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

- 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;

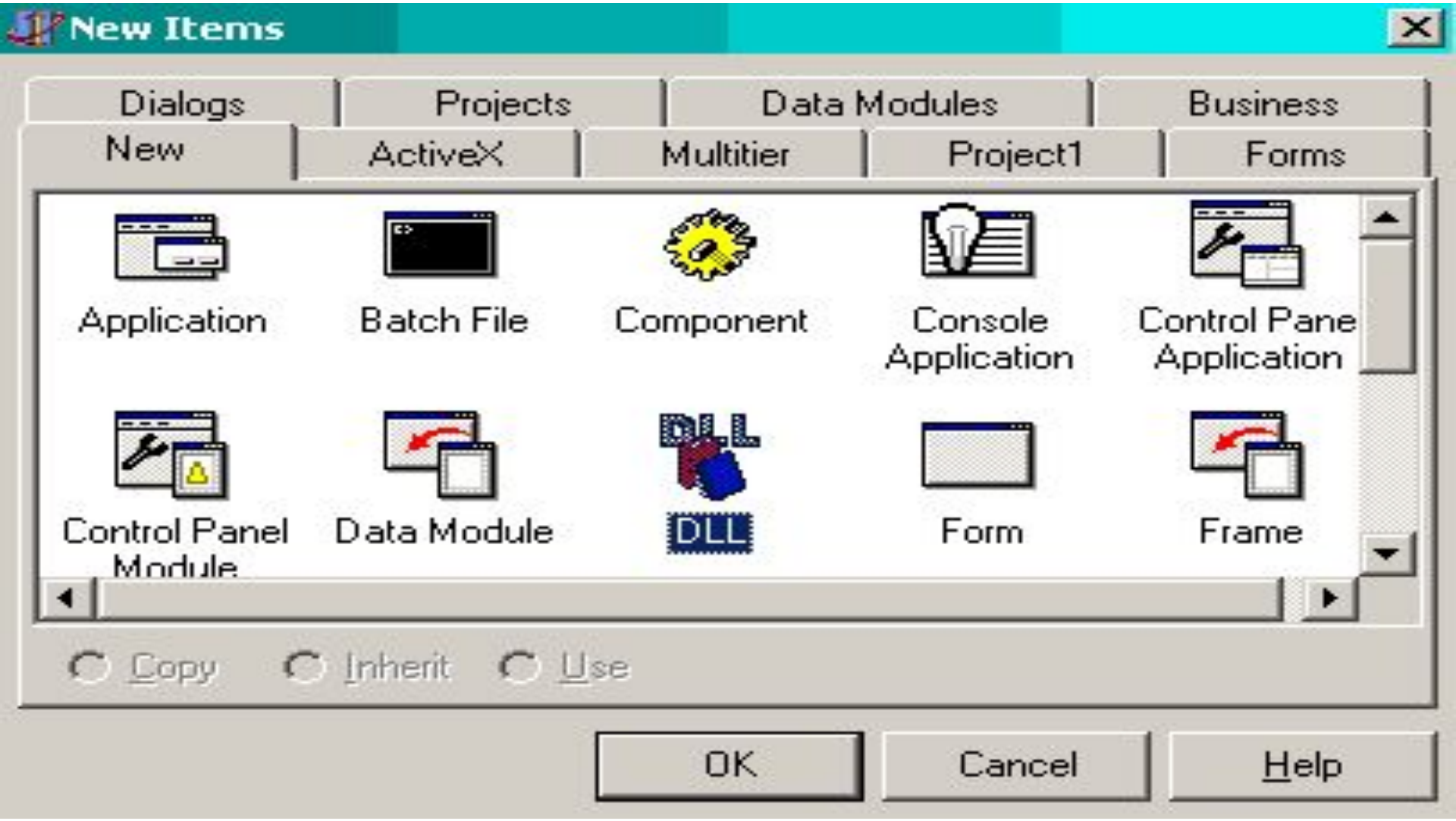
Использование DLL в Delphi (импорт)

- 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

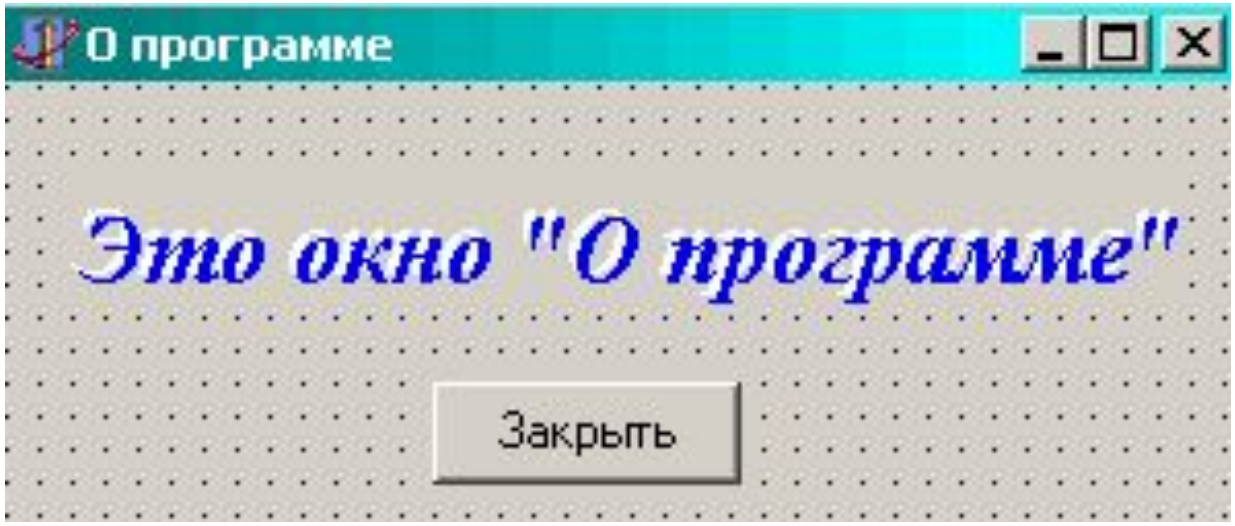
type TMyProc = procedure(Handle: THandle); stdcall;
процедурный тип функции подгружаемый из библиотеки;
var
Handle : THandle; - *указатель на библиотеку*
MyImportProc : TMyProc; - *подгружаемая функция*
begin
Handle:=LoadLibrary('MYDLL');-*динамическая загрузка DLL*
@MyImportProc:=GetProcAddress(Handle,'MYEXPORTPROC');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

Пример


New Items
Dialogs
Projects
Data Modules
Business
New
ActiveX
Multitier
Project1
Forms
Application
Batch File
Component
Console Application
Control Pane Application
Control Panel Module
Data Module
DLL
Form
Frame
Copy
Inherit
Use
OK
Cancel
Help

- *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.