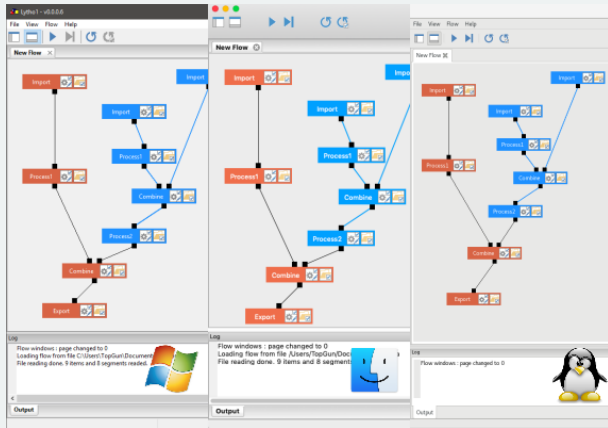




Кроссплатформенная разработка программного обеспечения

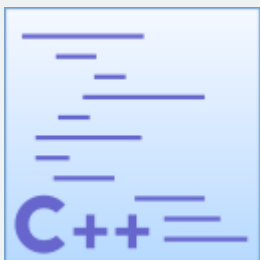


Лабораторная работа №3

Общие вопросы разработки ПО под различные платформы.

Возможности использования кода в различных проектах

Исходный код



Скомпилированный код

Статическая библиотека



Qt5Core.lib

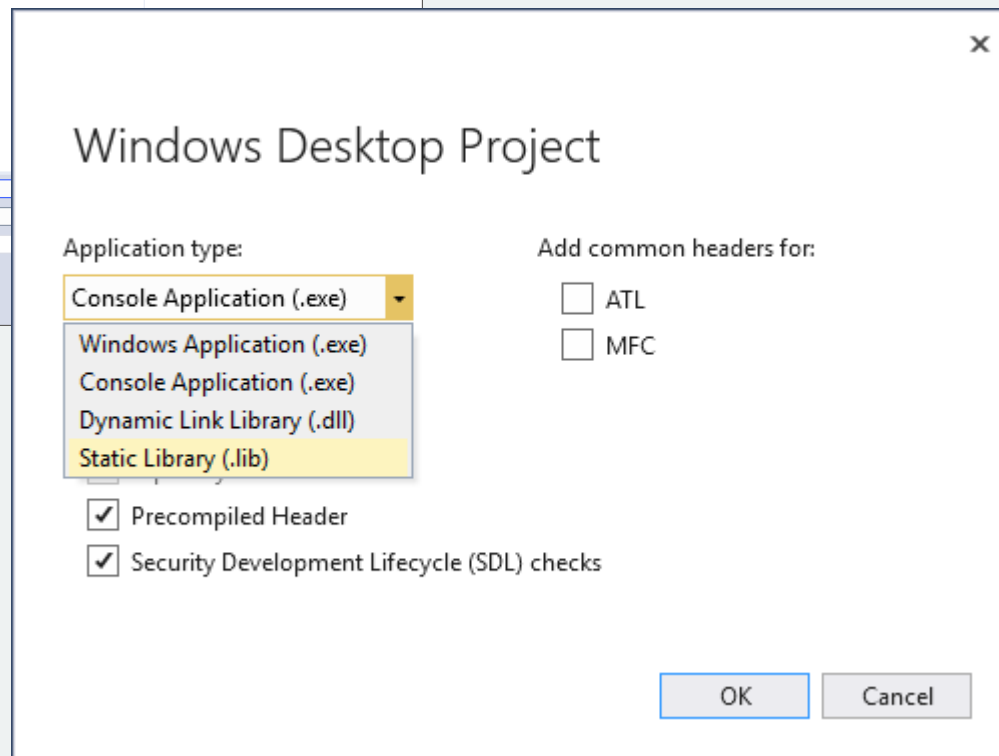
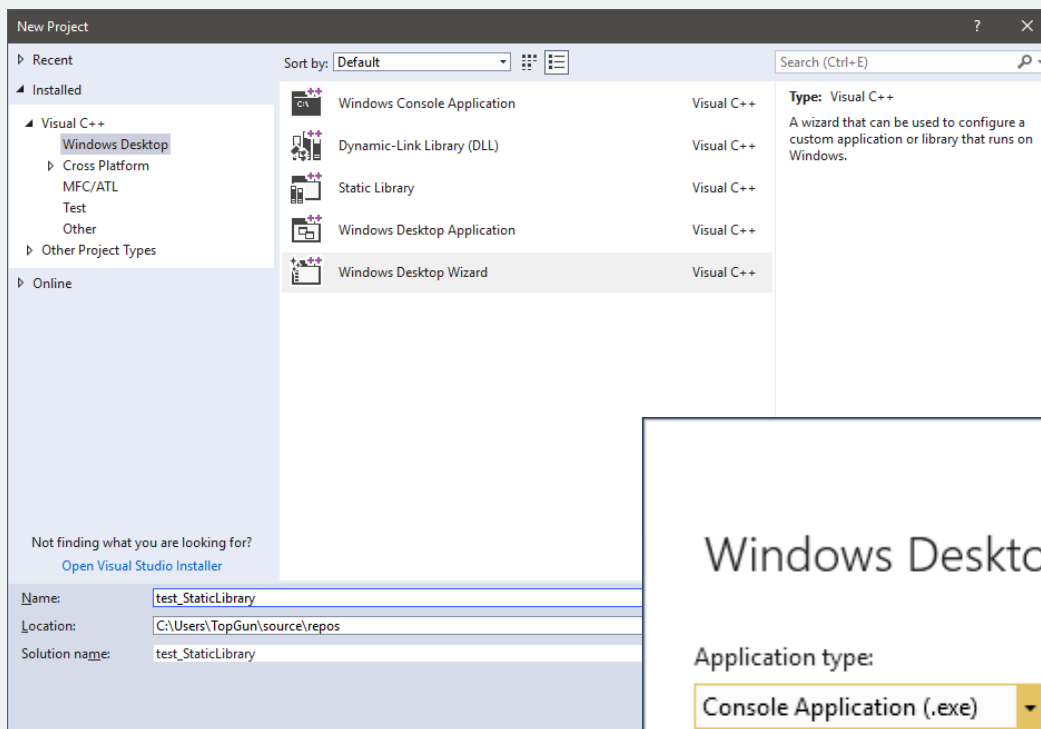
Динамическая библиотека



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 pane shows the 'Configuration Properties' tree with 'VC++ Directories' selected. The right pane shows the 'General' properties, with 'Library Directories' selected. A dialog box for editing the 'Library Directories' property is open, showing the path 'C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test_StaticLibrary'. Three red circles with numbers 1, 2, and 3 are overlaid on the image, with arrows pointing to the 'VC++ Directories' property, the 'Library Directories' property, and the path in the dialog box, respectively.

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

General

Executable Directories	\$(VC_ExecutablePath_x86);\$(WindowsSDK_ExecutablePath);\$(VS_ExecutablePath_x86);\$(VC_ExecutablePath_x86_amd64_x-arm64);\$(WindowsSDK_ExecutablePath_x86_amd64_x-arm64);\$(VS_ExecutablePath_x86_amd64_x-arm64)
Include Directories	C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test_StaticLibrary\Include
Reference Directories	\$(VC_ReferencesPath_x86);
Library Directories	\$(VC_LibraryPath_x86);\$(WindowsSDK_LibraryPath_x86);\$(NETFXKits_LibraryPath_x86);\$(VC_LibraryPath_x86_amd64_x-arm64);\$(WindowsSDK_LibraryPath_x86_amd64_x-arm64);\$(NETFXKits_LibraryPath_x86_amd64_x-arm64)
Library WinRT Directories	\$(WindowsSDK_MetadataPath);
Source Directories	\$(VC_SourcePath);
Exclude Directories	

Library Directories

C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test_StaticLibrary

Evaluated value:
C:\Users\TopGun\Documents\Visual Studio 2017\Projects\test_StaticLibrary

Inherited values:
\$(VC_LibraryPath_x86)
\$(WindowsSDK_LibraryPath_x86)

Inherit from parent or project defaults

Path to use when LIB. is not specified: LIB.

OK Cancel

OK Отмена Применить



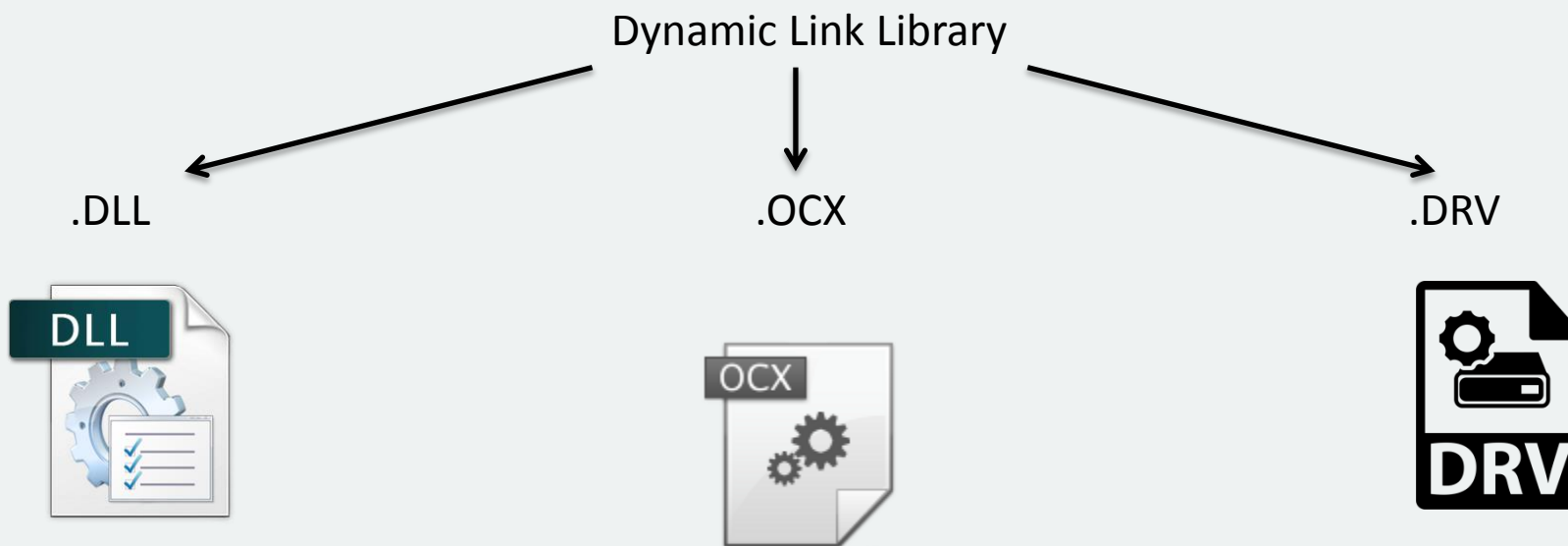
Использование статической библиотеки: написание кода (2)

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 box highlights the 'Module Definition File' property in the 'Input' section. A red arrow points from circle 1 to this property. Another red arrow points from circle 2 to the 'Module Definition File' property in the 'Additional Dependencies' section. A third red arrow points from circle 3 to the 'Module Definition File' property in the 'Additional Dependencies' section. A fourth red arrow points from circle 4 to the 'Module Definition File' dialog box, which is open and shows 'test_DLL.def' entered in the text field. The dialog box has 'OK' and 'Cancel' buttons. At the bottom of the Property Pages dialog, there are 'OK', 'Отмена', and 'Применить' buttons.



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;
}
```



task_01



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