

**DLL**

**Dynamic Link Library**

- При запуске нескольких экземпляров одного приложения, Windows загружает в оперативную память только одну копию кода и ресурсов - модуль приложения, создавая несколько отдельных сегментов данных, стека и очереди сообщений, каждый набор которых представляет из себя задачу, в понимании Windows.
- Копия приложения представляет из себя контекст, в котором выполняется модуль приложения.
- DLL - библиотека также является модулем. Она находится в памяти в единственном экземпляре и содержит сегмент кода и ресурсы, а также сегмент данных

- DLL - библиотека, в отличие от приложения не имеет ни стека, ни очереди сообщений. Функции, помещенные в DLL, выполняются в контексте вызвавшего приложения, пользуясь его стеком. Но эти же функции используют сегмент данных, принадлежащий библиотеке, а не копии приложения
- Экономия памяти достигается за счет того, что все запущенные приложения используют один модуль DLL, не включая те или иные стандартные функции в состав своих модулей.

# Создание DLL в Delphi (экспорт)

- *library ProjectDLL;*
- *{ информативные строк.}*
- *uses*
- *SysUtils,*
- *Classes;*
- *{\$R \*.RES}*
- *exports*
- *begin*
- *end.*

# Exports

- *exports*
- *Func1 index 10 name 'Fun',*
- *Func3 index 11,*
- *Func4 index 11, //Ошибка, такой индекс уже существует*
- *Func5 name 'Don';*

Объявлять можно и так:

- *exports Func1 index 10 name 'Fun',*
- *exports Func2 Insert,*
- *exports Func3 index 11*

Резидентные

- *exports*
- *ExportByName name 'MYEXPORTPROC' resident;*

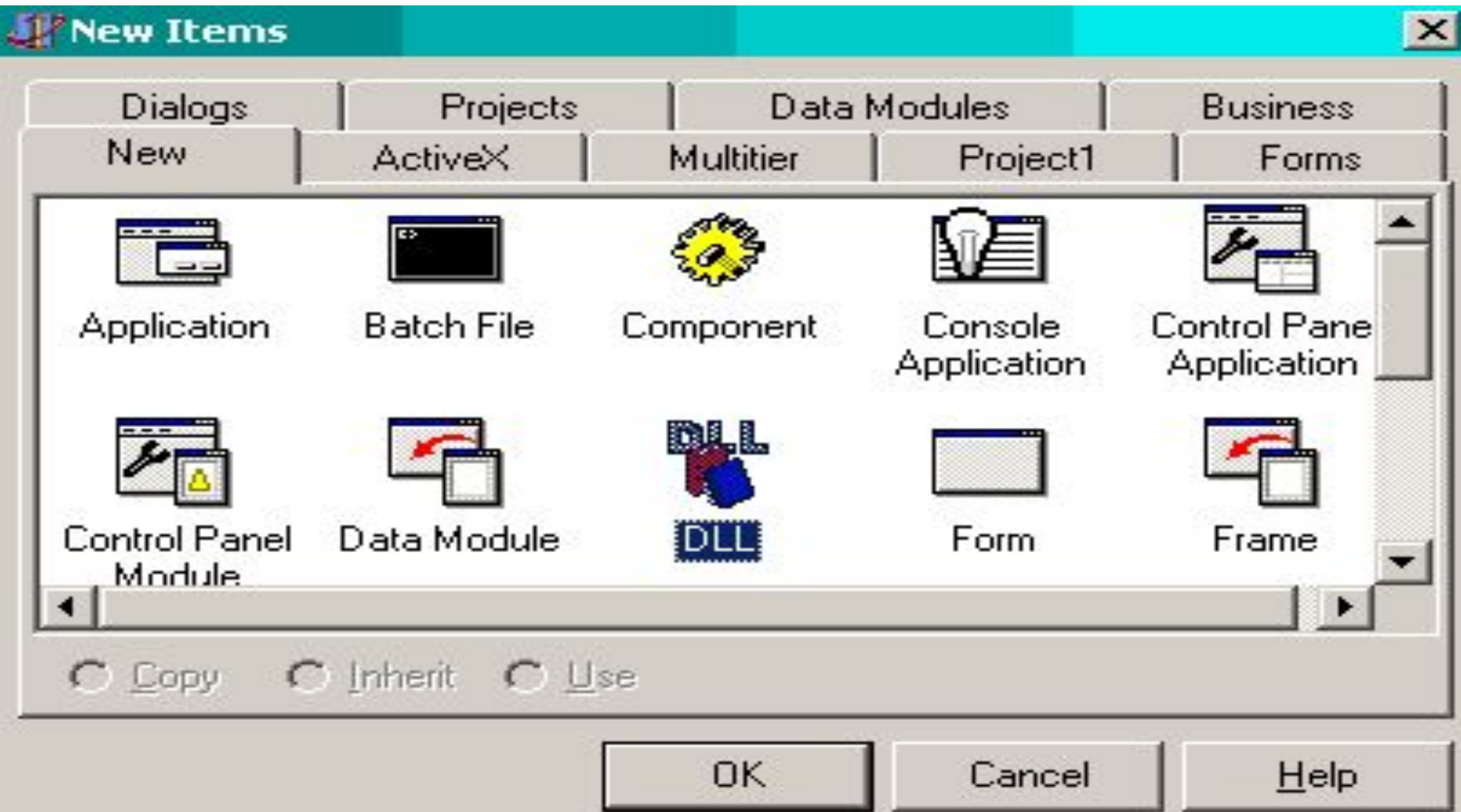
# Использование DLL в Delphi (импорт)

- В вашей программе следует объявить функции, импортируемые из DLL таким образом:
- `procedure ImportByName; external 'MYDLL' name 'MYEXPORTPROC';`
- `procedure ImportByOrdinal; external 'MYDLL' index 10;`
- `procedure MyExportFunc1; external 'MYDLL';`
- Этот способ называется статическим импортом.

# Динамическая загрузка dll

```
type TMyProc = procedure(Handle: THandle); stdcall;  
    процедурный тип функции подгружаемый из библиотеки;  
var  
    Handle : THandle; - указатель на библиотеку  
    MyImportProc : TMyProc; - подгружаемая функция  
begin  
    Handle:=LoadLibrary('MYDLL');-динамическая загрузка DLL  
    @MyImportProc:=GetProcAddress(Handle,'MYEXPORTPROC');  
    Получение адреса точки входа нужной функции  
    FreeLibrary(Handle); - Освобождение ресурса  
end;
```

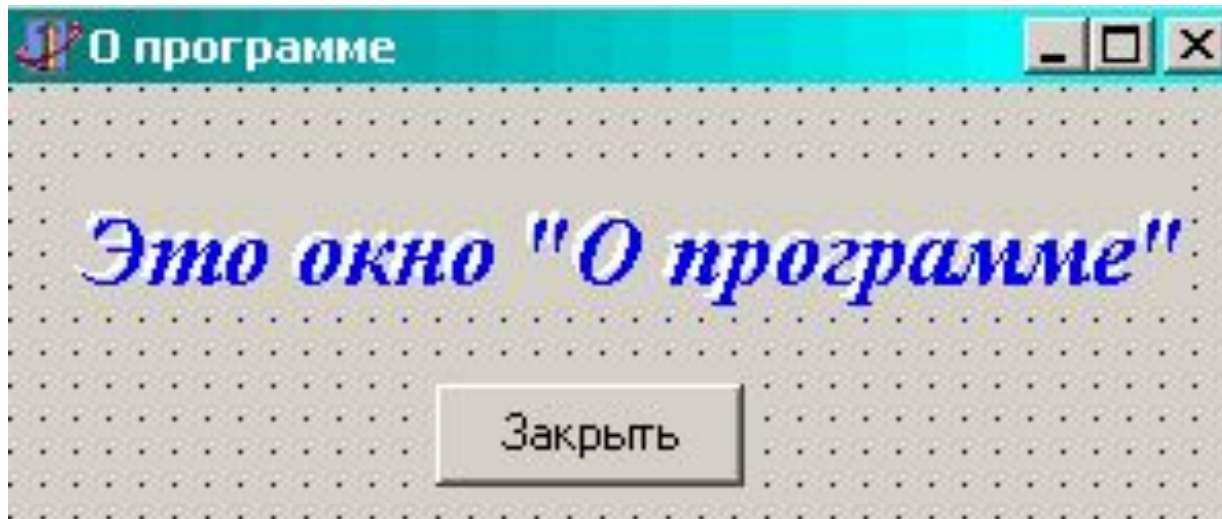
# Пример





- *library ProjectDLL;*
- *uses*
- *SysUtils, Classes;*
- *{ \$R \*.RES }*
- *exports ShowAbout index 10;*
- *begin*
- *end.*

- *File->New Form*



# ТЕКСТ МОДУЛЯ

```
var  
  Form1: TForm1;  
  procedure ShowAbout(Handle: THandle);export;stdcall;
```

после *implementation* и ключа {\$R \*.DFM}:

```
procedure ShowAbout(Handle: THandle);  
begin  
  //Установить указатель на приложение  
  Application.Handle := Handle;  
  //Создать форму  
  Form1:= TForm1.Create(Application);  
  //Отобразить  
  Form1.ShowModal;  
  //Очистить  
  Form1.Free;  
end;
```

# В новом проекте

***unit Unit2;***

***interface***

***uses***

***Windows, Messages, SysUtils, Classes, Graphics, Controls,  
Forms, Dialogs, StdCtrls;***

***procedure ShowAbout(Handle: THandle)stdcall;***

***type***

***TForm1 = class(TForm)***

***Button1: TButton;***

***procedure Button1Click(Sender: TObject);***

***private***

***{ Private declarations }***

***public***

***{ Public declarations }***

***end;***

***var***

***Form1: TForm1;***

***procedure ShowAbout;external 'ProjectDLL.dll' index 10;***

***implementation***

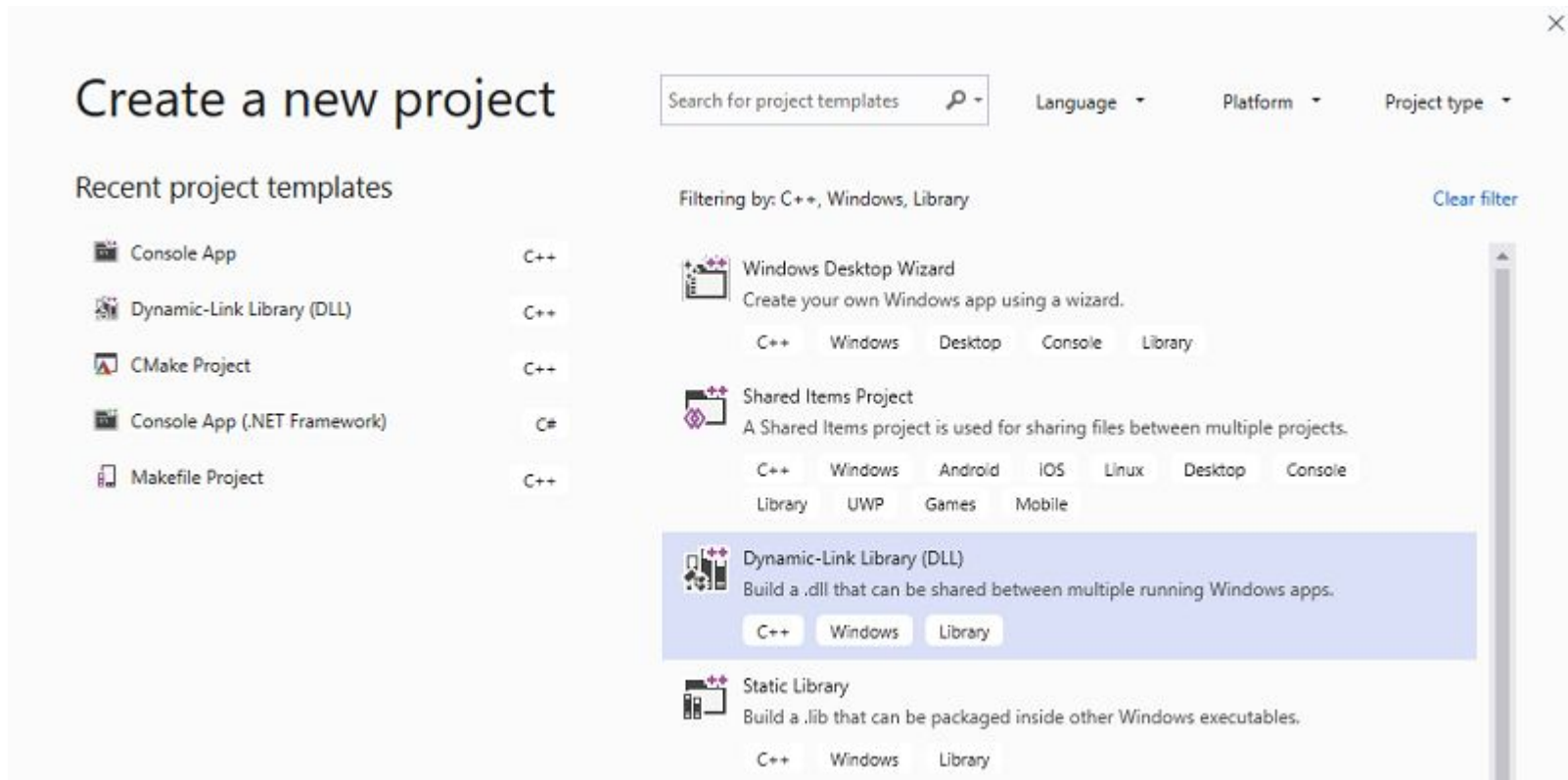
# Вызов функции из DLL

- Теперь поместим на форму кнопку и создадим для неё следующее событие:
- ***procedure***  
***TForm1.Button1Click(Sender: TObject);***
- ***begin***
- ***ShowAbout(Handle);***
- ***end;***

- library MyFirstDLL;
- uses SysUtils, Classes, Forms, Windows;  
procedure HelloWorld(AForm : TForm); begin
- MessageBox(AForm.Handle, Hello world!', DLL  
Message Box', MB\_OK or  
MB\_ICONEXCLAMATION);
- end;
- exports HelloWorld;
- begin
- end.

# Создание проекта библиотеки DLL в Visual Studio 2019

- **Файл > Создать > Проект, чтобы открыть диалоговое окно Создание**



- В верхней части диалогового окна для параметра **Язык** установите значение **Язык C**,
- для параметра **Платформа** — значение **Windows**,
- для параметра **Тип проекта** — значение **Библиотека**.
- В отфильтрованном списке типов проектов щелкните **Библиотека динамической компоновки (DLL)** , а затем нажмите кнопку **Далее**.



- На странице **Настроить новый проект** введите *MathLibrary* в поле **Имя проекта**.
- Примите заданные по умолчанию **Расположение** и **Имя решения**.
- Для параметра **Решение** задайте **Создать новое решение**. Снимите флажок **Разместить решение и проект в одном каталоге**, если он установлен.
- Нажмите кнопку **Создать**, чтобы создать проект.

- Затем вы создадите файл заголовка для объявления функций, экспортируемых вашей библиотекой DLL, и добавите определения функций в библиотеку DLL, чтобы сделать ее более полезной.

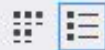
- Чтобы создать файл заголовка для функций, последовательно щелкните **Проект > Добавить новый элемент**.
- В диалоговом окне **Добавление нового элемента** в левой области щелкните **Visual C++** . В центральной области выберите **Заголовочный файл (.h)** . Укажите *MathLibrary.h* в качестве имени для файла заголовка.

Add New Item - MathLibrary



- Installed
  - Visual C++
    - Code
    - Data
    - Resource
    - Web
    - Utility
    - Property Sheets
    - ATL
    - HLSL
    - Graphics
- Online

Sort by: Default



Search (Ctrl+E)

- C++ File (.cpp) Visual C++
- Header File (.h) Visual C++
- C++ Class Visual C++

**Type:** Visual C++  
Creates a C++ header file

Name: MathLibrary.h

Location: C:\Users\username\Source\Repos\MathLibrary\MathLibrary\

Browse...

Add

Cancel

- Нажмите кнопку Добавить, чтобы создать пустой файл заголовка, который отображается в новом окне редактора.

- // MathLibrary.h
- - Contains declarations of math functions  
#pragma once
- #ifndef MATHLIBRARY\_EXPORTS
- #define MATHLIBRARY\_API  
\_\_declspec(dllexport)
- #else #define MATHLIBRARY\_API  
\_\_declspec(dllimport)
- #endif

- В обозревателе решений щелкните узел **Файлы решения** правой кнопкой мыши и выберите пункты
- **Добавить > Новый элемент.** Создайте новый CPP-файл с именем *MathLibrary.cpp*, аналогично добавлению нового файла заголовка на предыдущем шаге.

- В окне редактора выберите вкладку **MathLibrary.cpp**, если она уже открыта. Если нет, то в **обозревателе решений** дважды щелкните файл **MathLibrary.cpp** в папке **Исходные файлы** проекта **MathLibrary**.
- Запишите код функции