
gdi32.lib

Inherit from parent or project defaults

Macros >>

OK Cancel

**Additional Dependencies**  
Specifies additional items to add to the link command line. [i.e. kernel32.lib]

OK Отмена Применить

## Создание статической библиотеки: код библиотеки

Файл main.cpp:

```
#include <iostream>
#include <locale>

void print_message() {
    setlocale(LC_CTYPE, "rus");
    std::cout << "Привет из статической библиотеки!" << std::endl;
}
```

Файл header.hpp:

```
#pragma once

void print_message();
```



## Использование статической библиотеки: написание кода

Код:

```
#include <iostream>
#include "Header.hpp"

int main(int argc, char *argv[]) {
    print_message();
    return EXIT_SUCCESS;
}
```

# Подключение статической библиотеки в проекте (1)

The screenshot shows the 'test\_StaticLibCaller Property Pages' dialog box. The 'Configuration' is set to 'Active(Debug)' and the 'Platform' is 'Active(Win32)'. The 'VC++ Directories' property is selected in the left-hand tree. The 'General' tab is active, showing the 'Include Directories' property with a value of '\$(VC\_IncludePath);\$(WindowsSDK\_IncludePath);'. A red circle with the number '1' points to the 'VC++ Directories' property in the tree. A red circle with the number '2' points to the 'Include Directories' property in the list. A red circle with the number '3' points to the 'Include Directories' dialog box, which is open and shows the path 'C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test\_StaticLibrary' selected in the file list. The 'Include Directories' dialog also shows the 'Evaluated value' as 'C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test\_StaticLibrary' and the 'Inherited values' as '\$(VC\_IncludePath)' and '\$(WindowsSDK\_IncludePath)'. The 'Inherit from parent or project defaults' checkbox is checked. The 'Include Directories' dialog has 'OK' and 'Cancel' buttons. The main Property Pages dialog has 'OK', 'Отмена', and 'Применить' buttons at the bottom.

## Подключение статической библиотеки в проекте (2)

The screenshot shows the 'test\_StaticLibCaller Property Pages' dialog box. The 'Configuration' is set to 'Active(Debug)' and the 'Platform' is 'Active(Win32)'. The left-hand tree view shows 'VC++ Directories' selected, with a red circle and arrow labeled '1' pointing to it. The 'General' tab is active, and 'Library Directories' is selected, with a red circle and arrow labeled '2' pointing to it. A 'Library Directories' dialog box is open, showing the path 'C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test\_StaticLibrary' selected, with a red circle and arrow labeled '3' pointing to it. The dialog box also shows the 'Evaluated value' and 'Inherited values' sections, and a checked 'Inherit from parent or project defaults' option.



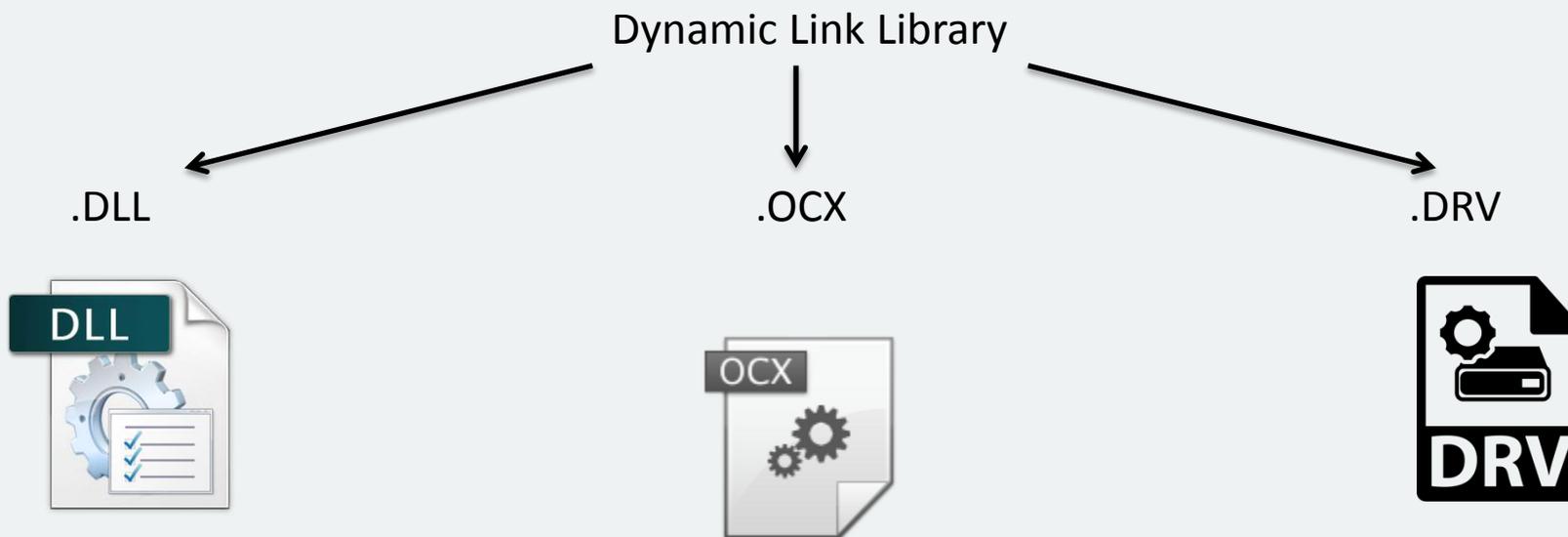
## Использование статической библиотеки: написание кода (2)

Код:

```
#include <iostream>
#include <Header.hpp>

int main(int argc, char *argv[]) {
    print_message();
    return EXIT_SUCCESS;
}
```

# Dynamic Link Library (DLL)





# Использование динамической библиотеки: создание проекта

Windows Desktop Project

Application type:  
Dynamic Link Library (.dll)

Add common headers for:  
 ATL  
 MFC

Additional Options:  
 Empty Project  
 Export Symbols  
 Precompiled Header  
 Security Development Lifecycle (SDL) checks

OK Cancel



## Разработка динамической библиотеки: код

Файл main.cpp:

```
#include <iostream>
#include <locale>

__declspec(dllexport) void print_message() {
    setlocale(LC_STYPE, "rus");
    std::cout << "Привет из динамической библиотеки!" << std::endl;
}
```

Лог компилятора:

```
1>----- Build started: Project: test_DLL, Configuration: Debug Win32 -----
1>main.cpp
1>test_DLL.vcxproj -> C:\Users\...\test_DLL\Debug\test_DLL.dll
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
```

# Разработка динамической библиотеки: файл определений

Файл определений `test_DLL.def` :

```
LIBRARY "test_DLL"  
EXPORTS  
    print_message
```

The screenshot shows the 'test\_DLL Property Pages' dialog box in Visual Studio. The 'Configuration' is set to 'Active(Debug)' and the 'Platform' is 'Active(Win32)'. The 'Linker' section is expanded, and the 'Input' sub-section is selected. A red circle labeled '1' points to the 'Linker' section, a red circle labeled '2' points to the 'Input' sub-section, a red circle labeled '3' points to the 'Module Definition File' property, and a red circle labeled '4' points to the 'Module Definition File' dialog box. The dialog box shows the file name 'test\_DLL.def' entered in the text field. The 'Module Definition File' dialog box also has 'test\_DLL.def' listed in the file list below the text field. The 'Module Definition File' section at the bottom of the Property Pages dialog box contains the text: 'The /DEF option passes a module-definition file (.def) to the linker. Only one .def file can be specified to LINK.'



## Использование динамической библиотеки: написание кода

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

typedef void(*func_ptr)();

int main(int argc, char *argv[]) {
    setlocale(LC_CTYPE, "rus");
    HMODULE hDll = LoadLibrary(L"test_DLL.dll");
    if (!hDll) {
        std::cout << "Не могу загрузить файл DLL :(" << std::endl;
        return EXIT_FAILURE;
    }

    func_ptr func = (func_ptr)GetProcAddress(hDll, "print_message");
    if (func != NULL)
        func();

    FreeLibrary(hDll);

    return EXIT_SUCCESS;
}
```



## Компиляция кода в Linux

```
#include <stdio.h>

void main() {
    printf("%s", "PKIMS compiles under linux! :)");
}
```

Самый простой пример:

```
g++ ./main.cpp
```

Компиляция по шагам :

```
g++ -c -o ./main.o ./main.cpp
```

```
g++ -o ./main ./main.o
```

Компиляция в один шаг:

```
g++ -o ./main ./main.cpp
```

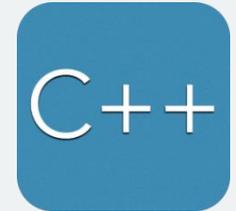


# Работа с динамическими библиотеками под ОС Linux (1)

Код библиотеки (library.cpp):

```
#include <iostream>

extern "C" void print_message() {
    std::cout << "Hello from the .so!" << std::endl;
}
```



task\_01

## Работа с динамическими библиотеками под ОС Linux (2)

Код вызывающей программы (main.cpp):

```
#include <dlfcn.h>
#include <iostream>

typedef void(*func_ptr)();

int main(int argc, char *argv[]) {
    void *handle = dlopen("./library.so", RTLD_LAZY);
    if (!handle) {
        std::cout << "Can't load shared object" << std::endl;
        return 1;
    }
    func_ptr hello = (func_ptr)dlsym(handle, "print_message");
    if (hello != NULL)
        hello();

    dlclose(handle);

    return 0;
}
```





## Компиляция библиотеки в Linux

Компиляция динамической библиотеки:

```
g++ -shared -o library.so library.cpp
```

Компиляция вызывающей программы:

```
g++ -o main main.cpp -ldl
```



## Работа с Makefile в Linux (1)

```
all:                main

main:               func.o main.o
                  @echo Building all together
                  g++ func.o main.o -o main

func.o:             func.cpp func.h
                  @echo Building func
                  g++ -Wall -c func.cpp

main.o:             main.cpp
                  @echo Building main
                  g++ -Wall -c main.cpp

clean:
                  rm -f *.o
                  rm -f ./main
```



**task\_03**

## Задание на лабораторную работу

Реализовать модули решения и записи результатов моделирования.

Метод решения:

Номер варианта	Поддерживаемый формат
1, 4, 7, 10, 13	Метод Гаусса
2, 5, 8, 11, 14	Метод Якоби
3, 6, 9, 12, 15	Метод Гаусса-Зейделя

Формат записи результатов моделирования:

Номер варианта	Поддерживаемый формат
1, 4, 7, 10, 13	PSF
2, 5, 8, 11, 14	CSV
3, 6, 9, 12, 15	CSDF